Distributed Optimization and Machine Learning

Prof. Mayank Baranwal

Computer Science & Engineering, Electrical Engineering, Mathematics

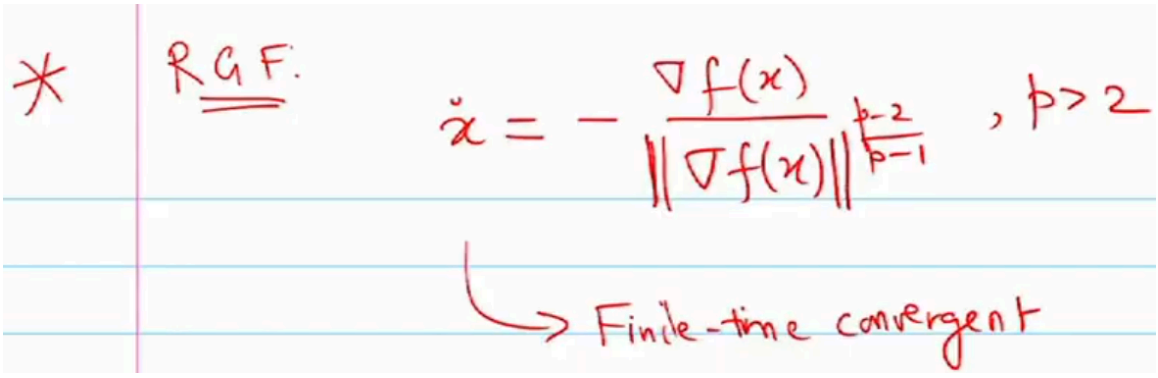
Indian Institute of Technology Bombay

Week-6

Lecture - 19: Advanced Results on PL inequality: Part 1

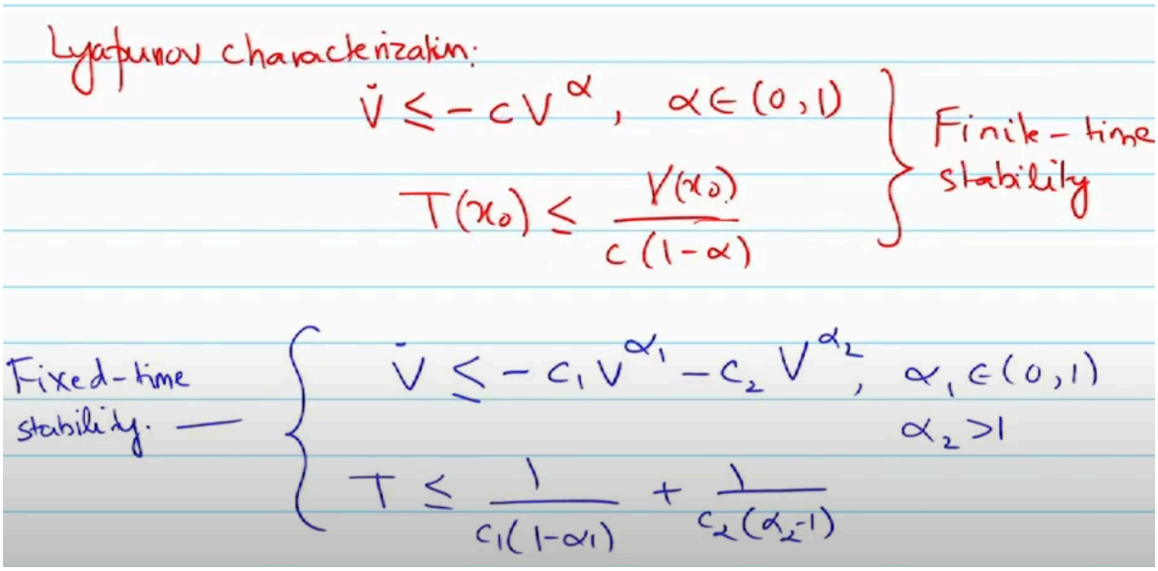
So, the focus for today's lecture is going to be somewhat I mean the content that we are going to be covering is some going to be somewhat more advanced most like very topical. In fact, we are going to be looking at some of the results that first came on 2016. So, which is more about like some advanced results on PL inequality and we will be using those results for showing the robustness of the optimization algorithm that we had looked at in the previous class. So, just to quickly recap. So, we started looking at this notion of finite time and fixed time stability and we looked at this particular rescaled gradient flow which was defined as x dot is some with p greater than 2 and this was shown to be finite time convergent. In fact, we looked at the Lyapunov characterization of finite time stability of equilibrium and so, if I look at the Lyapunov characterization.

So, if your Lyapunov inequality the time derivative of the Lyapunov function satisfies this particular inequality with alpha being a number between 0 and 1. And you can show the trajectories they converge to the equilibrium in a finite amount of time which is dependent on the initial condition x naught and that settling time function is actually upper bounded by c times 1 minus alpha. So, this was finite time stability. We can also talk about fixed time stability of such equilibrium.



If the Lyapunov function satisfies inequality of this form c 1 v to the alpha 1, c 2 v to the alpha 2 with alpha 1 between 0 and 1, alpha 2 is a number greater than 1. So, this you can show is in fact now fixed time convergent meaning that the settling time function is independent of x naught and this is going to be upper bounded by 1 over c 1 1 minus

alpha 1 right and this is for fixed time state. Yes, thanks for pointing that. So, this should be v naught or v of x naught ok. So, this is v of x naught alright.



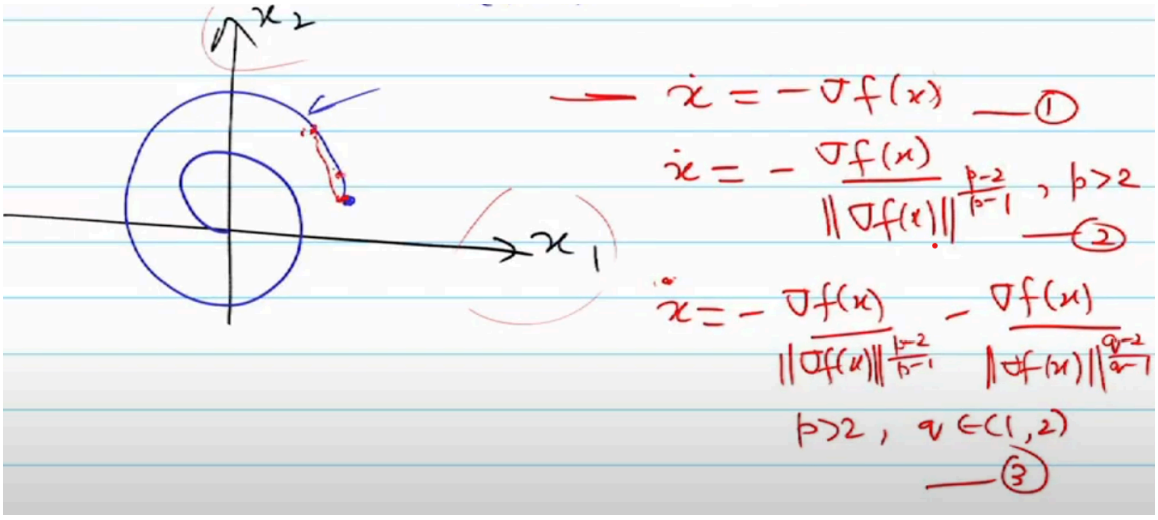
And as I said this is nothing but re-parameterizing your simple gradient flow. So, in terms of trajectories that your x 1 and x let us say x is 2 dimensional ok. Let us assume x is 2 dimensional. So, you have x 1 and x 2 and let us say origin is the equilibrium or optimal solution. So, if your gradient flow let us starting at some x naught, if your gradient flow let us say follows this kind of trajectory.

So, the path that is traced by this particular dynamical system or path that is traced by an equivalent dynamical system where you have fixed like I mean you have the notion of fixed time convergence, it is going to be exactly identical. So, it is basically we are accelerating the speed like we are essentially accelerating along that path. So, we are changing the speed with which we trace this path, but the path is going to be exactly the same that path traced by it. As a function of time x 1 t and x 2 t would look different obviously, but if I look at the path traced by this particular dynamical system, this is same as this is going to be the same as the path traced by a nominal gradient flow without normalization. for let us say if I look at this a simple gradient flow like this.

So, let us call this 1 a finite time convergent flow which looks something like this with p greater than 2 and of let us call this 2 and a fixed time convergent gradient flow p minus 2 p minus 1 minus with p greater than 2 and q is a number between 1 and 2. So, all these three dynamical system they are going to trace the same path. So, it just that you are doing curve reparameterization. So, in so essentially what you are doing is you are changing the like basically you are sort of changing the coordinate system and in the transformed coordinate you are converging maybe you are running this new gradient flow right. So, essentially you converge much faster and essentially you would change the speed with which you trace this path, but the path that is traced by all these three

dynamical systems these are going to be exactly the same ok.

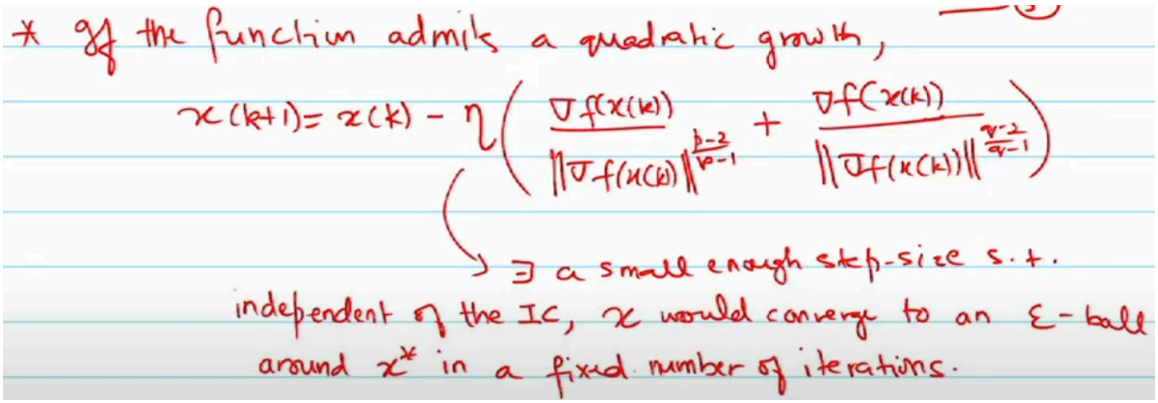
Yeah that would yeah any scaling would be the same though not when once you did once you discretize it that may not be the case right because you are then then sort of like you are talking about discretized updates. So, that may not be the same, but in continuous time they are going to be the same. No, because like let us say in for the discretized one from after this point. the update like for the same step size this simple gradient flow would suggest an like would suggest an x which will like which will be somewhere over here whereas, maybe a fixed time gradient flow would suggest an update which is somewhere over here right. And so, they would not be like the path trace there would not be the same because it is due to discretization, but in continuous time they would trace the same path.



here ok. So, that is another question that kind what kind of guarantees can we provide once we discretize these fixed time convergent gradient flow. So, well the guarantees are not of the form of that like for instance with gradient flow when you look at gradient descent for l smooth function we had a specific characterization of the step size which was in 1 over l kind of step size right. We do not have that kind of interesting bounds on the step size, but the kind of guarantees that you can provide is if kind of results that exists is if the function admits a quadratic type of growth admits a quadratic growth at max quadratic growth. Then if I look at this particular this particular discretized step let us say step size eta So this is basically the discretization of your fixed time convergent gradient flow. So the kind of results or kind of guarantees that we have is there exists a small enough step size such that independent of the initial condition initial condition you like one x would converge to an epsilon ball around x star in a fixed number of iterations.

the kind of guarantees that we can provide are only of existence type. So, there exists a small enough step size such that the even the discretized flow or the discretized fixed time like the discretized version of the fixed time stable gradient flow because in discrete

time you cannot talk about exact convergence to the optimal solution right and you can talk about in that in continuous time, but in discrete time even maybe as a result of discretization you may just slightly overshoot your x star or undershoot your x star and so on right. So, if you fix an epsilon ball around your x star, you would converge to that epsilon ball no matter where you start will converge to that epsilon ball within a fixed number of iterations independent of it, independent of initialization, but what would be the bound on the step size and things like that, that is in still an open problem. So, all you can I mean all we have is the existence type result and not the bound on the specific bound on the step size. So, that is I mean that is where the current sort of state of the art is.



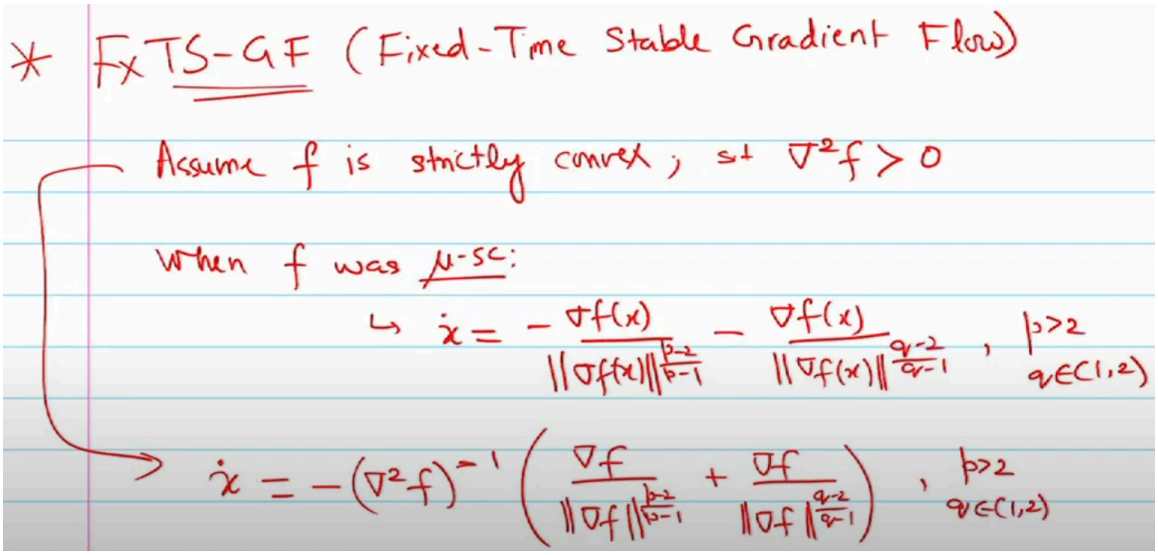
We want to know the number of iterations. Yeah. In that we want to bond over time. Yeah. So, here depending upon theta.

Yeah. So, we do not know that. The number of iterations as a function of theta, we do not know that. Yeah, yeah. Right, right, right, right, yeah. So in fact, if you have a dynamical system whose equilibrium is exponentially stable you can potentially make it fixed time stable by changing by this kind of normalization.

So, what like if you have any for any dynamical system that is exponentially stable this can be achieved. So, you just have to do the normalization and that would make it make that particular equilibrium fixed time stable or finite time stable depending on what kind of normalization you end up doing ok. let us say we are going to be working with. So, we want to design a gradient flow, a rescale gradient flow which is fixed. So, let us call this, let us give it a name.

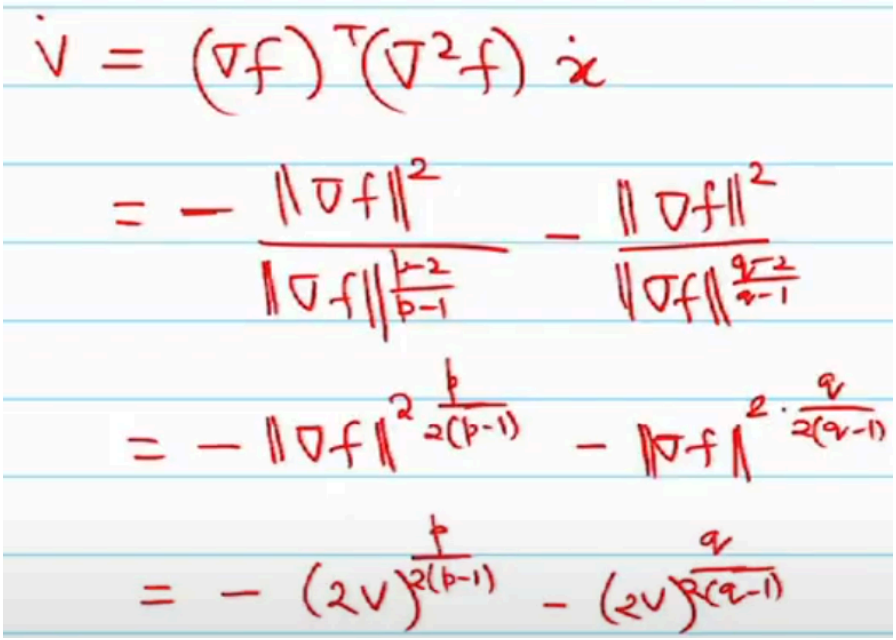
We will just from now on we will call it FxTS-GF which is fixed time stable gradient flow. Let us see. So, suppose I want to design a fixed time stable gradient flow for So, we assume f is strictly convex. So, in the last lecture we looked at the case when f was strongly convex right and we were able to design a fixed time stable gradient flow for that. But what if f is strictly convex? What kind of algorithm should work if f is strictly convex? Or let us say f is strictly convex such that hessian of f is positive definite.

right. So, the question is we want to design a gradient flow which is convergent in a fixed like which converges in a fixed amount of time. So, in the previous lecture we looked at the case when f was strongly convex mu strongly convex and we were able to like for the when f was strongly convex we had this particular gradient flow right simple p minus 2 over p minus 1 ok and we showed that this was convergent error fixed amount of time. Now, if f is strictly convex, how can we modify this? like basically Hessian inverse right the same we did like I mean the same thing that we did with the simple gradient flow right. So, when f is strictly convex. So, we are now going to be designing a new type of gradient flow which is going to be again it since everything is inspired from simple gradient flow and then.



So, the equivalent modification of the Newton's flow or which is of this form. ok and let us see if this works. So, when we have strictly convex function what would be a good choice for Lyapunov function like what is a good Lyapunov candidate for that half this thing right. Because once you start differentiating it you will get Hessian and you have an Hessian inverse sitting over here. So, V dot is essentially transpose hessian times x dot right and x dot is now hessian inverse times this.

So, hessian hessian inverse that becomes identity. So, what you are left with is v dot is negative Is this clear? So, this would mean that I can rewrite this as. So, 2 minus the same exactly the same approach. So, this would turn out to be right and similarly 2 times q over 2 k times right and this is nothing, but minus 2 v p over 2 p minus 1 minus 2 v q over 2 ok. And for the specific choice of p when p is greater than 2 p over 2 p minus 1 is a number between 0 and 1.

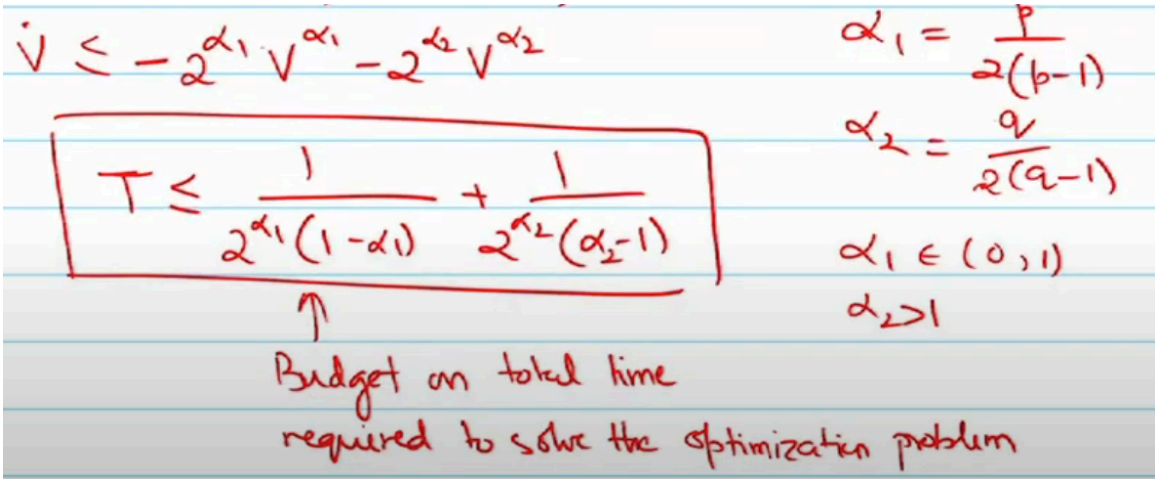


and for q between 1 and 2 this would be this exponent would be the number greater than 1. So, essentially we are in. So, that basically ensures that v dot is essentially less than equal to certain thing. I mean if it is equal to you can always write the like I mean write this as less than equal to and you recover the. So, let us call this to be alpha to be p over 2 p minus 1 and alpha 2 to be q 2 q minus 1.

So, what you have here is 2 to the alpha 1 v 1 alpha 1 minus 2 to the alpha 2 v of alpha 2 right. So, this satisfies the with alpha 1. So, alpha 1 is a number between 0 and 1 and alpha 2 is greater than 1 for these choices and therefore, that means this is fixed time convergent with settling time t less than equal to 1 over 2 to the alpha 1 1 minus alpha 1 plus 1 over ok. So, for strict like whenever you see a strict convexity with Hessian positive definite, it makes sense to use Newton's type of flow or a modified Newton's flow ok like this. So, it so that I mean this is basically an accelerated version of the same Newton's flow that we had looked at in the previous class.

Another interesting thing to observe is let us say I am looking at this particular optimization problem and I have a specific budget on how much time we want to allocate to solve this particular optimization problem right. So, I can choose my p and q in such a manner. So, that I basically come up with the bound on the time it is going to take to converge right. So, it is not just fixed time optimization it is also like predefined time optimization. So, if if you give me a bound that you want to solve this problem in this much amount of time you will accordingly choose your alpha 1 and alpha 2 that will actually like ensure that the settling time is upper bounded by that that particular time and therefore, you can you are guaranteed to converge in that much amount of time right.

So, this also if you have a budget on time. So, it also sort of specifies budget on total time required to solve the optimization problem . For fixed time now right. So, once you have these two. So, that is the difference between finite and fixed right.



if you have a gradient like gradient flow like this, then it depends on the initial condition, but if you have a gradient flow like this, then you make it independent of the initial condition. Yeah. Yeah yeah yeah definitely. So, that is why I said like the I mean you have guarantees in this like when you discretize it, but the guarantees are only of the existence type. So, for small enough eta this would still hold, but how small the eta should be it is specifically if like even if let us say f is strongly convex and l smooth can we still specify eta in terms of mu and l that is also an open problem.

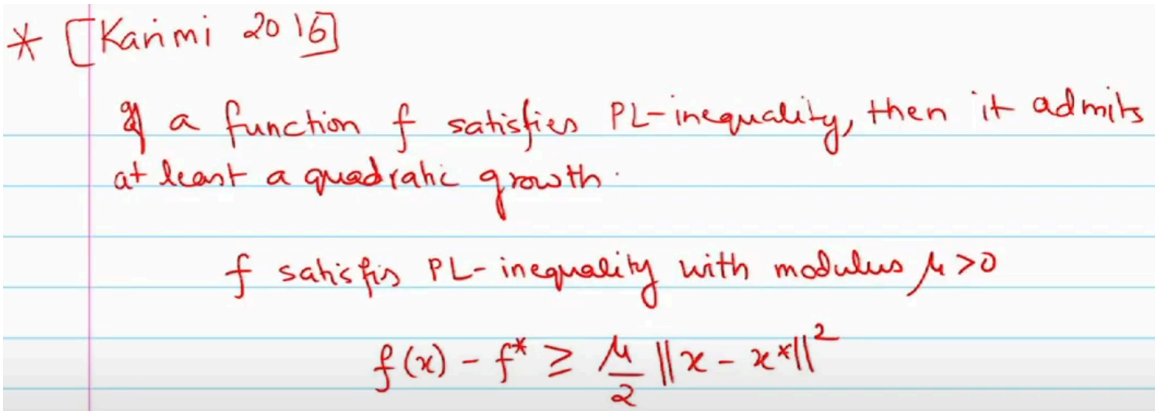
it also depends on, but then for the at least for Euler, Runge-Kutta all these standard discretization this I mean this result sort of holds true. Yeah, that is that there is a possibility that using simplistic Euler you can possibly make it work, but in fact if the system is homogeneous as in like 0 is the like x dot is of the form x dot is negative x. So, I mean that would not. So, for instance if I am trying to minimize x square mean I obviously know the optimal solution which is x equal to 0, but the corresponding simple gradient flow would be x dot is negative x or negative 2 x whatever right. But this is an example of homogenous system, if I am trying to minimize x dot I mean something like x minus 1 whole square.

So, x dot is negative of x minus 1 and that is a non homogenous system right. So, So in general, I mean for optimization it does not help, but if you have, if you are looking at just at the stability problem, forget about the optimization problem, if you are looking at the stability problem, if you have a homogeneous vector field to work with, then in fact you can specify this eta, you can specify the number of iterations as a function of eta. but only for the homogenous system. It does not help us in the context of optimization, but in the context of stability if you have origin being the equilibrium, then you can actually specify the amount of iterations it would take to converge to an epsilon ball around the

origin. So, only for homogenous vector field you have explicit bounds, but otherwise I mean the bound of on eta is I mean that is an open follow for non homogenous fields.

that can be No vector field, no cos, vector field in this case is always a gradient right, negative of gradient is the vector field. So, x dot whatever is there on the right hand side is your vector field, cos function is always f of x. So, obviously I mean since it works for strongly convex function, you can also extend it for function that satisfy PL inequality. So, we have been talking a lot about PL inequality, but what does PL inequality really imply ok. And this is again one of the seminal results from Karimi in 2016.

So, if a function and the result says, if a function satisfies real inequality, then So, then it admits at least quality growth. So, I will tell you what this means. Let us say there is a function f that satisfies PL inequality with exponent mu with modulus mu greater than 0. So that means f of x minus f star, so it is lower bounded by a function which grows quadratically. Function has at least a quadratic growth is what this particular result says.



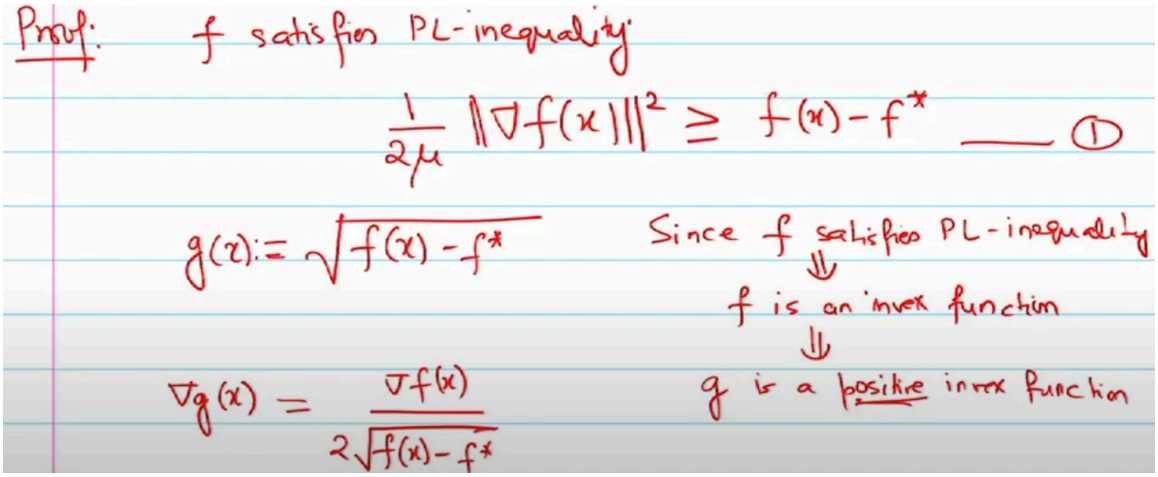
Is the statement clear? So function growth is at least quadratic if function f satisfies Peale inequality. And for strongly convex function that was in fact the key point like if you if you are working in strongly convex function even when you are close to optimal solution because of this quadratic growth or at least quadratic growth you always have non-managing gradients or you have big enough gradients to converge faster to the optimal solution which is in the case with strictly convex functions for instance right. So and the fact that most of the results that work for strongly convex functions also extend for functions which are potentially non-convex like the functions that satisfy field inequality. that may also have to do with a fact that I mean this these functions also admit some kind of quadratic type of growth.

So, let us let us try to derive this ok. So, we know that f satisfies PL inequality ok. So, what does it mean? f need not be convex. This can be an Invex function, any any Invex function like some function that satisfy PL inequality are called Invex function. So, f need not be convex. Yeah, yeah. So, if PL inequality is satisfied like I mean if a function

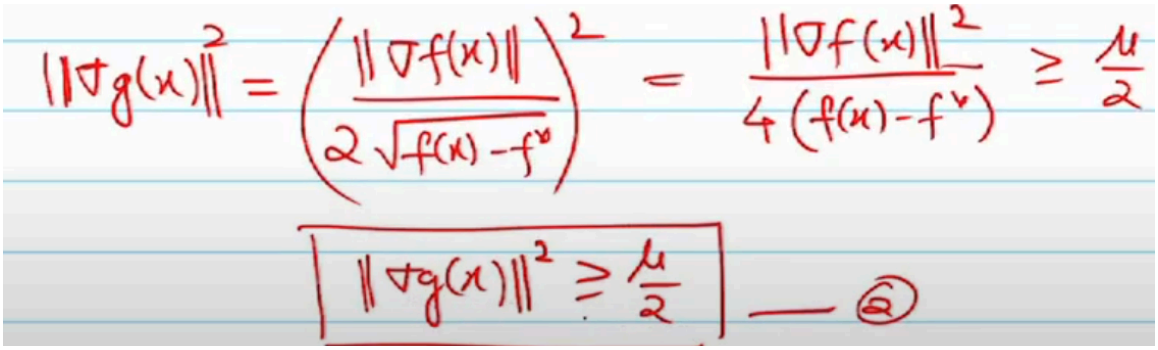
satisfies PL inequality, you only have two options. Let me let me just write this and this will also be clear from here. So, this is your PL inequality right. So, there I mean only. So, either the minimizer is unique or you have a constant function.

So, these are the only two possibilities. I mean we eliminate the trivial case. So, we assume that I mean we assume that we are working with unique minimizer all right. So, we have this as your field inequality ok. So, let us now define another function g of x.

s square root of f of x minus f star. Is g of x a valid function? Is square root well defined here? Right, right because f star f of x is always going to exceed f star. So, this is a positive function and since f satisfies PL inequality. So, this implies f is a f is an inverse function and g of x is simply defined by offsetting f of x and taking the square root. So, you also conclude from here that g is a positive invex function. I mean invex is not as important as much important as the positive part in this that g of x is always going to be positive ok.



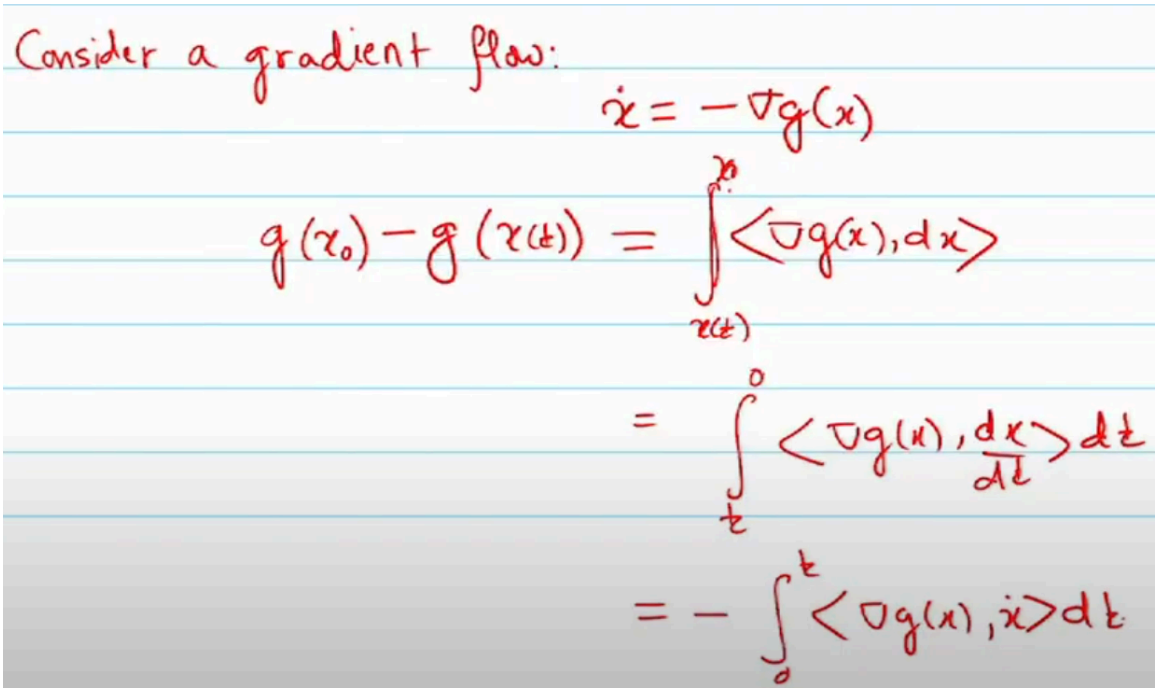
So, that is one thing that we need to keep in mind. So, what is gradient of g of x? That is going to be gradient of f of x ok. Is this clear? And what about the norm of this gradient? that is going to be the norm of this gradient of f of x. So, if I take the square of it, it will be square of this thing right, which is and here I can use PL inequality on f right, I can use this one which would be greater than equal to mu over 2 ok. The result that I have gotten is ok. So, before we proceed further with the proof, why do you think we have defined the function like this? because we want to show this quadratic growth right.



So, that means g of x is greater than equal to square root mu over 2 times x minus x star is what we want to show ok. So, that is one of the reasons why we have defined the function this way. The other thing is we somehow somewhere we had to use a PL inequality and that is used to show that this particular term is greater than equal to mu by 2 ok all right. So, now let us consider a gradient flow. Suppose we run this gradient flow x dot is negative of gradient of g of x ok.

Suppose we run this particular gradient flow. So, this would have its own x as a function of time right like if you simulate this particular flow. So, what would be g of let us say you start at x naught and at any time t g of x t. what would this be by definition ok. So, this is just in like if you integrate grain of g of I mean basically this particular term that is what you are going to get right.

We are looking at x naught g of x naught minus t right. which is same as let us call it t naught or even 0, let us say t naught is equal to 0 and does not make a lot of difference ok. So, this is nothing but minus 0 to t gradient g of x, x dot dt. is everyone with me on this. So, what is gradient x dot for this particular flow negative of gradient of g of x.



So, that means g of x naught minus g of x t is 0 to t ok. and from equation 2 or inequality 2, we have this constraint on gradient of g of x square. So, this is greater than equal to mu t over 2, ok. So, what do we know about this particular function g? g is a positive inverse function, right. So, from here we have g of x t which is less than equal to g of x naught minus mu t over that means there must exist a finite time when x of t basically becomes x star right ok, because this function the minimum value of this function is 0. So, there exists a finite time when this function this x of t becomes x star.

So, let us say that finite time is capital T since g of x t is positive. or other non-negative or in fact, I mean otherwise yeah. So, there exists some time t finite time t, t less than infinity such that g of x t is g of x star right because at x star g of g is 0 ok. So, now we have to somehow. So, we now have obviously, we are going to integrate from 0 to capital T now because So, what is the length of a path that like if let us say if I if I try to cut basically find the length of the trajectory starting at x naught and ending at x star what should that length be equal to dx.

$$g(x_0) - g(x(t)) = \int_0 \|\nabla g(x)\|^2 dt$$

$$\geq \frac{\mu t}{2}$$

$$g(x(t)) \leq g(x_0) - \frac{\mu t}{2}$$

Since $g(x(t))$ is positive, $\exists\ T < \infty$, s.t. $g(x(T)) = g(x^*)$

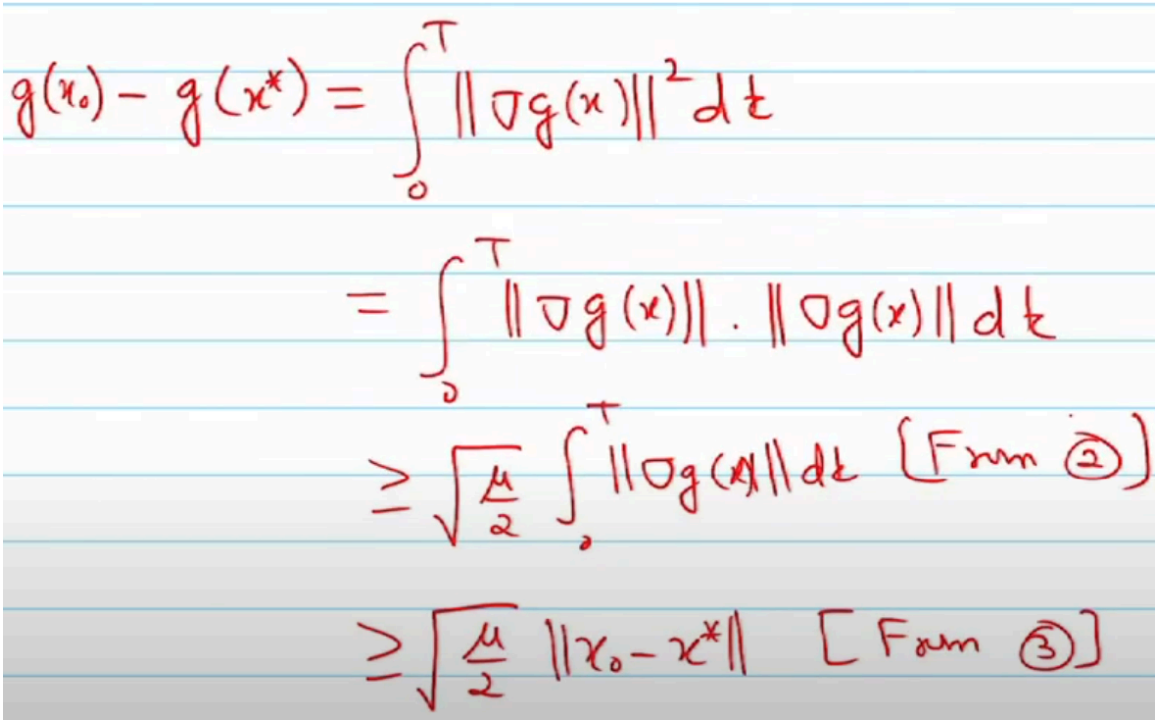
So, basically actually it is a modulus of dx right because it is a length. So, you can write this as x dot dt. from 0 to capital T ok, length of the path traced by this particular dynamical system ok. And what is x dot by definition gradient of negative gradient of g of x. and we know that this length is always going to be greater than or equal to if I join these two points by a straight line right.

So, that means x naught minus x star ok. The straight line distance between these two points is always going to be less than or equal to the total length that is traced by this particular dynamical system on length of the path raised by this particular dynamical system right, starting at x naught and ending at x star. So, let me call this as x naught x star, so that it is also clear that we are looking at the length from x. So, let us call this equation or inequality 3 ok. So, if I revisit this particular definition here or rather this term over here. So g of x naught minus g x star is essentially 0 to capital T, just rewriting this particular thing from 0 to capital T, which I can write this as 0 to capital T gradient of g of x, dt.

$$\mathcal{L}_{x_0}^{x^*} = \int_0^T \|\dot{x}\|\, dt$$

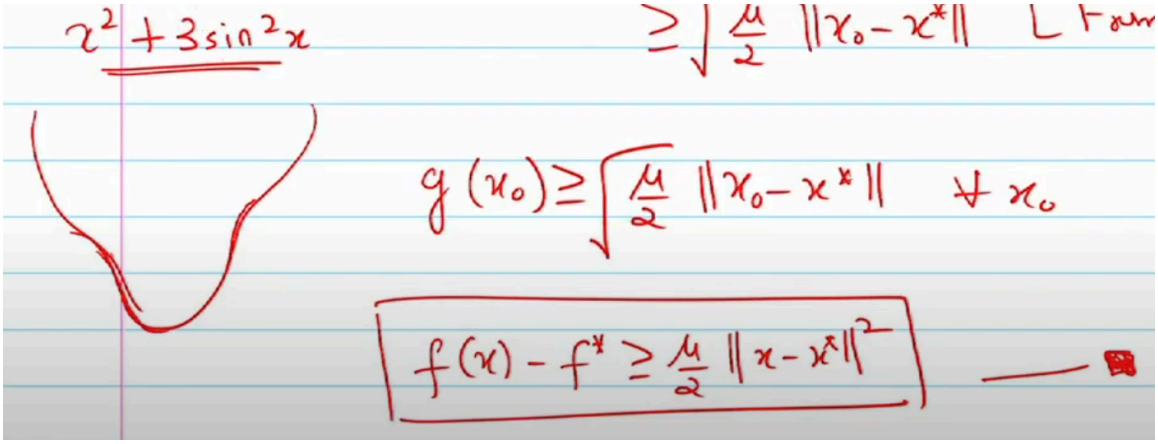
$$= \boxed{\int_0^T \|\nabla g(x)\|\, dt \geq \|x_0 - x^*\|} \quad \text{— (3)}$$

So, from this one inequality 1 or rather inequality 2 gradient of g of x is greater than equal to square root mu by 2. So, we can write this as ok and that is why we can use this inequality 3 which is. So, this is from 2. and this is greater than equal to square root mu by 2 x naught minus x star, this is from 3.



What is g of x star by the way? 0 right, g of x star is 0. So, that means g of x naught is greater than equal to square root mu by 2 x naught minus x star for every x naught right. and if I just write g of x in terms of f of x. So, this is nothing but saying that f of x minus f star is greater than equal to mu by 2 x minus x star whole square and this completes the proof ok. So, having a PL inequality with some modulus mu that means a function at least has some quadratic type of growth. So, that is why you can also accelerate optimization of functions that satisfy PL inequality.

Even though those functions may not be convex, but because they have this quadratic kind of like they are lower bounded by this quadratically growing function, you can also accelerate optimization of such functions. Is this clear? No there is no local minima right. So, invex function also have unique minimizer, but they may not be convex. So, this is an example.



So, we looked at one particular example right x square plus 3 sin square x. So, this is not strongly convex because of this in fact this is not even convex. This is the function looks almost like this, but if you if you look at the if you basically this particular function satisfies field inequality ok. Thank you.