

Instructor (Stephen Boyd): They should install it before then. Okay. Great. We'll start [cuts out] announcements, the first which we put on the website also, which is the authoritative source of information is that this week's section which would have been yesterday will in fact be today. As a matter of fact, it'll be at pretty much the worst possible time. It'll be at 11:00 a.m. today, in other words almost immediately following this lecture. So it'll be 11:00, 191 Skilling, so I guess that's upstairs. That's the section this week. Let's see another announcement is just to remind you if you were a victim of EE Quals and have filled out the proper paperwork, then your Homework 1 is due today. Then you should jump right into Homework 2. And we're working on Homework 3 for you, so it should be – Homework 3'll be nice.

Okay. I'm trying to think of this. Any other announcements? Oh, there's one other. In the next week, we're making the transition in the class – actually, even in today's lecture from sort of boring background material to actually stuff that's useful. So that's actually gonna happen in the next week. But part of that is that by next Monday and by Homework 3, there'll actually be a software component to the class. So next Monday's section – I'll announce this again, but next Monday's section will be on CVX, so we'll explain how that works and so on. It's actually quite well documented. I would recommend if you get bored or something like that, if Homework 2 isn't quite enough for you, or something like that, you get bored or who knows what, what you should do is install CVX. Now actually you should install it I'd say periodically as we silently fix errors, and you can watch the build number increment. But you should go ahead and install it, maybe read the manual. This is just if you get in some weird mood where you feel like doing this, you should do that. Anyway, you won't have a choice. By next Monday, you will have to install it and start reading the manual and all that kind of stuff. And it'll tie into everything you've seen so that should be okay. So that's not really – it's just looking ahead. The other of course is I should mention that the reading assignment is that you should be reading Chapter 4. Actually, hopefully you should have skimmed it already at the very least because that's what we're gonna work on today, and probably the next lecture as well. So any questions about last time? Otherwise, we'll zoom forward and actually get to – actually, it's gonna almost be useful soon, maybe even today.

Student: Are you gonna discuss generalized inequalities?

Instructor (Stephen Boyd): Am I gonna discuss generalized inequalities? No. That was the question. That was my answer, too. No. No, it's clear enough in the book. And when we get to something where it's relevant, like experiment design – it'll also be relevant in detection and estimation, then I'll go back over it. Also in multi-criterion optimization which we're gonna do later. So I'll go back over it. Okay. So today we're gonna go through optimization problems. The first part is a bit boring. It's just setting down basic terminology like what does it mean to be feasible, what are the constraint sets, actually what is a convex optimization problem, but then it'll transition to actually useful, so you actually find out about what a linear program is, what a quadratic program is, second order cone program, and these types of things. We'll get to those maybe later – hopefully,

this lecture. So we'll start with just the boring nomenclature of an optimization problem. So an optimization problem looks like this. This is a notation for it is to minimize an objective, $f_0(x)$ subject to $f_i(x) = 0$. That's from $i = 1, \dots, m$, and $h_i(x) = 0$. So this is sort of a standard form for an optimization problem.

By the way, if you read the Russian, the Soviet literature, you'd see different things, which is actually – I guess many Russians are now teaching in U.S. and European universities, and it's now propagated and you might as well know what it looks like. They would write it this way. It's actually a perfectly good notation like that. So you will see this, and it means that $f_0(x)$, the objective is to be minimized. So I guess I don't know how to pronounce this in Russian, but it's something like that. So you'll see this, and you'll see some other ones. Now you'll also see some things that drive me insane, and I'll mention one of them. You'll see this, and that really drives me insane because the semantics is wrong. Min actually has a meaning. Like you have the min – min takes as argument a set of numbers. In the strict definition, min takes as argument a finite set of numbers, and it returns a number. Minimize is actually a verb. It's not a mathematical operator, and it's part of the verb that you use to create an optimization problem, so these are different things. But you'll see this.

Actually, sometimes they just barely escape by putting a period at the end, which suggests that it wasn't min. It was short for minimize. But anyway, you'll see that. Sloppy. Now here x is the optimization variable. This is called the cost. It's sometimes called the objective. It's got all sorts of other names or – if it's in a specific application area, it'll have a name, too. Then these are called the inequality constraint functions, and the h s are the equality constraint functions. Now here the right hand sides are zero, but we're gonna see very quickly how you would – if the right hand side weren't zero, how you could transform it. So it's just useful to have a standard form like this.

Now the optimal value of this problem – so the type of object or data what this is that's an optimization problem without the min, or with the min period. That's a problem. By the way, it has no numeric – by itself, you can't say this problem is three or something like that. But you can talk about the optimal value of a problem, and that's simply the infimum or minimum of $f_0(x)$ over all x that satisfy the equality constraints. And there's a completely standard convention that the infimum of an empty set is infinity, so that's absolutely standard. That says you'd say that the optimal value of this problem is infinity. If there's no feasible x , if there's no x that satisfies these inequalities and equalities. And that makes sense because if your objective is to minimize something, then the value plus infinity basically means you've done infinitely poorly. You could not have done more poorly if your job was to minimize something if the value is infinity. And that is after all what infeasibility means. It means you didn't even get to the point where you could actually compare two points and say yours is better than mine because there weren't any that met the hard constraints. Okay. Now on the other hand, this could be minus infinity, and that would be the case if there were a sequence of x s all feasible with $f_0(x)$ k say going to minus infinity.

By the way, in different problems this is referred to different ways. For example, in something called risk sensitive estimation, they have a wonderful name for that when the problem is unbounded below. It's called euphoric breakdown. That's the technical name for that case. They have all sorts of other names for it in different fields and things like that. I guess it's infinitely good, I guess. What it really means is you probably set up – your modeling of your problem is not that great, but okay. So mentioned the idea of optimal and locally optimal points. Well, you say a point is feasible if it's in the domain of the objective and it satisfies the constraint. By the way, if it satisfies the constraints, it must also be in the domain of the constraint functions because you certainly wouldn't say it satisfies the constraints if it didn't also – to satisfy the constraints, $f_i(x)$ actually has to return a number and not the NID token, the not in domain token because there's no way, if that returned not in domain, that you could actually say that the inequality was valid. Okay? Okay. And a point is optimal if it's feasible and if its objective value is equal to the optimal value of the problem. That's optimal. Capital X_{opt} not totally standard notation, but I think fairly standard. That's the set of optimal points. And this can be empty, for example if the problem is infeasible of course, it's empty. If the problem is unbounded below, this is empty. And it can be even when p^* is finite, this can be empty. So there's – you can have everything, in which case –

By the way, to say that X_{opt} is the empty set, the way people would say that is you'd say the optimal value is not achieved is what you'd say. If it's finite, it – I'll give you an example. Let's just minimize $1/x$ on the positive axis. So if you minimize $1/x$, the optimal value is obviously zero, but it's not achieved because there's no x for which $1/x$ – there's no x in $\mathbb{R}^{++}$ for which $1/x$ is equal to zero. So you can get as close as you like and so on and so forth. So in this case, you would have X_{opt} is empty, and you would say that's a problem with optimal value zero which is not achieved. Now you say a problem is locally optimal if when you add a constraint – notice that this depends on x – if you add a constraint, that you not stray far in norm from x . $\mathbb{R}$ has to be positive here. If you can add such a constraint, and x becomes optimal, then it's locally optimal. And then these are kind of – you can just work through these. These are very straightforward. So if you minimize minus log, it's unbounded below obviously because I just take infinitely large x s and the sequence of x s getting larger and larger, negative log gets smaller and smaller.

Entropy here – that negative entropy – that actually has a perfectly well-defined minimum that occurs at $1/e$. You just differentiate, set it equal to zero, and the optimal value is $-1/e$. So this would be one with none of the pathologies here. If you have $f_0(x) = x^3 - 3x$, here it's unbounded below. Obviously, as x goes to minus infinity, this thing goes to minus infinity, so it's unbounded below, but you have a local optimum at $x = 1$. So if you take a look at this, you will find that around $x = 1$, this thing goes like that. The derivative is zero, and in a little neighborhood around it, actually it's locally convex there, and that means as you stray from $x = 1$, the objective value goes up, so that's locally optimal. I'm going quickly over this because it's kinda boring. And then when we get to the actual specific problem classes, it's more interesting. Now when you have this standard form problem, when you write this down, there's the implicit condition – this basically says this function is what you're gonna use to compare to proposed points that

meet the basic requirements. The basic requirements are that you should be feasible, but there's even something even more basic.

You have to be in the domain of all the f_i s, the objective and the constraints, and you have to be in the domain of all of the constraint and objective functions because if you're not in the domain of them, it means that one of these things evaluates to – it doesn't make sense. And you would certainly not say something is feasible if one of these things – if you were out of the domain of one of these. So there's an implicit constraint that the only x s you'll consider are in the domain of all functions, all objective and constraint functions, the objective and all constraints. So that's sometimes called an implicit constraint because you don't write it down explicitly. Actually, merely by writing down the symbols f_i and then called on x , you are actually imposing the constraint that x is in the domain of f_i . So that's why it's implicit. Later we'll see there's actually a difference between implicit and explicit, and it will make a difference, but it's something to think about. When you write out the constraints explicitly like this, these are called explicit constraints. And you say a problem is unconstrained if it has no explicit constraints. Here would be a very common example, one in fact that we'll see a great deal of. It's minimize the following function. It's the sum of the negative $\log(b_i - a_i^T x)$. Now to talk about the log of something, at least if you're not in a complex variables course, not in the context of a complex variables course, to talk about the log of something, this has to be positive.

And in fact, our definition of log which would differ of course from the definition of log used in a complex variables course, complex analysis course – our definition of log has a domain that is $\mathbb{R}^{++}$. So merely by stating $\log$ – if I write down $\log(u)$, I don't even have to put that in a constraint. That has a side effect, and the side effect – I'm talking about a side effect the way when you call a function. That has a side effect, and the side effect is that requires you to be positive, so that's the way it will work – I mean this is all standard, but that's the idea. So here, these have to be positive here. Now to say that $b_i - a_i^T x$ is positive says that there's actually a positive slack in the inequalities $a_i^T x < b_i$ like that. So the set of x that satisfies this is actually an open polyhedron. So it's an open polyhedron where this even makes sense, the domain of this – open polyhedron. Now on that open polyhedron, this function here is convex, smooth, everything. So what you would say is if someone asks you, "What about this problem," you'd say, "Well, that problem is unconstrained." I guess maybe the right – if you really wanted to be – specify, you'd say it's formally unconstrained. It has no constraint functions. That's what you'd say.

Of course, there's an implicit – and then if someone says, "Unconstrained? Does that mean I can throw in any x I like?" The answer to that is no. You can throw in any x that satisfies this, and that's an implicit constraint inherited from the domain of f_0 . Okay. Let's see. Let's talk about the feasibility problem. One variation on the optimization problem is this, and it's sometimes written this way. Find x subject to – by the way, you will also sometimes see subject to written as such that, which I don't actively – That's a question of style, and by the way, this is bad style. But nevertheless, it's within the – it's only in the – it's between the green and – it's somewhere in the yellow range for me. It's not – that's not what such that means in English, by the way. Subject to are the English

words that say exactly what happens when you do something subject to restrictions. You don't do something such that restrictions. That's actually not English, but you will see this, but not from me you won't. And the worst of course is min this. Oh, and by the way the initials S period T period go either way, though it's really your – I don't know, so S period T – that's even worse, though. So find x subject to these constraints.

This just says there is no objective, and what that means is that any x that satisfies these constraints is equally attractive. There's just nothing – you're completely neutral. It just means find anything that works there. That's called a feasibility problem, and it's really the same as this, or it's convenient to make it the same as this. You make it minimize the constant objective function zero subject to, and then these inequality constraints. Now that looks stupid, but it's not and it encodes everything correctly. So what it means is this: the optimal value of this problem is either zero or plus infinity. If there is a feasible point, then – any feasible point here is optimal because if I have a feasible point, I could ask what is its objective value, and the answer would be zero, but that's as small as the objective value gets among feasible points. If there is one, that's p^* . Therefore, any feasible point is optimal here. On the other hand, if it's infeasible then the p^* is you take the infimum of zero over the empty set, and that's plus infinity so everything works out just fine when you do this. Yeah?

Student:[Inaudible] x offset, just the intersection of every domain and every constraint?

Instructor (Stephen Boyd):No. It's not the intersection of domains. The optimal set here coincides with the feasible set, so basically in this problem, in this formulation if you're feasible, you're optimal.

Student:But doesn't that include being in the domain of everything?

Instructor (Stephen Boyd):No.

Student:How can it be in a domain and not satisfy that?

Instructor (Stephen Boyd):Here, I'll make an example. Ready? Let's do one. I could say $\log(1 + x)$ is – doesn't really matter. I'll make it easy on myself there, something like that. Now I have to figure out what that means, but anyway – and then I'm gonna add the following constraint. Or I guess – let's see we haven't done that. We haven't turned things around yet, so I'm gonna write it like this. And then I'm gonna write another one which is – $x = -10$.

Now to say that you're in the domain here, there's no domain restriction from this one. There is a domain restriction here, and it's that $1 + x$ should be positive. But that's not the same as the feasible set. The feasible set – this says that x should be bigger than or equal to ten. That's the feasible set so that they're not the same. You have to go back and look at it carefully. Now I can say what a convex optimization problem is a mere three weeks into the class. Actually, I kinda said earlier. So here it is. There's a restriction, and the restriction is this: the objective must be convex function and the inequality constraints

must also be convex. The equality constraints, that is the h_i , must be affine. And that means I can write them this way. I can also just write it as $Ax = b$ in very compact notation.

Now if the objective is quasi-convex, then this is called a quasi-convex optimization problem, and you might just write it out – this is where I’ve just substituted for the equality constraints a single matrix vector equality constraint. Now the feasible set of a convex optimization problem is convex, but watch out because there’s another definition of convex optimization problem. I guess it’s what I call – there’s the strict definition, and there is the looser definition. The looser definition actually makes more sense, is more appealing, and so on, and in terms of actually doing convex optimization, it’s useless but let me say what it is. And it sounds right. If someone just bumps into you on the street and says, “What’s a convex problem?” you would say this: it’s minimizing a convex a function over a convex set. What could be more natural? And in fact, that’s called the abstract of the loose definition or something like that. And in fact, it’s okay but I wanna point out here that for us an attribute – so to say it’s a convex optimization problem, it is not an attribute of the feasible set. It’s an attribute of the problem description, period.

So there’s other weird ways to write $Ax = b$, and they will not work with these things. For example, I could write this. I could write $\|Ax - b\| = 0$. You have to admit if $\|Ax - b\| = 0$, $Ax = b$. So I haven’t changed anything here. That’s not a convex problem in the way we’ll describe it, even though it’s absolutely equivalent to one with $Ax = b$. Okay, so for us whether or not you’re a convex optimization problem is an attribute of the description of the problem and not for example the feasible set or something like that. And this will be clear in a couple of cases. Here’s a simple example. Let’s scan this problem for convexity and see what happens. Well, the objective – what do you think of the objective? Well, the objective is okay because it’s convex and you’re to minimize it. So the objective is cool. Let’s pass on to the first constraint.

Well, we’re already in trouble with the first constraint because this constraint here defines a convex set. The first constraint – it’s nothing but $x_1 = 0$ obviously, right? $1 + x_2^2$ is positive. I can get rid of it. However, this is f_1 . As I’ve described this problem, that’s f_1 and that is not a convex function, and so this problem is rejected. It is not convex. And the second one also is not convex because – that is indeed a convex function of x_1 and x_2 , but you’re not allowed to have convex function equals zero. You can have a convex function less than or equal to zero. So this – I would have to call that f_2 now – this actually would’ve been okay. That would’ve been fine. That’d be a convex problem because you have a convex function here less than or equal to zero. But the point is here is you take these and you rewrite it in an equivalent way. By the way, the problem – these are not identical problems. The problems are identical only if the objective functions and constraint functions are identical. Then the two problems are identical. However, they’re equivalent, and we’ll use a kind of an informal idea, but nevertheless completely clear idea of what equivalent means. Equivalent means that by solving one, you can construct a solution of the other and vice versa. And construct it means not using too much work. So it’s something like – I mean if you’ve had a course on complexity theory and computer science or something, it’s something like that except we’re not gonna be very formal

about it. It's something like a polynomial reduction of one problem to another. So that's what we mean by equivalent except that we won't get too fancy about equivalences are there. For us, it's gonna be an informal idea. Yeah?

Student: When you say they share a solution set, do you mean both in terms of the value of x and in off the value of x ?

Instructor (Stephen Boyd): Absolutely. In this case, I didn't change the objective, so that's why they –

Student: But you could change it?

Instructor (Stephen Boyd): I could change the objective. Right. So for us, equivalent problem means this – I think the best way to say it is that you could write a very thin wrapper code around one problem that would then solve the other problem. That's what it means. And in this case, it would be very thin. I mean what would happen is this: to solve this problem, you would solve that one, and then return the solution. So that's a pretty thin wrapper that goes around this. In other cases, we'll see you might do other things. Actually, we're gonna see lots of little equivalences, and I'll make it very clear what's happening. But that's the right way to think of equivalence as we'll use it. It means writing a wrapper around one to solve the other. That's what it's gonna mean. And a wrapper by the way that doesn't do like a huge amount of work. Okay. Let's say a little bit about local and global optima. So if you have a locally optimal point of a convex problem, it's globally optimal. This is by the way the kind of thing that people on the street would know. This is kind of the basic thing you'd know about convex optimization would be this, you know variations on it. And it's actually very impressive if you think carefully about it. It basically says if you have a problem, if you have an optimization problem – it says if you were to restrict your search – you have a point which is a candidate for being optimal, and it says if you restrict your search arbitrarily closely, locally – but if you do a full search in there and find that there's actually no better point locally, you can make the stunning conclusion from having examined this R ball which is tiny – in fact, it can be as small as you like – you can make the stunning conclusion that in fact even if you were to search over everywhere, there'd be nothing better.

Although, you know after a while you get used to it that the proof of these things is like three lines or something like that, so it's not a big deal. Still, what it's saying is actually stunning if you think carefully about it. To say that by – well, it's actually highly implausible when you think about it. It basically says if you do a competent search nearby and find nothing better, then in fact you don't need to go and look anywhere else. Actually, this is the best you can do. Of course, that's inherited from all the properties of convex functions and so on. So let's see how you would show this. It's actually totally straightforward. Suppose you have a point that's locally optimal but not globally optimal. What that means is there's a point – another point that's optimal – in fact, you don't even need the point to be optimal. So you just need a point that's feasible and has the objective value better than your locally optimal point. So suppose that happens. Then it's very simple what to do. You simply construct the line segment between your local optimum

and this better but feasible point. You construct the line segment, and you start moving towards – from where you are locally optimal to this point that's better. What happens is – of course, as you move on that line, you remain feasible because x is feasible, y is feasible, the feasible set is convex. Therefore, all along that line segment you will be feasible.

Then what can you say? Well, now you have a convex function that basically is locally optimal at first, but then later actually achieves a value lower. And of course, that's impossible. So that's the idea. It's very simple to show this. And I won't go through all of these details, but that's kind of the idea. What I just described is it. Okay. Let's look at an optimality criterion. Again, it looks very simple. It's very geometric, and it's gonna generalize what you've seen before, this idea of the gradient vanishing being the condition for optimality. So here it is. A point x is optimal – this is for a convex optimization problem – if and only if it's feasible, and the following is true: it says that the gradient of the objective evaluated at the point $(x)^T(y - x) = 0$ for all feasible y .

Now geometrically, it means this. Here's your feasible set here. These are the level curves of the objective. So this is clearly the optimal point. It's the point in the feasible set with the lowest objective value. So that's right here. You write the gradient of the – actually, you write the negative gradient of the objective function. That would be the normal to the hyperplane here, the local hyperplane of approximately constant objective like this and it says that this hyperplane here actually supports the feasible set at that point. It looks like that. That's the picture. I can actually sort of explain this. It just comes from this thing, right? It basically says the following. You know the following is true, right? $f(x) \leq f(y)$ if f is convex, this is true for all y . So if these are the level curves of your function like that, and I evaluate it say here, and the negative gradient points in that direction – it says basically if I draw this hyperplane like this, this inequality tells me something. It says that every point in this entire half-space here has a function value higher than the function value there. That's what it says in particular because if this is non-negative, then that's bigger than that. So that's the basic – the basic inequality comes from there, or the basic geometric conclusion is that when you evaluate the gradient of a convex function, and it points in this direction, it says that all the hyperplanes – I guess well the hyperplane – the half-space with that normal, all those points have worse value of f , and there's no point even looking at it because it's absolutely guaranteed.

So what this says is all feasible points live within this half-space here where the objective is worse than or equal to the value there. Well, guess what? That means that point is globally optimal. Okay? And by the way, to show the if and only if, it's very easy. I just showed one way. The other way's even easier. If this is true, then this basic inequality proves it for us. So it's quite straightforward. So this is it. By the way, this innocent little condition here – and notice that if the problem is unconstrained, let's see what this says. We'll look at these in list fashion. Let's take this simple – by the way, we're gonna come back and we're gonna look at optimality criteria in much more detail later, much more detail. But for now, this is just your first glimpse at optimality criteria for convex problems.

Student: What happens if the objective is not differentiated by that point?

Instructor (Stephen Boyd): Then this doesn't apply. Yeah. So the question is what happens if the objective is not differentiable by that point, and my response was then this doesn't apply. Okay. So let's look at an unconstrained problem. Well, let's see. What would mean to say if it's unconstrained, then this is just basically for all y in $\mathbb{R}^n$, so wait a minute. If I can choose y to be anything I like in $\mathbb{R}^n$, this vector here can be anything I like. What does it mean to say that I have a vector – let's call that g . Suppose I told you that $g^T z = 0$ for all z in $\mathbb{R}^n$. What's your conclusion? Yeah, $g = 0$. But how do you know that, by the way? What's a quick argument for that?

Student: Because if g is not zero [inaudible].

Instructor (Stephen Boyd): Okay. Yeah, sure. Here's a quick snappy one. Try that z , and you'll get your conclusion real fast because you get minus norm $g^2 = 0$. So g has to be zero. That's the only way. So this condition which looks kinda complicated – actually, the only thing you have to remember is this picture. So you remember this picture, then there's nothing more to say. So here, the condition is that the gradient is zero. Equality constraint, that – we can also get this, and this will recover these things like Lagrange multipliers which you've probably seen earlier and you will see again. So you wanna minimize $f_0(x)$ subject to $Ax = b$. Well, x is optimal if and only if – I mean the condition is this: you're feasible and the gradient plus $A^T \lambda = 0$. Now to show that – and I'll just give a quick plausibility argument, not give the whole argument for that, but I'll show that. It says basically the following: the gradient of $f_0(x)^T(z - x) = 0$ for all z that satisfy $Az = b$. Okay? Something like that. Well, I can – there's many ways I can write this. Actually, this is – what's even more interesting, we also know that $Ax = b$, and what we see is that it means $z - x$ is in the null space of A . That's what this says.

So this says that the gradient of $f_0(x)$ is orthogonal to everything in the null space of A , so we're gonna write that this way like so. Oh, you know what? I'm sorry. I skipped a step. Sorry. The gradient has non-negative inner product with everything in the null space of A . However, if you have a subspace – if you have a non-negative inner product with something in a subspace, then it also is non-negative when you – well, if you reverse the sign of $z - x$, which is still in the subspace, it's still in the subspace. Therefore, you have a non-negative inner product with a vector, and you have a non-negative inner product with its negative. That means you have to be zero. I skipped that. So that says this. $\text{grad } f_0(x)$ is in the null space of A perp or something like that. It means it's orthogonal to the null space of A . Okay? Now that is $R(A^T)$, so that says $\text{grad } f_0(x)$ is in $R(A^T)$. That means that $\text{grad } f_0(x) = A^T$ times some u , and I'll later make u – I'll mess with u to make it fit our form. But actually, I'm kinda done because I could write this as $\text{grad } f_0(x) -$ let's see how I wrote it here. Yeah. I can write plus $A^T(-u) = 0$ for some u . And I should – I guess I [cuts out] smart, I would've put the minus, but it would've been kind of weird, so anyway. There we go. That's the condition. By the way, don't worry about these. This is just your first glimpse at this. We will spend a week, week and a half on all these things later in much greater detail. Okay.

Now it starts getting interesting. It could be argued – I mean I presume that you would know this or have heard of it. I would presume you heard this: generally, Lagrange multipliers are taught as a behavior in an advanced calculus class somewhere where someone says how do you solve that and you're taught the following – then you should do this. And it doesn't make much sense, but you're told that's what has to happen. This would maybe be the first one where it's not obvious, and something maybe you haven't seen. So what are the conditions under which – when would a point x in the non-negative orthant minimize a convex function? That's the question. Well, let's just apply the – we'll end up deriving it, but it's gonna look something like this. It basically says $\text{grad } f(x)^T(z - x) = 0$ for all z positive. That's our basic condition. And let's simplify this now. So let's take a good look at this. This has to be positive for any non-negative z here, so let's see what happens. First of all, I can plug in a bunch of things. I can plug in $z = 0$, and I get the following that $\text{grad } f(x)^T x = 0$. Everybody agree with that? That's from $z = 0$.

And now I can do the following. I can let z – if an entry of this vector were negative, I'm in big trouble because if an entry were negative, I would take z – if the i th entry of this thing is negative, I take $z = t$ times e_i , and then I let t get super big. Then what happens here is this thing here – as t goes to infinity, this thing is an affine function of t with a negative slope, and there's no way it's gonna be bigger than or equal to zero. So we can also conclude that this vector is positive. There's a squiggles there. These are straight. These are squiggled. Okay? But now it's getting interesting because look at this. I have the inner product of two non-negative vectors and it's less than or equal to zero. Now when you take the inner product, you take the sum of the associated products, but obviously if you're doing that with two vectors that are non-negative and you end up with something that's less than or equal to zero, there's only one choice. It has to be zero, No. 1. And No. 2, it has to be complementary. In other words, we actually can say this. So basically, it says if x_i is positive, then that correspondent – that component of the gradient is zero and vice versa. If this is positive, that's zero, period. This is called complementarity, and you're gonna see it a lot, or you'll see it a bit. Okay? So that's the condition here as we just worked out this. And these are already kinda – these are not totally obvious. Okay. Next topic is equivalent convex problems, so as I said we're gonna use an informal notion of this. You say that they're equivalent if the solution of one is obtained with modest effort from the solution of another. What we'll do is just give a bunch of common transformations that would preserve equivalence, just examples to give you a rough idea.

Here's one. You could eliminate equality constraints. So I wanna minimize $f_0(x)$ subject to $f_i(x) = 0$ and $Ax = b$. I can eliminate these, and the way I do that is I find a matrix F and a point x_0 so that $Ax = b$ is equivalent to having the form $Fz + x_0$ for some z . Now how do you get F ? I mean these are quite easy. F is like the null space. The columns of A would be a basis for the null – not a basis, sorry. They would span. That's all we have to do. They would span the null space of A . And then x_0 is any particular solution of $Ax = b$. For example, it could be the least norms solution for example. So this is linear algebra to go from $Ax = b$ to F and x_0 . But the point is this is what people would call a constrained representation of an affine set. And this is what people would call a free parameter

representation. So this is parameterized by z which is now free. Okay? If it's a minimal one, you could say something about the size of these things, but that's just linear algebra.

So what I'll do now is I'll make a new problem which is this. I'll take $x = Fz + x_0$ and I get this problem here. I don't need this constraint because basically $A(Fz + x_0) = b$ for all z automatically because AF is zero. That's how you constructed F , and Ax_0 is b . That's how you constructed x_0 . So you don't need this constraint. You drop the constraint, and you get this. Now these are not the same problem. I mean they're totally different. They have dimension in variables. This one has equality constraints. This has none. But they're clearly equivalent, and the idea would go something like this. If you wanted to solve this problem, you would solve this problem and then return as x , $Fz + x_0$. That's what you'd do. Okay? Now the idea that it has to be a modest transformation to do this has something to do with the following fact that when you get A and b , it has to require modest computation to calculate F and x_0 , and indeed it does. That's just basic numerical linear algebra.

Student: [Inaudible] F again?

Instructor (Stephen Boyd): Yes? Oh, F – the columns of F are for example a basis for the null space of A . That's good enough. There's many ways to construct such an F like that. Okay. And by the way, if this problem is convex, so is this one. And the reason is that the objective is actually a – you pre-compose with an affine function, but if you pre-composed with an affine function, that preserves convexity. Okay? This one is unconstrained, no equality constraints. Now you can do the opposite, and in fact the shock or hopefully within three or four weeks you will – I mean later in the class there'll be like – I don't know. When you – somewhere near the ninth week of the class, you'll think what were some of the shocking things – that a lot of it kinda sorta made sense intuitively, but some of it will be – there'll be things where you'll say that wasn't obvious. Now I'm gonna tell you one. This is just a topic, but we'll get to it. It's this. It's actually often extremely useful to introduce equality constraints, so to “uneliminate.” Everyone has a bias towards elimination because people like to solve problems with smaller numbers of variables. I will cure you of that before this quarter is over. So if you know what you're doing, not only is it sometimes not a good idea to eliminate constraints, it's often a good idea to introduce new variables and constraints, which sounds very odd when you first see it, but trust me before the quarter's over, it'll be very natural to you. Yeah.

Student: Yeah, why do the equality constraints have to be affine? If we're just gonna compose it and need to preserve convexity, couldn't we just have any convex –

Instructor (Stephen Boyd): Well, my answer to that which is – the question was why do the equality constraints have to be affine, so my answer to that is gonna be very unsatisfying to you. It's this: that's our definition of a convex problem. By definition, a convex problem – the only equality constraints we have are affine. Sorry. Well, generally speaking, other affine constraints don't lead – then the constraint set generally is not convex. And then you don't even meet the broader definition of a convex problem, which

is minimizing a convex function over a convex set. So sorry, it's by fiat. Convex problem has affine constraints. I told you you wouldn't like the answer. Yeah. Oh well. Sometimes there's good answers. Sometimes there's unsatisfying ones. That was the latter. Okay. So how do you add – I mean you can actually introduce equality constraints, which by the way some people call unelimination. I kinda like that. So it works like this. You have a problem that looks like this. You could do the following. I'll introduce a new variable y_0 here. I'll introduce a new variable y_i here. And I'll have this problem. Minimize over x and now also y_i , so I've added new variables – $f_0(y_0)$ subject to $f_i(y_i) = 0$, and then these equality constraints.

Now you could reasonably ask – it's entirely reasonable to ask the question why on earth would you add extra ridiculous variables because this doesn't look like progress. The problem started with no equality constraints. After this unelimination step, you have added variables and added equality constraints. This does not look like progress. Later you'll see shockingly this is the first step to lots of progress. I know it's weird, but this will all be clear later. Another trick is to introduce slack variables for linear inequalities. Now that works like this. You're gonna say I have $A_i^T x = b_i$. That's fine. What I'll do is I'll convert that to an equality constraint. I'll convert that to $A_i^T x + s_i = b_i$. I mean it's stupid. This is the same as saying s_i is the slack. It's $b_i - A_i^T x$, and then I'll write that the slacks are bigger than or equal to zero. Or to put it in our absolutely canonical form, I would write $-s_i = 0$. Okay? Here – to put it this way. Again, you know these are simple. Actually, what's remarkable is that we're gonna look at a bunch of transformations. Most of them are really stupid like these, and it's weird to think – yeah, it's like everything else. You learn about 15 trivial things, and then you put them together in just the right sequence, and all of the sudden something that was not obvious at all will come out. So yeah?

Student: When you minimize that problem, don't you know for a fact that the minimal s_i will be zero, or s will be zero since [inaudible] push up against the bound?

Instructor (Stephen Boyd): No. Also, that's not true. That's just not true.

Student: So s_i can be actually finite?

Instructor (Stephen Boyd): Yeah. The slack can be positive. Sure. It depends. It's basically you solve that problem – I mean if you wanna – by the way, each of these needs an argument that they actually are equivalent. So don't just trust me. After you do a few of these yourselves, work these out, you'll see. Also, you'll probably do some of these numerically, and basically if you don't know what you're doing, you'll just come out with the wrong answer, so that'll be another feedback mechanism for you. But let's just check here that they're equivalent. Well, if I solve this problem, then I can define $s_i = b_i - A_i^T x$. Now if you solve this problem, it's feasible, so the s_i s are positive, and those s_i s are feasible for this, and in fact they're optimal, but I'd have to argue that, which I'm not gonna do. They didn't go backwards. And you can say suppose someone solved this. We're gonna make a – the wrapper that would allow you to solve this problem if you had code to solve this, or if you had the solution for this or whatever. So if I solve this

problem, I drop x and I return – the solution of this one gives me x and s , and I simply drop x and forward s , and argue that it's actually – here, it's optimal here. So that would be the argument. I'm actually not gonna do it because these are actually shockingly boring, but should be done once or twice or something or more, but not many more times than that, and you will. I promise.

Student:[Inaudible] x and s – is there another objective? Or what is –

Instructor (Stephen Boyd):No. It just means I have more variables. So it means that when you minimize over x here, it means that x is what you can mess with. Here when you minimize over x and s , it says you have x and s to mess with, so that's what it means. Here's a simple trick. It's called the epigraph trick. It's actually quite useful, and it's widely used in practice. Let's see how that works. I mean it's really dumb. Here's what it is. You wanna minimize f_0 . You introduce a new scalar variable t , and you introduce this new constraint, which is $f_0 - t$ – this is basically $f_0(x) = t$, and you minimize t . Now this is kinda dumb because if you solve this problem, first of all you'd be an idiot to choose any t other than $f_0(x) = t$. Not an idiot, you'd just be – that t couldn't possibly be optimal is the point because your goal is to minimize t . However, I cannot adjoin this to this problem. It is an equivalent problem to write equals here. That's absolutely equivalent to the original problem. Unfortunately, it's not convex unless f_0 is affine. However, with a less than or equal to, it's convex.

It's a really dumb thing. Let me just draw this. Here's a function that you wanna minimize like this. The original problem basically says find the point here that minimizes f_0 . This one does this ridiculous thing and introduces a new t that's up here like that, and it says that's your feasible set or something like that, and then it says go as far as you can in the direction $-t$, right? Like that, actually I guess the direction is -1 , and you would recover this point here. By the way, this has immediate practical application. This says for example that if you're gonna write a code to solve convex problems, you don't need to – without any loss of generality, you can just minimize a linear objective. That's quite standard. So you look on – if you go to Google and type C source and then some name of some convex problem, you will find it'll just – many times, they won't even say it. It'll just say it solves the following problem, and it's a linear objective. You go hey, I can't use this. My objective is not linear. Well, it's just expected that you know this, so this is just expected. By the way, it's generally not in the documentation which is actually interesting because that's how widely this is supposed to be known. It's also how unhelpful the documentation is. So this is the epigraph trick, and you'll see this used quite frequently. Oh, the other thing you'll hear is this. You'll actually have people say just without loss of generality, the objective can be assumed to be linear because that's a linear objective. It's a linear function of x and t . So that's just completely standard.

You'll also hear people say things like this: they would say linear objective is universal in convex optimization, so whatever that means. It means something like if you can do linear objective, you can do anything. A couple of other equivalences – here's one. You can minimize over some variables, so if you have a problem that says minimize $f_0(x_1, x_2)$ subject to $f_i(x_1) = 0$ for example, I could minimize this thing over x_2 , and that would

reduce – that would give you this function, f_0 tilde of x_1 , and that would give me this. By the way, for those of you who have seen dynamic programming for example, that would just be this – a fancy version of this would just give you all of dynamic programming, and basically then there's nothing to say about it except it's just this. Oh, except interesting point – dynamic programming preserves convexity of a problem. You have a problem of a few variables, or in a general case a whole bunch of variables – for example, stages in an optimal control problem or something like that. If you minimize over some of the variables, you have to take account of that in the constraints, the objective, everything, but when you minimize over one family of variables in any convex function, the result is a convex function.

So we can now say something like this. Dynamic programming, if you know what I'm talking about, preserves convexity, so when you optimize over the last stage variable, and end up with an – if the original problem were convex over capital T stages, the $T-1$ stage problem also convex, and the $T-2$ and so on. Okay. Just out of curious, actually how many people have heard about dynamic programming and stuff like that? Okay, so not an empty set. Okay. Quasi-convex optimization – well, here f_0 is quasi-convex. There are lots of actually practical problems that are gonna be – we'll see lots that are quasi-convex or something like that. And here you have to be very careful because for example this most basic thing which says a locally optimal is globally optimal is completely false. And here's a perfect example. I'm making it exactly flat here. So that's a locally optimal point, no doubt about it. And it is obviously not the global optimum here. And that's a perfectly valid quasi-convex function.

So a lot of things are different now and need not be right. But here's how you solve them with total effectiveness. It works like this. Generally speaking, if you have a quasi-convex function, I mean unless you're just doing a theoretical analysis, if you're actually gonna actually do anything – that is to say construct an algorithm that solves a problem, solve an actual problem, or anything like that other than just talk about it – then what's gonna happen is this: you're gonna actually want – you want a convex representation of the t -sublevel set. So it's not enough to say – so you construct a – now, a convex representation looks like this. It says that you look at this sublevel set, and you must write it as – I've written it as one convex inequality parameterized by t , but you don't have to. It could actually mushroom into a bunch of convex inequalities. It just doesn't matter. You need a convex representation of the sublevel set is what you need here. Here's an example. It's a general fact that if you have a ratio of a convex to a positive concave – actually, non-negative convex over a positive concave function, this function here is actually quasi-convex. It's very easy to show that because – well, I mean there's some horrible details, but let's do this. You wanna know is that a convex function of x .

By the way, if this were a convex function of x and t , that would be the proof that this function is actually convex in x . It's not convex. It's quasi-convex. That means t consider fixed, and you wanna know is the set of x that satisfies this convex. That's the question. Okay? So if you look at this, well you just multiply through, and you get $p(x) = tq(x)$. Now you're in pretty good shape because I write that as $p(x) - tq(x) = 0$. That's convex. That's concave. t is bigger than or equal to zero without loss of generality because if t is

negative, the sublevel set is empty and convex, so I've already dealt with that case. Then this is convex minus non-negative multiple of concave. This whole thing is convex. That's a convex representation. Okay? It depends on t affinely, but that doesn't matter. In fact, the dependence on t is completely irrelevant. Okay? Now the way you solve a quasi-convex problem is this. You just solve it – if you think about what you can do now, if you have this convex representation for any T , it means you can do the following. You can actually ask a question like this. You could say is the optimal value less than or equal to t ? And you have to solve a convex feasibility problem to determine it.

So you could say is the optimal value less than three, and the answer could be no. Then you'd start with a value of t where the answer's yes, like t equals 10. Then of course you'd do bisection, so you'd take the average of these. You'd take 13, and then the next query would be 6.5 and so on and so forth. So that would be basic bisection, and that looks like this. So you start with a lower and an upper bound on p^* . By the way, you might not have that, in which case sort of a practical version of this would start with $t = 1$ and query. And if that is infeasible, you would then try $t = 2$, then four, then eight, then 16 or something like that. And then when you get to a big enough number that you wouldn't be interested even if it were feasible, you would quit. So that would be one way. If it's feasible and you don't have a lower bound, you'd start with $t = 1$, then you'd try a half. If that's feasible, you'd try a half, a quarter, and when you get down to something very small, you'd quit. Okay. Well, sorry that's all on the assumption that it's a positive function that you're minimizing. So you start with a known upper and lower bound on p^* , and you simply query at the midpoint of your known upper and lower bounds, and you solve the convex feasibility problem. So this is just bisection. And every time you do this, your interval – you end up with two points, an l and a u . l is you actually have an x where the – sorry, you don't have an x . For u , you have an x that actually beats – that has at most the value $f_0(x) = u$. And l on the other hand is a value known to be infeasible. There's no x that satisfies $f_0(x) = l$. Anyway, and then at each step that interval of ignorance divides by two, so your ignorance halves at each iteration. So it takes log ten steps, and you reduce your ignorance by 1000, and 20 steps a million roughly. That's the idea. And yes, you'll definitely do this at some point.

Student:[Inaudible] wrote up there has objective zero implicitly?

Instructor (Stephen Boyd):Right.

Student:Okay. And what does the colon equal mean?

Instructor (Stephen Boyd):Where?

Student:In the bisection method?

Instructor (Stephen Boyd):Here?

Student:Yeah.

Instructor (Stephen Boyd): That's an assignment.

Student: Okay.

Instructor (Stephen Boyd): Okay. Now we're gonna start in on – we're gonna look at some problem families. This is actually quite interesting, quite useful. The problem families are interesting mostly because you can actually – all the things we're gonna look at actually are extremely practical. I mean what we've done so far is all very general and follows – actually would follow a traditional course on this material, but it would stay at that level. We start talking – the things we're gonna talk about now are extremely interesting theoretically. In fact, some of them are at the height of fashion in various fields at the moment, not linear programming but other things we'll see. All right? The absolute height of fashion even among people doing strictly theory, and it's very interesting. But actually, what's mostly interesting about it is that all of these things are extremely doable, so you can do this. In fact, you will shortly. So that's sort of the – we're now moving into things that this is more than stuff you talk about. These are things you can actually do. So I just thought I'd point that out. So linear programming – a linear program is almost sort of the simplest optimization problem there is in one sense. All of the functions involved are affine. Everything is affine. The objective is affine. The inequality constraints are affine. By the way, and people call them linear so that's okay. And the constraints are affine, again called linear constraints. It depends on whether when you say affine or not – linear, you allow an arbitrary right hand side. If you allow an arbitrary right hand side, then of course when someone says linear it means affine. Okay.

So that's an LP. By the way, you know this is not a new topic, just to make that clear, not remotely a new topic. So Fourier wrote a paper about this which is a hint that this is not exactly a new topic. In fact, he wrote a paper on how to solve linear programs. It was a method that would be very effective if there were like four variables, but hey that was the beginning of the 19th century. Anyway, so modern history actually has a lot to do with Stanford, but to be fully honest it has a lot to do with Moscow State University and later Stanford, but it has a lot to do Stanford because this was sort of popularized in around 1948 by a bunch of people who were here, like George Dantzig. So the feasible set of course is a polyhedron, so your image of what an LP is should be this. It's a polyhedron, and then the objective is affine, so I mean the level sets of an affine function are hyperplanes, so you're just – here's negative c , and you're told go as far as you can in this direction. And this is a little example makes it totally clear that's the solution. Okay? Now, before you imagine that this picture really fully does the trick of understanding it, I should point a few things out. If you have a linear program with I don't know 30 variables and like 100 inequalities, the feasible set oh it's a polyhedron. It's a polyhedron. Wanna know how many vertices it has? Someone wanna take a guess?

I forgot what numbers I said, so what I'm about to say is a lie. It would become true by increasing those numbers modestly. The answer's just completely astronomical. It's just absolutely astronomical. It goes up very quickly. So you'd write down what's called a small linear program, and the number of vertices it will have on this polyhedron will

simply be literally astronomical. It doesn't take much. You can easily – you will solve in this class many linear programs where the number of vertices in this – in the feasible set will exceed probably by a long shot the number of subatomic particles in the universe. You will solve many of them, quickly I might add. So I just point this out because this is the picture you should have. This is what I think of when I think of an LP, but be forewarned that this doesn't really give you a full – when you look at this – you know if you were to go -- if your roommate asks, "What are you doing?" and you said, "It's really – yeah, we learned today how to like go as far as you can in this direction on this set," they'd look at that and they'd go, "When's the drop deadline?" So just watch out. The pictures explain it, but I don't think you get the full idea of it. So let's look at some examples. Here's a famous one. This is just here for historical purposes, so it's the diet problem. So you're supposed to choose a bunch of foods, so you have n possible food types, and you're gonna choose some quantities that you're gonna feed some group of people. I don't know, presumably soldiers. I don't think anyone would eat this willingly, what's gonna happen here, but you're gonna feed some people some amounts of food.

One unit of food costs – you have these costs which are c_j , and each one contains a_{ij} of some nutrient. That's like whatever 10 grams of the food or whatever our unit is here. And a healthy diet requires at least b_i units of some nutrient. So how do you find the cheapest healthy diet? Well, you minimize $c^T x$. Now $c^T x$ here is literally the cost because x_i is the quantity of each type of food you're gonna put into this mixture. And then when you take the inner product of c , that gives you the actual cost. So that's minimizing the cost subject to $Ax = b$. A is nothing but this matrix here. It's the nutrient matrix or whatever, and then b is this minimum daily allowed whatever, something like that. Okay? So this is a famous problem. x has to be positive. You can't feed people negative amounts or whatever. So that's the – now, whether you'd wanna eat this diet is another story, and of course it doesn't take much here to realize it's a fairly poor model. It assumes these things sort of add linearly which might not be the case. You might impose – could I impose an upper bound on the amount of a nutrient or something like that here? Of course, I could. That would still be a linear program and so on. But this is just for historical reasons. It's sort of the first one you would hear about. Okay. Here's one: piecewise-linear minimization. So you wanna minimize a piecewise – the maximum of a bunch of affine functions. So you just wanna minimize that over x . Okay? Now, that's an interesting problem because it's not remotely linear, not even close. In fact, when it's linear it's boring because I know how to minimize a single affine function. The answer's minus infinity.

So this is not linear, not remotely. There's nothing linear about that problem. It's also quite useful because actually piecewise-linear functions can be used to approximate actually any convex function. So although this looks kinda silly, it's way more powerful than you think. So just minimize some piecewise-linear function. It's an LP. You just put in epigraph form. So you just write it this way. I'm gonna minimize a new variable t subject to $a_i^T x + b_i = t$. Done. Now it's linear – now the variables are x and t . These are affine linear inequalities, or to use the term that most people would use, these are linear inequalities in x and t . That's it. By the way, the reason you need to know this – there's some reasons you need to know it because if you went to Google and typed source code

PWL minimization, you'd find some – actually, what you'd find is terrible stuff from total backwater fields where they don't even know it's a linear program. That's what you'd find. And it'd be the worst stuff you've ever – if you found anything.

On the other hand, you won't find anywhere anything that would tell you if you didn't know that piecewise-linear – so I think that's the problem. Everyone knows piecewise-linear minimization is equivalent to an LP. So what you really want to do is get LP solvers of which there is tons. In fact, sadly if you went to Google and typed that, you'd probably go to this PDF file, but the others would be very bad, I assure you. We'll look at another one, and it's the Chebyshev center of a polyhedron. Now we already talked about this a little bit. We talked a little bit about the idea of yield maximization. So yield maximization was this problem. And let me just – this is just to sort of set this up, so you'd know why you would wanna do for example Chebyshev centering or something like that. Yield maximization was that we had some polyhedron of acceptable values of some parameters. We're gonna set a target for manufacturing. So we're gonna set our machines to attempt to manufacture these points. In manufacture however, you will add a random perturbation to this point, and so what you'll get is you'll get a PDF that is centered – has mean value at this point, but otherwise tails off. And then we're interested in maximizing the probability that you're in this set. That would be the yield. Okay?

That's actually generally a hard problem, although convex if the PDF is log concave. But there's actually a simpler method, I mean it's to simply say look, what you really want is you want that point to be well-centered. You want it deep in this set. For example, if the PDF were n zero i, so if the perturbations were Gaussian, uncorrelated, then basically it says that if you look at the PDF, the level sets would be circles. And so basically what you might argue then is you want this thing to be – to find the point here where the largest circle around it fits in this set. It's a heuristic, but it's something that would actually work very well, just something like that for a nice rounded perturbation. So that leads you to this idea of the Chebyshev center of a polyhedron. And what is that? Well, I'd have a bunch of inequalities. I have my polyhedron. And what I wanna do is I wanna find the center of the largest ball that fits inside. So another way to say it is that's the point that is deepest inside the set. I can define the depth of a point in a set as the minimum distance to a complement of the set. It's basically how far you are from the outside. That's the depth of a point. And what I would want to do is I wanna find the maximum depth point. I wanna find the point deepest in this set. Now by the way, you know you're drawing pictures with spheres and balls, the whole thing says find me a Euclidian ball, but that means that – that's not gonna trigger the linear portion of your brain. It shouldn't anyway, right?

So if you talk about linear things, flat is what – the portions that light up when you think about flat things should be flashing. But when I start talking about things like ellipsoids, and balls, and round things, that should just suppress all of the brain activity in the lobe that handles affine things. It turns out that's wrong, so watch this. So I'm just saying that when someone tells you please find the center of the – when you see like ball, you should think round. Round should suppress the linear, and you would not at that point imagine it's gonna turn into an LP. Well, it is and the reason it's quite straightforward is this. aiTx

$= b_i$ for all x in this ball centered at x_c . That's easy to do. I simply maximize this thing over the ball and see whether that is less than b_i . But this is nothing. It's $a_i^T x_c$. That's the center. Plus, the sup of $a_i^T u$ when u is constrained to be less than r , and it's equal to this. Okay? Now so this – the inequality that you should stay on one side of the half-space is in fact here, and shockingly it's linear in x_c and r . So you can do Chebyshev centering this way. You maximize r subject to $a_i^T x_c + r \|a_i\|_2 = b_i$. Now watch out because the norm should also be triggering your nonlinearity detectors. However, the norm is applied to data and it has no effect. The variables here are x_c and r , and these are linear inequalities in x_c and r . Question?

Student: I don't see why this thing has to be unique, especially with like a regular [inaudible].

Instructor (Stephen Boyd): Why what has to be unique?

Student: The Chebyshev center.

Instructor (Stephen Boyd): Right. It doesn't have to be. You're right. It does not have to be unique, so that's a very good point. However, what this does is if it's not unique, this LP will construct an optimal solution. And by the way, I should mention essentially that's the semantics of an optimization problem.

So if I put an optimization problem down on the table in front of you, all that can be asked of you is to produce a solution. So if someone then comes back later and says, "Hey, wait a minute. I've got two different solutions. That's not cool," it's only not cool if one of them is either infeasible or if one has an objective better than the other, in which case you have to get rid of the person who produced the alleged – who alleged that their non-optimal point was optimal. But in general – By the way, if it really matters to you that it should be unique, then there's actually ways where you can add a secondary objective or something like that to this. But that's a very good point, and in fact for a Chebyshev center, it doesn't have to be. Just take a rectangle, and it's completely clear. Take a rectangle. You can fit a ball in it, but that ball can move. But any of those points technically solves the Chebyshev centering problem. Now for yield maximization, you would further want that ball to be in the center, right? But just to give you – but that's a very good point. Yes?

Student: Kinda like the [inaudible] LP?

Instructor (Stephen Boyd): No, you could – this would be easily generalizable to other norms, very simple. All that happens here is if that's a norm, that's the dual norm. So no – here, I'll just do it right now. There. Done. Now it's general norm. Now it's – you can't call it the Chebyshev. It's the general norm maximum depth point ball in polyhedron problem or something like that anyway. You could probably bet some good acronyms out of that, but we won't do it because we're out of time.

I think the only thing I will say about this is this is a really simple example, but it's actually very cool. Let me tell you why it's cool because these are example – like the piecewise-linear minimization and Chebyshev centering, these are sort of – It should be giving you a hint that a stupid problem family like linear programming that's been around since Fourier, was fully understood by Gauss – actually, I don't know that but it would appear to be a safe statement since everything I think was fully understood by Gauss but that what's cool about it is this kind of very innocent looking boring thing that if you go to the OR department, you'll find out about diets and some other stuff – What's cool is it actually has lots of – and a lot of them are not – they're not obvious things, right? I mean it's not obvious that finding the biggest ball that fits inside a polyhedron is a linear program for the reasons I mentioned before. So that's – we'll quit here. I should remind you that at 11:00 today, we have the section that should've been, would have been yesterday.

[End of Audio]

Duration: 80 minutes