ConvexOptimizationII-Lecture02

Instructor (Stephen Boyd):Hey, I think this means we're on. Are we on? We're on, okay. So well, the first announcement I guess is kind of obvious, but I guess if you're here. We've moved. It was a challenge actually getting the word out, because the registrar hasn't yet populated the EE364b all list.

No, sorry, it has populated it. It has just me and the TAs on it, so. In fact, let me actually start by introducing the TAs. You should know them. Jacob and [inaudible]. So familiar faces, maybe, from 364a.

Okay, so the obvious first announcement, kind of obvious, is yes, we're now on SCPD. Now that's too bad in a way, because I'd just gotten used to, well, the idea of teaching in the basement of the history corner and saying basically whatever I want and not worrying about it.

So now, unfortunately, since I guess these'll end up – these will be posted, you know, just on the web and open to anybody, I'll have to behave a little bit better. But we'll just try to pretend – I don't know, we'll just try to pretend these aren't going to be posted, and maybe if I say anything too terrible or whatever, we'll go in and fix it before it's posted, or something like that.

So okay. And by the way, please do not – or no, please? Let's see – no, that's too polite a word. How about this: do not take that as license to sleep in. We don't sleep in, so we don't see why you should. So our requirement of attendance, we still have – that's still enforced, even though we are now going to be televised.

Okay, let's see – and a couple other announcements. I presume people will be coming, wandering in in the next ten minutes; that's perfectly okay because if you're not on this – if you weren't actually registered or on the access list or something like that, there'd be – you'd be scratching your head reading this sign over at the other room about right now, and I'd say about ten minutes later we'll have a bunch of people coming in.

So I'll just cover some basic stuff over the next ten minutes. The first is – let's see, we posted homework one, but right before class I thought of another problem, so we have to write the solution for it to make sure it actually – it's a doable problem. And if it turns out it's actually doable, then we'll repost it and there'll be three homework – I mean, these are small things, right? These are just so you don't just watch me do subgradients, you actually get to do some subgradients yourself.

So you'll hear more about that. Let's see, the other thing is we posted a resources page. It's very crude right now on the course website, and that's something you should look at for sure. So you should be thinking about projects. I mean, you don't have to be spending all waking moments thinking about it, but you should think about it a little bit. I've already spoken to some people about it; that's good.

The resources page basically says kind of what we – you know, the first order, here are the first order of places you should look. So, you know, and it lists the obvious things, but just to be complete. I mean, definitely anything in 364a, the convex optimization book, and we've listed some other places – Google.

I mean, these are – don't come and say "I'm thinking of doing X." First – I mean, just because of the size of the class and we're doing projects, just to make it manageable, just do a little – I mean, please go and Google everything first and get a couple of papers, even though they're going to be terrible. At least then you know there's some papers on something before you come and talk to us.

We'll set up sections next week. They're not going to be sections in the normal sense, so they're not going to be, like, you know, here's a hint on homework one or something like that, because that's not what this class is supposed to be about. What they are gonna be is actually maybe just kind of – like the sections I think I'll do them for a while, or we'll all do them, and they'll just be discussions of potential projects.

The idea of a project and so on, especially early on, because – and that's something that everyone should probably come to, and we'll just – people will suggest projects and we'll just discuss sort of and aim to push the projects into a form that makes sense for the class and so on, and that would be something. And if it brings up some cool topic, that might be useful for everybody.

Oh, there is one thing about the projects. You are going to have to – you may have to read ahead. So, you know, obviously there's a small causality issue. The causality issue is this: it may be that your project will depend critically on material that we'll cover in this class the seventh week, or eighth. Or maybe we'll never get to it at all, or who knows? Or something will happen. Because it's going to be – the whole idea [inaudible] is going to be a little bit less organized and so on, be more fun.

That means, you know, you may have to go ahead and read ahead, and in fact that's why I guess someone was asking about the notes being – so we have posted a bunch of notes. Yes, those are the notes from a year and a half ago, which was the first time this class was taught. I think they're okay. You know, we reserve the right to update them, improve them, and stuff like that.

But they're there precisely so you can read ahead. So for example, if your interest is going to be something involving global optimization or if it's going to heavily involve distributed optimization, if you want to make some distributed thing or something like that, or ultra-large scale, you're going to have to read ahead. So that's kind of obvious, but I thought I'd just mention it.

Okay, any questions? I guess we probably have a few minutes before our second wave of people arrive. No? Okay.

Then I guess we'll just start in. It's going to be interesting – well, okay. I guess I should have a laser pointer. I'll bring one next time, or something like that, so I can point to various things.

Okay, so what we're gonna start today from the idea – so last lecture we looked at the idea of a subgradient, which is essentially, it's a generalization of the idea of a gradient, but for a non-differentiable problem. Actually, there is another generalization of a derivative to a non-differentiable problem. We're going to see that other definition soon, and in fact they're going to be duals, in a certain sense, so we're gonna see how that works.

So subgradient is one. It turns out it's actually the useful one. The useful one is the subgradient one – useful in terms of many things, like for example if you want to come up with simple algorithms to optimize things, subgradients are very, very useful.

Okay, so last time we got as far as this, we said that – we observed that if you have a subgradient of a convex function at a point, that's G is a subgradient of F at X. Now of course there can be many subgradients, but if you have any subgradient, what it tells you is this: it gives you essentially – I mean, here's a point – I can actually reach this with a laser pointer – so here's a point where there is only one subgradient, because it's differentiable here, and it's the gradient, and it points this way.

And what this tells you is that this entire half-space here, this whole half-space here has a function value of higher than the value at this point X1. And that means, for example, if you wanted to minimize that function F, there would be absolutely to consider this half-space.

So another way to say it is a subgradient rules out an entire half-space from consideration. So that's what a subgradient is. And by the way, that's true if the function is convex or not. It just happens only to be useful in the case when the function is convex, because it's actually true – I mean, this is true even if the function is not convex.

The problem is you'll never know. You'll never find – there's no way to find the subgradient and so on and so forth. There's no way even to know that something is a subgradient unless the function is convex.

Okay, so let's look at a useful generalization of this occurs when you have a function that is quasi-convex. So remember what quasi-convex means, it means that you have – quasi-convex means that the sublevel sets are convex, and you would typically, if you had an objective like this, you would solve it by some kind of bisection, for example.

You would ask for – if you can attain a level, that's a convex feasibility problem, and then you might do bisection, for example.

Okay, so quasi-gradient is a vector that satisfies the following. It says that if X transpose Y minus X is bigger than or equal to zero, then F of Y is bigger than F of X, so that's a

quasi-gradient. I should warn you that if you do look in the literature, you will find tons of variations on these. You'll have, like, a strong quasi-gradient, a pseudo-gradient, a this and a that and that, so just watch out if you look in the literature or anything like that.

And so the basic idea is this: it says that if you have a quasi-gradient, something like here, it basically says that everything on one – it rules out that half-space. It says that everything in that half-space has a higher than or equal function value from the other one. So that's quasi-gradient. So obviously if the function's convex, a subgradient is a quasi-gradient, because that's where the whole idea comes from.

So let's look at some examples. The canonical example of a quasi-gradient function is linear fractional. So we look at a linear fractional function on the domain – the domain is going to be the open half-space where the denominator is positive, and so here if you work out – I mean, you can just easily work out what a quasi-gradient is; it's just a quick calculation.

But for example here you calculate one this way. It's just A minus F of X, zero, C. I mean, this is easy, you simply write F of X is less than F of X0, or F of X is bigger than X0, and actually that's a half-space anyway. So that's a quasi-gradient, and to check, all you have to do is check, is this, is if you say you have A transpose X minus X0, if I multiply this thing out here, that's bigger than F of X0 times C transpose X minus X0, and that tells you if you simply divide here and add D to both sides here, the X and the X0, that says F of X is bigger than F of X0, okay? So that's an example.

As another example, here's a quasi-convex function, and it is the degree of a polynomial as a function of the coefficients, right? So it's just interesting because integer value than – so that's what it – and it's kind of pedantic, even to call the degree quasi-convex, although it's not a bad thing to do.

Because the way you solve it is really silly. If you had some convex problem involving a polynomial like this, you would – and you wanted to say, for example, minimize the degree of the polynomial that satisfies something – for example, find me the lowest-order polynomial that actually satisfies some interpolation conditions, or approximate something well enough.

It's quite simple. For each fixed degree, it would be a convex feasibility problem to check if such an approximation exists. And then you could just do it bisection. You take N, you take zero, and then you take N over two and round up or down – I mean, your choice, it won't make any difference – and run bisection.

Okay, so here, what you need is a quasi-gradient, and here you could just work out what it is. It turns out you can just take the sign of the K plus first coefficient in the K plus first position, and you can actually just check, that's quite straightforward, because you get G transpose B minus A is, well, G transpose just picks out the K component and multiplies it by AK plus one, the sign of AK plus one. So this will do the trick, and of course this will imply that BK plus one is not zero, and that will imply, if BK plus one is not zero,

that implies that the degree of B is at least A plus one, which is bigger than K or whatever, so that's how that works. So these are examples, and these are not hard. I mean, you just – these are, like, little exercises.

Okay, now we'll look at optimality conditions. So – yeah, by the way, I want to point out we sort of – it is important to understand. So far we've done nothing but analysis, we've had – so it's been just, it's math. But it's even worse, it's kind of circular math where you just introduce a bunch of definitions and then do things but never actually say anything that helps you or whatever. So that's a current state.

That'll change, actually, later this lecture, but for now that's the current status. Actually, the first inklings of something – that you've actually done something will occur now, and they just sneak up on you very quietly, so they're really quite weird. So here it is.

You have a differentiable function, the condition is for optimality – necessary and sufficient is that the gradient should vanish, that's it. And, you know, one way it's very easy to show. If the gradient vanishes, you write down this inequality that says – let's see if this is gonna work.

If a function is – who's going to be able to [inaudible] through the sight line very carefully here? Let's try this. So we have this, and the question would be, can anyone see that? Okay, and that's good enough. So here's your basic inequality. Obviously if the gradient is zero, it basically says that for all Y, F of Y is bigger than F of X, but that's basically the definition of X being the global minimizer of F, right? So that's simple enough.

And then the other direction is also easy to show as well. In fact, the other direction you can show lots of ways. One would be to say suppose the gradient isn't zero? If you took a small step in the direction of the negative gradient, the function would go down, okay, which would kind of cast some serious doubt on your assertion that X was optimal.

I mean, what I just said was actually a proof, but just, you know, in words. Okay, all right.

So non-differentiable, I mean, it's really simple. It looks totally innocent, and it's this: a non-differentiable function is minimized by X, star if and only if – now here the sub-differential is a set, so, and if and only if zero is in the subgradient. So that's the condition.

One way is completely trivial to prove. It's the same thing, right? Because in fact here's what a subgradient is. The definition of a subgradient is this. The definition of a subgradient is that for all Y, this holds. But if zero is a subgradient, it says that for all Y, F of Y is bigger than or equal to F of X. So one direction here is just a tautology.

I mean, the other direction is just as easy as well. In fact, it goes the other way. Suppose I told you X is the minimizer of F? Global minimizer. By definition, that means that for all

Y, F of Y is bigger than F of X. That's what it means to be the global minimizer. Therefore it's for sure true that this inequality holds when G equals zero. That means zero is the subgradient, okay?

So, now when you first see this you think, nothing happened. That was completely trivial, that was circular, there's nothing there. So in other words, not only is this true but it is completely trivial.

Okay, that's sort of it. By the way, this case is an interesting one, right, and it shows you – it says that it does not say that zero is the only subgradient. And in fact in this case there are many subgradients. What you can say about the subgradients here is they straddle zero, is what happens here, obviously, right? You have negative slopes and positive slopes, but zero is in the subgradient.

One comment: weak subgradient calculus will not help you with this. I mean, remember weak subgradient calculus, I mean, if you want to think about it operationally, is this: you have a method that will return a subgradient at a point. And then point here is I could easily return for this function. At X0, I could return the subgradient minus point one. And it is a subgradient, there's absolutely no doubt about it. It is not zero, right?

So weak – basically what that tells you is that stopping criteria based on weak subgradient mechanisms are not gonna be simple. Now if you have strong subgradient conditions you can do that, because you have something that somehow abstractly in some way returns the entire subgradient, and of course you have to have the containment question. Can you check if zero is in the subgradient or something like that? Actually, we'll see there's a very good way to check that.

Okay, so what's weird about it is that although it seems like we've done absolutely nothing, it turns out we actually have done something. It's kind of weird, but let's try it. So let's take these piecewise linear minimizations, you want to minimize piecewise linear function. And according to this trivial analysis, it says zero – the optimality [inaudible] zero is in the sub-differential of F at X star. But we know what that is, because this is the max of a finite number of functions, and that's equal to this.

It's the convex hull of the AIs where I are the active constraints, okay? So that's what the – and it's a polyhedron, that thing. Now the convex hull is just – I mean, another way to say it is just you're basically saying this: there are lambdas which are positive lambdas that add up to one, so it's like a probability distribution, and the expected value of the As is zero, okay?

Oh, and I should say this condition. In fact, this should look very familiar, it's a complementarity condition, right? Because it basically says these lambdas here – I assigned a lambda for everybody, but up here, that's the convex hull only of As corresponding to active ones, and therefore I have to make this thing mean the same as that. I have to say the lambdas are zero if you're non-active. This is what it means to be non-active. Yeah?

Student:[Inaudible] do that again?

Instructor (Stephen Boyd):Yeah, we did that last time. It was – yeah, we looked at – no, it shouldn't be obvious, but it's true, it's not hard to show, but it's certainly – it's actually not hard to show at all in this case. I think we looked at that last time, so that was our strong subgradient formula, right? It said that if you have the max of a bunch of things and you wanna calculate the sub-differential, you would calculate this way.

In the max you'd look at – first you'd single out every one who's active, and you'd screen for the active guys in the max, and then for each of those you'd ask them to produce a sub-differential at that point. Then you take the union of those and the convex hull, and that's the sub-differential of the max, the parent in the tree.

Okay, okay. Now this looks pretty familiar because they're exactly the KKT conditions for this LP formulation. So that's the epigraph form, and I should add, you know, this is how you would solve this problem. I mean, anybody here, this is how you, if you had to solve piecewise linear minimization, you would do it by basically forming that.

Or, for example, if you use CVX or one of these other high-level things, that would transform that for you into this top thing, this top LP, and then solve it. But whether you know it or not, you'd be solving that problem up there. And the optimality – the dual of that is this thing right here, it's maximized B transpose lambda, subject to lambda positive, and A transpose lambda is one, and one transpose lambda is one, and that means lambda's really a probability distribution.

And this should fit well with all sorts of things. For example, suppose you minimize a piecewise linear function in – it doesn't matter, R2. Let's minimize it in R2. And suppose when you find the optimal X, what you allege to be the optimal X, suppose only one of the functions is active there. What does it mean? First of all, could it happen?

Yeah. It means it can only happen if the slope is zero, and that'd be a piecewise linear function that looks like this, right? It only looks like that. But more generically, it's going to take a couple, and that means you're going to be at a point where you're going to be at a sharp point. So that's how that works. Okay.

Okay, let's do the constrained case. I mean, it's beautiful because it's sort of the generalization, it's the prettiest generalization of the KKT conditions. So if you optimality – I mean, I could add the equality constraints in, that's absolutely no different. It just would clutter the slide. So here, this is the constraint case, but now F0 and FI are non-differentiable. They are not differentiable, so you can't even talk about the KKT conditions there.

So these are the generalizations of KKT conditions, and they basically are the following. Obvious, if X is primal optimal, it's got to be feasible; that's a requirement for being optimal. The lambda Is have to be bigger than or equal to zero, that's obvious. And this is

the condition here, it's exactly the condition that would be gradient, and gradient if these were differentiable.

In the case where they're gradient and gradient, that would be equals because the right-hand side would be a vector, not a set. Here, it's a set, and of course everything here is overloaded for sets on the right. So that's what that is.

So the actually only difference, it's quite pretty if you switch out the symbol – well, you switch that to an equals, from an equals to an N, and you switch the gradient symbol to this symbol here, the sub-differential symbol. And you have to reinterpret everything properly.

So these are the conditions, and they're quite pretty, and it gives you a rough idea of how these things work. It's actually quite easy, I think, to show – at least one direction is easy. The other direction's not hard or something like that. Okay.

So now we're gonna cover this second idea of a derivative, so let's go back to the beginning and the issue is this: here's what you wanna do. You have a non-differentiable function and you wanna talk intelligently about a derivative of it. Well, I mean, the normal derivative doesn't exist, right? So that's – I mean, at some points it doesn't exist.

Even more than that, at points that we are unusually often interested in, it doesn't exist, right? Because if you're doing L1 or anything like this – of course if you do any kind of optimization, you're going to tend to hit these points that are corners and things like that. I mean, that's kind of what optimization does.

Okay, so one generalization is subgradient, and you've already seen kinda what it does. I mean, optimality conditions are beautiful, all sorts of other stuff comes out very nicely with subgradient.

But it might not be even the first one you'd look at. The first one you might look at would be something called a directional derivative. So the directional derivative of a function is – well, it's what it sounds like. You say the directional derivative of F at X and in the direction delta X, so this is a function of delta X, is simply the limit, as H goes down to zero from positive of X, plus F of X plus H delta X minus [inaudible].

Now if it's differentiable, that thing there converges exactly to grad F of X, transpose delta X. In other words, if F is differentiable, then F prime exists and is a linear function of delta X, okay? In fact, the linear function is grad F of X transpose delta X, okay?

Now, this function here is extremely easy to show. This thing has to be – it's homogeneous, it's always homogeneous. What it might not be is linear, and we'll see simple examples where it is not linear, but it's homogeneous.

So if it's linear, it tells you this. Now this is actually not a trivial fact, but it's – and I'm not gonna go in – I mean, there's, like, nine books and things, many books, many, many

books on convex analysis that goes into the horrible details of this. But a very basic result is this, is if you have a convex function, roughly speaking – I mean, let's say and you're inside the – and you're in the interior of the domain, you're away from the boundary, then you have a directional derivative in all ways, all directions, okay?

So, I mean, that kinda makes sense. Let me go over here and draw some pictures. So if you have a function like this, right, you're not differentiable there, but you have a directional derivative. And in this case, it's homogeneous, but a homogeneous function on R, a positive homogeneous function on R, you only have to say what it is from the left and from the right, right, because you know what it is for plus one and minus one, you're okay.

And in fact, in this case, F prime of X and then one is what we would generally call I guess F prime plus of X? That's the right-hand derivative, and it's this slope. And then if you do this like this, minus one, that's the directional derivative in the direction minus one, you get in this case F minus prime of X.

Now if you're differentiable, these two things are equal and they both coincide with the derivative. In this case, there's a gap between them. And for example, this thing – well, they're both negative, but that's less negative than that, okay? Everybody see that?

But what's interesting about this is this happens in RN, not just R. So take some horrible function in RN, non-differentiable, go to some point where it's got all sorts of sick stuff there. The point is that in any direction you move in, if you just move along a ray, it will be differentiable, it will have a directional derivative. Okay? Now if that directional derivative function is linear, then it means the function is actually differentiable. That's directional derivative.

Now the question is how is it related to a subgradient, and in fact they're related by duality. So, which sort of makes sense. And I think I can – yeah, so I think we can – so here's the general formula. It says this: the directional derivative in the direction delta X is the soup of G transpose delta X where G is in the sub-differential. And so that is the – this is called the support function of the sub-differential.

So to write a very compact expression for it, you'd say that the support – so you could write something like this: F prime equals, you know, the support function sub-differential F at X. So that's just a very compact formula that tells you this.

And let's actually look at this and see if we can understand how this works. Let's do the differentiable case. So in the differentiable case, what is the sub-differential? So your F is differentiable at that point, what's the sub-differential. What?

Student:[Inaudible]

Instructor (Stephen Boyd):Single point. Okay, it's a single point. So therefore, what's the supremum of, you know, the single point? In fact, that single point is grad F of X,

right? So what's the supremum of grad F of X transpose delta X? Sorry, there's no supremum because there's just one point. But then it just reduces to grad F of X transpose delta X, which is exactly what that's supposed to be, okay? So it's linear.

Okay, so that works out. And now let's try to figure out like in this case, what it is geometrically. So here's the sub-differential. By the way, when you see a big sub-differential like that, actually you should be visualizing what that function looks like at that point. Like for example, is it sharp? First of all, is it differentiable at that point?

No, because it's got a big old subgradient. Then there's clearly non-differentiable, and the question is is it a knife? So you know what I mean locally by knife? Knife is something that has a few sides that go in like this and you can cut with. Or is it a needle? These are technical terms, I'm not insane. I mean, sure, I just made them up, they didn't exist 14 seconds ago. But they make perfect sense, okay? So is it a knife? Why not? Exactly.

So a knife would correspond – a knife in the function like this, a crease – let's call it a crease. Whatever, it doesn't matter. If you had a crease, if the sub-differential gets big enough, it becomes a knife, it becomes dangerous. But let's just say it's a crease. So if it's a crease, but a one-dimensional crease like this, it means that actually in one direction it is differentiable, and therefore in that direction, the subgradient, sub-differential was small, and it would be a line, just like you said, a line segment.

This is big, and so this means this is a point. This is like a needle. So in other words at that point – oh, by the way, is that point global? Is it globally minimized? Is that point supported by a flat – is it at the bottom? How do you know?

Student:[Inaudible]

Instructor (Stephen Boyd):Zeros in, precisely. So it's a point, but it's not at the bottom of the function. It's sort of off like that. So you have a point over here. Okay.

And now you can actually calculate some things. So it says if you go in this direction – by the way, does the function increase or decrease if I step in this direction delta X? Does it increase or decrease? I'm asking what's the sine of F prime of X semicolon delta X for the one shown. Does it increase or decrease if you go in that direction? It increases.

Yeah, because – I mean, there's no scale here. So the point is that point is positive. That dashed line kinda shows you it solves that little – in this case that's an LP, because I drew this as a polyhedron. That's an LP. That shows the optimum value. It increases, okay?

Now let me ask you a question. Suppose you say okay, no problem, if I go in the direction delta X, the function will increase. You go, not a problem. Now we'll go in the direction negative X. Please tell me what happens if you go in the direction negative X. Negative delta X, sorry. Negative delta X. What happens now? Unclear? No, I think it's clear. You go in the direction –

Student: Increase, and if you go in the other direction, it'll decrease.

Instructor (Stephen Boyd):That's true, it's not guaranteed. Okay, so that's – oh, you know, damn, I drew it the wrong way. Sorry. Do you mind? Let's just visualize this thing going down to here. It is – your answer was exactly correct. It's unclear because you'd have to project a normal out – well, hm. What do you think? What do you think [inaudible]? I think you decrease if you go in that direction.

But let's imagine the subgradient went down here like that. It might have, right? So it goes down here. In this case, if you go in that direction, you can go a positive amount and that says you get this – so actually here I can make another one. How about this, if I go in this direction, you increase. Why? Because the supremum of X2 over this thing is the height.

The highest height you can go above this thing is positive. I mean, go up. If I go this way, if I increase X2 a little bit, my function goes up. If I decrease X2, my function goes what?

Student: Up.

Instructor (Stephen Boyd):It goes up, okay? So let's try that again, okay? If I increase X2 a little bit, I increase the second coordinate, the function value goes up. You say no problem, okay, so I will decrease X2 a little bit. And what happens to the function? It goes up again, okay? So my comment on that is welcome to the world of non-differentiability, right? Because if it's differentiable, then if you go in one direction – if a function is differentiable and you go in one direction and the function goes up, then you just – no problem, you turn around and you walk the other direction, it's gonna go down.

Why? Because you get a linear approximation nearby, so. And if a linear function goes up in one direction, it has to go down in the opposite direction. Everybody okay?

So in this case it means that that crease or point or whatever is such that X – and if you move X1 and X2, it's gonna – X1 and X2? Sorry, if you move X2 up or down, the function goes up. Okay.

Oh, let me ask you this, let's see. Oh, we'll get to that in a second. Okay, so we'll get to the idea of a descent direction. So a descent direction is – well, it's what it sounds like, it's a direction where if you step a small amount in that direction, the function goes down. Yeah, some – it can be less than or equal to zero or less than zero, but let's not worry about that. So that's a descent direction.

Now for a differentiable function, the gradient is always a descent – a negative gradient is always a descent function, always. You step a small enough distance in the direction of the negative gradient, and the function will go down, assuming the gradient's not zero, right?

So okay. Now for non-differentiable functions, this is the part that you have to sort of get, subgradients are not like – negative subgradients are not descent functions. Actually, this is going to be useful because later today we're gonna have algorithms that are gonna step in negative subgradient directions, and that's weird, because you're stepping in a direction where the function goes up.

You might ask why would anyone do that. Well, we'll get to that. But that's the story. So here's an example, it's this function, and this is a valid subgradient. That's a valid subgradient at that point. I mean, if you draw the – you extend the line, you'll see it's a subgradient.

Okay, now what would happen if you step in the direction negative G? You'd go in this direction, and you are pointing out, you're aiming out of the level curves. You're going uphill. So here, if you go in the direction negative G, you are going uphill. Okay? Then you say, no problem. Let's go in the direction G. How about G? Is that an ascent direction, descent direction? It's what? If you go in the direction G.

Student:[Inaudible]

Instructor (Stephen Boyd):It's an ascent direction, yeah. It's worse. But the point is that you got big trouble here. So if you step in the negative subgradient direction, it is not a descent direction, okay? So that's – and you know, this is kind of – well, anyway, so that's it. But it's something that's very important to understand, so what's gonna happen is for non-differentiable functions, the concept of a derivative bifurcates into two things, and they're kind of duals, right?

One is the directional derivative and one's the subgradient. So half the stuff you know, half of your [inaudible] about derivatives is gonna extend – let's say 48 percent of this stuff is gonna extend to the subgradient, and another 48 percent is going to extend to the directional derivative, and 4 percent will just be false in a non-differentiable case. I made those numbers up.

Okay, so then you might ask if you – is there a way to characterize the subgradient, and there is, and it's very interesting, and it's gonna be the key to understanding all of these – well, I could write down the algorithms right now, because they're one line, the algorithms we're gonna write down. They're gonna be one line, so we'll get to them later today.

I can write them down, but to actually understand it, this is the key to it. Otherwise it just looks totally bizarre, even though the proof is, like, three lines. So here it is. So what a negative subgradient is is this: a negative subgradient is a descent direction for the distance to the optimal point. Actually, the distance to any point that's better than the current point.

So let's just go back and check that here. Yeah, here, okay. So if you step in the negative subgradient direction – I'm not gonna write on it, don't worry. Oh my god, look at that.

The whole time I could have – this is great, okay, wow. Now it's not – I feel like I – oh well, anyway. All right, so this is – wow, look at that. Now I can actually see it now.

So okay, so negative G is you step in this direction. We've already discussed, if you step a small amount in the negative subgradient direction here, your function goes up, okay? Which does not sound like a winning move if you're trying to minimize the function, okay?

Or I should say if you're trying to minimize it greedily, this is not what we call a winning move, okay? However, the interesting part is that's the optimal point, the minimizer, and so what this thing says is if you step a small enough amount in this direction, your function went up but your distance to that optimal point went down, and that is actually correct, okay?

So basically we're gonna see a bunch of algorithms – boy, are they simple – and they're gonna – but the Lyapunov function – so does everyone know what a Lyapunov function is? So [inaudible] know? Okay, so in tons of fields, you wanna prove something converges or something happens or some condition holds, like in, I don't know, in computer science you wanna verify that something, you know, some condition always holds, for a dynamical system you wanna verify that a certain control system on an airplane will always – you know, where you'll never deviate more than 100 feet from your altitude or this kind of thing.

In optimization you wanna prove that something in an algorithm converges, and so people have lots of – one very general approach to this is to come up with a function that decreases at each step. It may be the function you're interested in. For example, if you're minimizing a function, it could be objective value. That'll be false here.

But generally, it'll be some – in any case, it can be any old function, right? And it's a function that's just going down. And that's the key to kind of proving something converges, okay? So that's called the Lyapunov function in control, it's called – in optimization, one name used for it, or algorithm design, it's called a merit function.

And, you know, you'd say the merit function goes down. Of course, I guess if it goes down it should be called a dismerit. Well anyway, it doesn't matter. It goes down. And it may not be what you want. I mean, it may not be the thing you're interested in. For exactly, very often in control systems it turns out a lot of these functions, which are Lyapunov functions, actually come from, like, energy-like calculations and things like that.

Okay, so all right. So what this means is – and by the way, all the algorithms you've seen so far, or differentiable things – you've seen Newton method, you know, gradient, scale gradient, blah, blah, blah, all these things, they're all descent methods. Oh, actually, we saw one that wasn't, we saw one that wasn't. Exactly one, and that was the [inaudible] Newton method.

But all the ones that you actually messed with were descent methods, in the sense that the merit function was the function. So at each step, the function went down. Of course, it's not enough to say that the function goes down at each step to prove convergence, except that a convergence has some value. You still have to prove that the value it converges to is the optimal one. But that's a start.

So it turns out here, what's gonna go down in algorithms based on subgradients is not the function value. The function value's gonna go up. In fact, it's gonna often go up. What's gonna go down, actually, ultimately, is the distance to the optimal set. That's what's going down. Of course, you don't know what the optimal point is or anything like that, but at least this is what's going down. Okay.

So to show this, you simply write this out and you write out the square. I write – I couple these two terms and I get that norm squared. I get then the norm squared of that guy, that's here, and then I get this cross-product, okay? Now here, this, I replace this with this, and this goes up because of my inequality here, and that's just the definition of subgradient here.

It says basically that – and G is the subgradient at X, or a subgradient at X, so that – I don't know, should I do this? Sure, why not? So it basically says – you know the following – that F of – let me get it right. Here we go. You know this is – you start from F of Y is bigger than F of X, plus G transpose, but should I have used X and Z or something? Well – yeah, well, you use this one.

And then you subtract, and I should have called that Z, maybe, or I should swap these or something like that to get it the right way. Do I want F of Z? Yeah, let me make this F of Z to make it work out nicely, like so, and then I claim that inequality shoved in here does the right thing. It does, because you just switch it around. That's F, so this says basically something like this: F of X minus F of Z is less than or equal to G transpose X minus Z.

I think that's what I want. And then that's good enough. So it says that – let's see, it says that this thing is smaller than that. It's a minus sign, so everything's cool. Okay?

All right, so – oh, and there's one more step of the proof, that's this: you choose – so actually, let's analyze the terms. I want that to be less than that, okay? That's not helping, because it's positive. But it scales by T squared. This term absolutely helped, because the assumption here is that F of Z – this is less than that. If this is the optimal point, if that's X star, this is positive.

In any case, [inaudible] positive, and so that's a minus sign, T is positive, that's positive. So this is the good term and that's the bad term. But the good term scales like T and the bad term scales like T squared. So for a small enough T, the good term overwhelms, and this thing is strictly less than that, okay? So that's the picture. Just save that as an idea.

Okay, descent directions and optimality. So if you have a convex function, finite, you know, near a point X, then two things [inaudible]. Either zero's in the sub-differential, in

which case you're done, right? But it's not particularly useful unless for that particular problem you actually have a handle on a strong subgradient calculus or something. If you have a method to calculate the full sub-differential, then this tells you something. Otherwise, this is useless. Okay.

Otherwise, if you're not optimal, there's a descent direction, and that says – there's a descent direction, and in fact the steepest descent direction is this: it's the smallest point in the subgradient in the sub-differential set, okay? And we can draw – let's see if I've even drawn some – yeah. So here's a picture. So here's a sub-differential, our function is quite pointed there, right?

It's got a – it's quite pointed. And zero is not in that function, so that means there is a descent there. It means number one, you're not optimal. So then how do you find a descent direction? You find the smallest point in that sub-differential, and that happens to be that vertex shown there. I can use my new technology – okay, there.

So that's the closest point to zero. That's the smallest point in the sub-differential. And then if you go in that negative direction, that – by the way, this thing, that is a subgradient. Well, of course, because it's right here in the sub-differential. This subgradient happens to be in the actual descent direction for the function. Why? Because if you find the smallest point, just the optimality conditions for finding the smallest point, is that well, if the smallest point were zero, you'd be done.

So if it's not zero, it says if you go in the other direction, you're completely free and clear of this. And that says that there's actually a descent. And the distance here actually gives you the direction, something like the negative directional derivative, okay? So that's the picture.

By the way, let me ask you a couple of questions. Is this a descent direction? It's going in that direction. See if your – how your [inaudible].

Student: Yes.

Instructor (Stephen Boyd):It is, okay. How about – how about if you go in this direction? Okay here, here, how about that? Is that a descent direction? I think that one's – what do you think? That was, like, absolutely not. Because here, you would go out to a point there and you'd have very – the soup, the support function evaluated there is going to be quite large and positive. You're gonna increase.

That's, like, ridiculous if you go in that direction, right? That's almost like the steepest ascent direction. And then down here is – how about, you know, if I went along this axis? If I stepped along this axis, would it go down or up? You go along this axis, what do you get? No, no, it goes up, way up. You evaluate the support function. You ask yourself how far can you go in this direction in this set, and the answer is way over here, and that's way, way positive.

So in this direction you go down, weakly, in fact – in fact, it depends on this little gap here. If you go in this direction, you go up pretty steeply. How about if you go, like, in this direction, here? That a descent direction if you go like –just staying along this line here. Is that a descent or not? No, that's an ascent direction. And if you go in this direction? Ascent.

Okay, so – by the way, you couldn't possibly have this with a differentiable function, right? I mean, there's no way you could go have an ascent direction one way, turn around, go the opposite direction, it goes up there, too. So, impossible.

Okay, so this is the picture. Actually, what this means is that there's a – this is the basis for a whole bunch of methods – by the way, not one of which we're gonna look at for minimizing non-differentiable convex functions. It goes like this. It basically says the following: at each step you get the sub-differential. Then you solve this problem.

You minimize over the sub-differential the norm. If you find that zero is in the sub-differential, which you would have to find if you actually solved this problem, then you stop and you actually have a proof of optimality, because you have zero in the sub-differential, okay?

Otherwise, you find the smallest point in the sub-differential, and you step in the direction, the negative of that direction. And that's called steepest descent for non-differentiable functions, and we're not gonna study it. It kinda works.

And if you read more about these other methods, you'll find that that's sort of a theme. The methods that are sort of based on this kind of idea that sometimes work quite well are things called bundle methods. You will definitely see that if you poke around. It won't take you too long to find those.

Okay, so now this covers all our analysis of – what is all that? [Inaudible] Well, why not? [Inaudible] Mm-mmm. Hm. Interesting. Will I be able to do this? See what happens – no, hm. Oh, let's see now, okay. How about just finding a terminal – oh, no, that's maybe [inaudible]. Sure, okay. Everything's fine, don't worry about a thing here.

Sure, okay, and that was – no, no, no. No. All right, all right, well, we'll just have to figure out what's here. Hm. Okay, better. Where is it? [Inaudible] method – slides, there we go. That gonna work? Let's see. Look at that, see? Okay, nope. Great. Yeah. There we go, okay.

All right, okay. So what we did so far is just analysis, just analysis. It's cool analysis. Some of it was not useful, I mean, basically. The stuff that requires strong subgradients like the steepest ascent stuff is not – that's not particularly useful, I think – I mean, or just it seems in practice it's not.

Subgradient methods are, so these are – we're gonna look at these. I can say a little bit of history of these, you'll never guess where these come from. Actually, you might be

wrong, because some people might argue they come from Kiev. So they're either from Kiev or Moscow, that's your only two possible choices.

So you'd say both, okay? So yeah, so guess what? These methods are from the Soviet Union in the sixties, surprise, surprise. But what's so weird about these things is how simple they are, and they look really stupid. In fact, I'll tell you a story about it in a bit.

Okay, so here's a subgradient – and I'm not kidding, it really is this simple. Here's what it is, ready? You take the current point, you find a subgradient, and you step in the negative subgradient direction. Multiply it by step size. So far, that's fine. So – and this is a weak subgradient. Any subgradient – by the way, including a subgradient where the negative subgradient is not a descent direction.

And by the way, what that means is this is not a descent method. You'll run this algorithm and the function will go up at certain steps. Do you have a question? Yeah.

Student: How do you know they're subgradients? Which one would be the best [inaudible]?

Instructor (Stephen Boyd):Well, if you knew all the subgradients, you would probably do pretty well by taking the [inaudible] of the norm, which would be the steepest descent point. However, interesting thing is I think these methods are actually structurally slow – they're slow no matter what. So in fact the nice thing about these – the bad thing about these methods if they're slow.

By the way, they can be fast in theory. The bad news is that they're slow. The good news is – well, in a weird, perverse sense, the good news is they don't get much faster if you're much smarter and try to calculate fancy subgradients and things like that. They go from being, like, terribly slow to quite slow. Now let me explain why that's good news.

That's good news because that's just work you don't have to do. You don't have to write some – I mean, you can be totally sloppy, right? I mean, I'll give you an example. Suppose you're doing minimax and you have the maximum of a thousand functions you wanna minimize. That comes up all the time, okay? So you could do this, you could evaluate – you know, okay, to evaluate the max, you have to evaluate all those functions, right? You have to evaluate a thousand functions, calculate the maximum of those.

Now, typically some will be tied. If there's no ties, no problem, right? But there'll be some ties. If you keep track of the ties and calculate that convex hull and solve a QP to get that best one, you could say – you know, [inaudible] look in the code, you'd say look, I'm being super-smart, I'm getting the best subgradient and all that, and this'll work slightly better. So you went from – anyway, so it's just not gonna have a big effect.

Okay. All right, so as I said, it's not a descent method. So because it's not a descent method, in a descent method if someone stops you early and says, what's the best point you've got so far? It's the current point, right? Because you make progress every time in a

descent method, so there's no – [inaudible] another concept of the – you only have the concept of the current point.

So with a non-descent method, though, we'll encounter several throughout the class, actually, you need to keep a pointer to the best you've found so far. You just – you know, if it gets worse, you keep going but you save somewhere the best point you found so far. Okay, and we'll call FK the best point found so far.

Okay, so here's some step size rules, and let me remind you how this – when you first see this, you say hey, that's just the gradient method generalization to non-differentiable functions. Let me assure you, it is not. It just looks like it. So – in fact, I can even tell you a story about it.

So the first time I found out about these methods I was a grad student, I was reading, and I got some Russian paper on something and at that time, I guess still now, there's something called Automation and Remote, or one of these things that was translated basically by a company funded by the CIA translated all the Russian papers badly. But I got very used to Russian typesetting, because they didn't – they just kept the typesetting the same, but the English was written by somebody at the RAND Corporation, something like that.

So I conjectured that they didn't know what they were doing, and the original Russian, although I haven't verified this, had subgradient, but it got translated as gradient. So I'm reading this paper, I'm on, like, page two, and it says we'll just use a gradient method.

So we'll just take a gradient – and I'm looking at it, I'm like but just a minute, that function is non-differentiable. There is no gradient. But then they looked just like that, and I said well, that's the gradient method, for sure. This function's non-differentiable. There's no way. This can't – and then I thought – so then I had to go cool off and thought okay, either this guy is a complete – has no idea what he's doing, I mean, at all – I mean, this is the most fundamental thing – or there's something very interesting here.

So it turned out to be the latter. So it was the subgradient method, badly translated as gradient. So you'll still find that. By the way, if you find Russian literature, you will actually find people referring to gradient, meaning subgradient. So you have to be very careful.

Okay, all right. All right, so this is gonna be – now in the gradient method that you know and love for differentiable functions, it works like this: you calculate – this is the gradient, and you calculate alpha K lots of ways. One way would be an exact line search, so you'd find out the best point.

You could do a backtracking line search, start with alpha equals one and times equals beta until you got some sufficient decrease. Everybody knows what I'm talking about, right? Okay. So you would chose alpha and the step length in a – I mean, you would

never choose it ahead of time. You'd never say that my 14th step size is going to be point two.

You'd wait until you got to the 14th iterate and try some step sizes, and if point two was the right thing, that's what you would do, right? I mean, that's it? Okay. So here are the step sizes for the subgradient method. I mean, it's just – the whole thing is looking really implausible, if you ask me. I mean, it just – you know, we start with the fact that you're gonna move in a non-descent direction when you wanna minimize a function, and the next part that really puts it over the top is this.

Is that for these methods, you basically set the step sizes ahead of time. In other words, you don't even adapt them. I mean, there are fancier methods where you adapt what the step size, depending on where you are. But basically, these methods work fine if you just say the step sizes are – for example, here, ready for some step sizes? One over K. So in the [inaudible] iteration, your step size will be one over K.

Oh by the way, the function has some kind of domain issue, right? So you can't do one over K. You have to modify that. But the point is here's one, just one over K, here you go. So the whole thing's ridiculous. I mean, here's some step sizes, all very implausible. You have a constant step size, another one, here's a constant step length. That means that you get the subgradient here, you divide the step length, and that means that your actual – the distance you step is exactly gamma here, okay? Even more – these are just, like, if you think – these are just ridiculous. Highly, highly implausible rules for step length.

Another one that's gonna come up is this, is step lengths which are square-summable but not summable. And so this would be, for example, one over K would be one. And then you have non-summable diminishing and this would be one where the limit of these things go to zero, but they're non-summable.

Obviously, this implies that, so. So this is the weakest. And the classic example here is one over square root K, and why that is will come up in a minute.

So I just wanna emphasize how silly this algorithm's looking. So it goes like this, it says you're at a point X and you call get subgrad at X, so you – and this thing returns. It has many choices of subgradients, I mean, if it's non-differentiable, and it returns one. And there's no restrictions on which one it returns.

In fact, multiple calls at the same point could return different subgradients. It makes no difference. Doesn't even have to be deterministic. Just returns the subgradient.

Okay. Which need not be – the negative of that need not be a descent direction. Could be an ascent direction. And in fact it could be even worse, because your alpha's gonna be positive. But in any case, it could be even a direction where either direction you step, the function goes up, okay?

Then you'd say well that's okay, because there'll be a sophisticated line search to take care of that. No, the line search is basically predetermined. It's just fixed, it's like point one, that's the line step.

So, I mean, I don't know about you, but this is not sounding like super – I mean, it doesn't seem that promising when you first encounter it.

Okay, so there's a couple of assumptions here. We'll just say let's assume that its actually [inaudible] below and there's an optimal X. And then we'll take a Lipschitz constant on the function F, and that's the same as saying that the subgradients are bounded by G – I mean, roughly.

And then the other thing is we'll take R to be the distance of our first iterate to the optimal point, so capital R. So actually capital R, especially if it's inequality, is something like it's a distance measure, it's measured in meters, if X were in meters, it's a distance measure of our ignorance, because it's how far.

Before you start the problem, if someone has given you an X one and an R and a G, and if someone says how far are you from the optimum, you have to say R. By the way, how far are you from optimum in terms of the function itself? You've done nothing. You've only been given the following data: X one, R, and G. How far off is F of X one from F of X star? It has an answer. RG, thank you.

So RG, R is your measure in meters, if X is in meters, of your ignorance. I mean, before you even start it, if your prior ignorance. RG is your ignorance in F star. Because if someone says you have F of X one, and someone says well, how suboptimal are you? You'd have to say I could be up to RG suboptimal, right? So actually just bear that in mind, because RG is gonna come up.

Okay. Oh, by the way, I should say that the proofs I'm gonna give here are super-simplified. You can go read the Russian ones if you like. They're actually a couple of pages longer, but they do away with, for example, this one, okay? So you don't need this. This is just because I like groups that fit on a half a page or a quarter of a page or something like that. But anyway, I think this is the main idea.

Okay, so here's some convergence results. It says if you run the subgradient method with a constant step size, then the following will occur: that the best – so the limit is [inaudible] infinity of FK best. By the way, this is the same as lim int of FK, of FK, right? That's what – it's of F of XK. It's the lim int of that, because the best is the [inaudible]. Did I say that right? I did, yeah.

Okay, so this is – so anyway, this goes to F bar. I mean, this thing goes down, because your best function value to date, that goes, that [inaudible] decreasing. So it has a limit, and we're gonna call that F bar. This could be, of course, F star. But here's what you're guaranteed. You're guaranteed that F bar minus F star is less than G squared alpha over two, and that says that if you use a constant step size, then you will end up being

suboptimal, but with some number that you can actually, you know, say what it is. So you'll converge to suboptimality, some suboptimality.

Okay – oh, I should say this. If the function is differentiable, then with constant step size you actually converge if alpha's small enough. But for non-differentiable, that's absolutely false, you will not. And it's actually easy to see that. Actually, a good example is just to apply the subgradient method to minimize the absolute value of X, okay?

So let's just do that in our heads, right? So you take absolute value of X. How do you get a subgradient? You're at a point X, and you get a subgradient for absolute value X. What's the subgradient? If X is positive, what's the subgradient?

Student: One [inaudible].

Instructor (Stephen Boyd):One, it's plus one, there's no choice. There's no other answer. If X is negative, what's the subgradient? Minus one. If X is zero, you can return any number between minus one and one. Okay? So for some reason, our code, it's gonna return minus 1.75.

Student:[Inaudible]

Instructor (Stephen Boyd):Hm?

Student: That's what – if you [inaudible] minus point five there.

Instructor (Stephen Boyd):Minus point five. No, you mean zero.

Student:[Inaudible]

Instructor (Stephen Boyd):[Inaudible] If you return zero, if you're at zero, you're making things easy because then the algorithm just terminates. Because if return zero is the subgradient, that's the condition for optimality, and it can quit. No, I don't wanna do that. I'm gonna return minus .75, just because – some sick reason, because who knows? Yeah?

Student: That goes behind what I was saying, that you could choose [inaudible].

Instructor (Stephen Boyd):Yeah, I know what you were saying. Yeah, I'm telling you, it's not – it turns out – everyone assumes this, so you can – well, we can, you can do a project on it. Everyone assumes, they think that this method, you know, it's so – that by clever choice of the subgradient you can do better. That is probably a true statement, but it gets to be a huge, long – I mean, this is on a [inaudible] somewhere that trades off simplicity of code, which is to say one line, with simplicity of proof, which we will get to maybe in, like, two lines, and the actual how well it works, which is slowly.

You can then have methods that are super complicated, try to estimate the best subgradient and blah, blah, blah, and bundle – they get all fancy and all that. Sure, they work a bit better, but they're, like, way longer and all that kind of stuff, so. But still, my basic comment on that is you'd be shocked how little you gain by being smart and returning a so-called good subgradient.

So, okay. So anyway, now let's imagine and make it simple that you return plus one if you're positive, negative one if you're positive, and if you're at zero, you return plus one. And it's a valid subgradient, can't say it's not. Okay? So now you imagine where you are. You're somewhere, and we do a fixed step – let's do a fixed, you know, a constant step.

They're the same, I guess, right? Because the gradient – okay. So then there's zero to point X, and so you step alpha. If you're positive, you go down alpha. Everybody visualizing the subgradient method? So you go down. When you go over, when you switch sine, now you go up alpha, right? And now you simply go back, you just oscillate like this. Everybody got this? Okay, you are not optimal, right?

Because you go there and it says the subgradient's plus one, it says go that way. You step that way, your sine changes. You go the other way. So that's – so you can see the – and that's, by the way, no matter how small alpha is, you will not converge, because you'll end up in an oscillation like this.

Okay. If you have constant step length, you again converge to something that's suboptimal. And here's the weird part. Diminishing step size rule? It converges. So actually, what we won't do is I think today we won't do the – go over the proof of that, which is, like, shockingly short, but I just wanna point something out.

Ready for – here's some diminishing step sizes. How about one over K, okay? So ready for the algorithm? Just put one over K there, okay? And you now have – and now here's the – ready for the theorem? It's this: it converges to the global [inaudible]. That's how stupid it is.

I mean, you have to understand how dumb this is, right? It's basically, you know, for I equals one to whatever, little K equals whatever – little K equals one [inaudible] capital K, it's basically X minus equals subgrad, subgrad F of evaluated at XK divided by K. That's it. That's one line, okay? And it converges.

I don't know if you're getting the thing. I mean, it's fine. You know, you might ask, well, why didn't we do this in 364? Well, this is kind of cool, but it's – I mean, these things, as you'll see, they're gonna be very slow. They will have some redeeming features, so. But they're gonna be quite, quite slow.

But still, this is very shocking, if you think about it. You also could say – I mean, if you also believe this meta-statement I made last time, which went something like this: that if you can evaluate a function, you can evaluate a subgradient. If you believe that, then basically what this says is you can solve any convex function. Any convex problem for

which basically if you can evaluate the objective and constraint functions – we haven’t gotten to the constraint case, so; but we will.

But it says if you can evaluate the objective function, you can minimize it. Period. So everybody – and the algorithm is, like, it's barely a line. It's not a line, it's 15 characters, right? That's it. I mean, it's that stupid. It's weird. So when you first see this, it's kind of shocking. We'll continue this Tuesday.

Actually, now that people are here, maybe I'll just repeat a couple of the things when people were in between this place. So the most obvious new development in the class is that we're here, and televised. But we'll try to keep it informal too as well, so we'll just pretend it's not here.

And if I say anything really way totally off base, we'll just go in to – I'll go to my friends at SCPD and we'll cut it out before it gets posted, and the people at MIT look at it and then I'm in big trouble. Or the police get it, or you know, whatever.

So we'll do that.

We posted homework one, and I don't know that there are any other – maybe there aren't any other announcements, so. Okay, so we'll quit here.

[End of Audio]

Duration: 74 minutes