Distributed Optimization and Machine Learning

Prof. Mayank Baranwal

Computer Science & Engineering, Electrical Engineering, Mathematics

Indian Institute of Technology Bombay

Week-5

Lecture - 17: Bregman Divergence

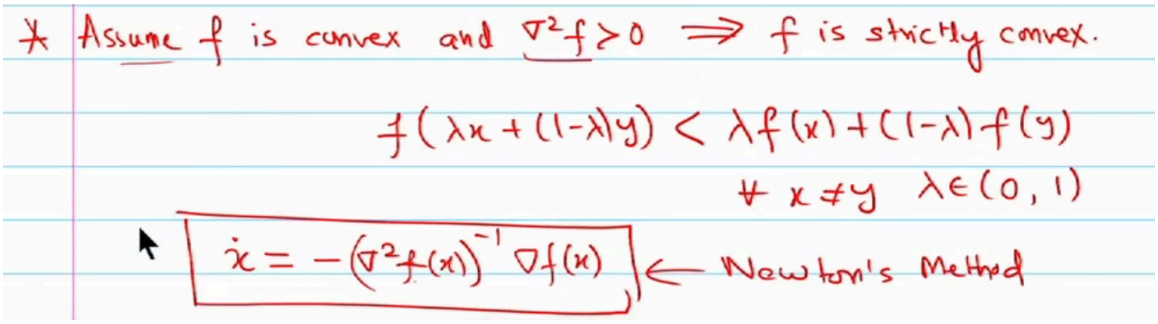
So, let us now assume a case. Assume f is convex and hessian of f is strictly positive definite. what does this imply? A straight convexity right. So, if the converse need not be true if f is strictly convex that need not imply that Hessian of f is positive definite an example would be x to the 4 right at x equal to 0 x to the 4 the Hessian is 0. but then the function is strictly convex and what is the definition like notion like sort of geometric meaning of straight convexity. Single optimizer.

Yes, single optimizer. So, the function all like basically if I choose any other value let us. So, essentially if I look at this particular thing f of lambda x plus 1 minus lambda y this is strictly less than lambda f of x. of y for every x not equal to y and lambda in open interval 0 to 1 right.

So, that means you do not have continuum of minima, you will always have a unique minima and that is going to be global minimum as well. So, if Hessian of f is positive definite, can we invert this matrix right right. So, it I mean if it is strictly positive definite or strictly negative definite you can always invert that matrix right. So, let us see what can we say about these type of functions when f is strictly convex. So, the results that we showed they work for strongly convex case right.

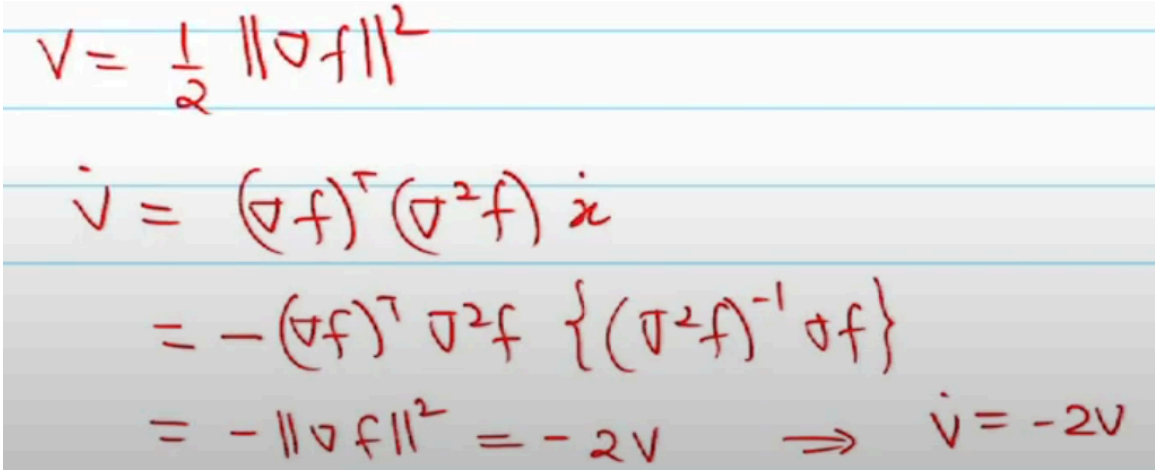
Can we guarantee something for strictly strict convexity case and the answer is yes, but then we may have to use a slightly different variant of gradient flow. So, this would be of the form. x. If any of you are familiar with Newton's method, you basically use the inverse of Hessian there. So, this is a continuous time variant of Newton's method.

So, instead of using the gradient directly, we essentially use Hessian inverse times gradient. Why did we use that? To account for the curvature. To account for the curvature, yeah. So, that is the idea. So, again when we talk about mapping optimization algorithms to dynamical systems, there are not many choices of Lyapunov function that you can work with, right.



Either you can work with f of x minus f star that is one of the Lyapunov functions, but that would only make sense when you have strong convexity because you can use peer inequality there. because eventually when you take the derivative you are going to get the gradients and if you want to substitute the gradient in terms of f of x minus f star you would have to use peer inequality that is not the case here right. So, the other kind of Lyapunov function that we can use is half norm gradient f square. So, when you consider this to be a Lyapunov function. So, v dot turns out to be gradient of f transpose h in f x dot.

And now if I substitute x dot, you can see I can basically get rid of Hessian because Hessian times Hessian inverse is identity. So, I can get rid of the Hessian and v dot turns out to be. So, let me just be more precise and x dot is basically negative, I have already looked So, Hessian times Hessian inverse that is identity and what you are left with is minus. Is this clear? So, that is why we use the Hessian inverse here because otherwise there is no easy way for us to actually get rid of this Hessian term that shows up. And for strictly convex case at least when you have Hessian to be invertible this may make sense right.



So, what do we get v dot is negative of this particular term which is minus 2 v ok. So, v dot this implies v dot is equal to minus 2 v or v converges exponentially fast. Again it does not tell us anything about the function about the optimizer x. but we know that the

Lyapunov function v converges exponentially fast that means a gradient they vanish exponentially fast ok. Alright, and that is why you see the class of functions which are strongly convex, they have specific relevance because at least for those class of functions you can guarantee accelerated convergence.

for straight convexity like x to the 4, when we are in the range minus 1 through 1, the gradients they are very shallow. So, it takes a lot of effort to accelerate the convergence and that is why in general for a strictly convex function even when the Hessian is positive definite, you cannot still guarantee convergence like exponential convergence to x star. So, you would see that most results on exponential convergence in the context of any optimization algorithm would actually end up assuming strong convexity. There are very specific cases like as I said PL inequality where you do not need to assume convexity to start with, but you still need to assume PL inequality right. So, as long as you have those kind of like sort of non vanishing gradients in picture, I mean you can still guarantee exponential convergence, but otherwise it becomes very difficult.

So, what if we end up using like let us say. Well not important, but at least I mean you know that the gradients I mean you ideally want everything to converge as fast as possible right. So, if you know something about the function maybe from by looking at the gradients then you can say something more, but you cannot in general say about the how x would converge to x star yeah. No, why not? I mean anywhere like near the optimal everything would every algorithm would slow down right because you are making tinier and tinier progresses, but at least you are making some progress. I mean you are so you are that progress.

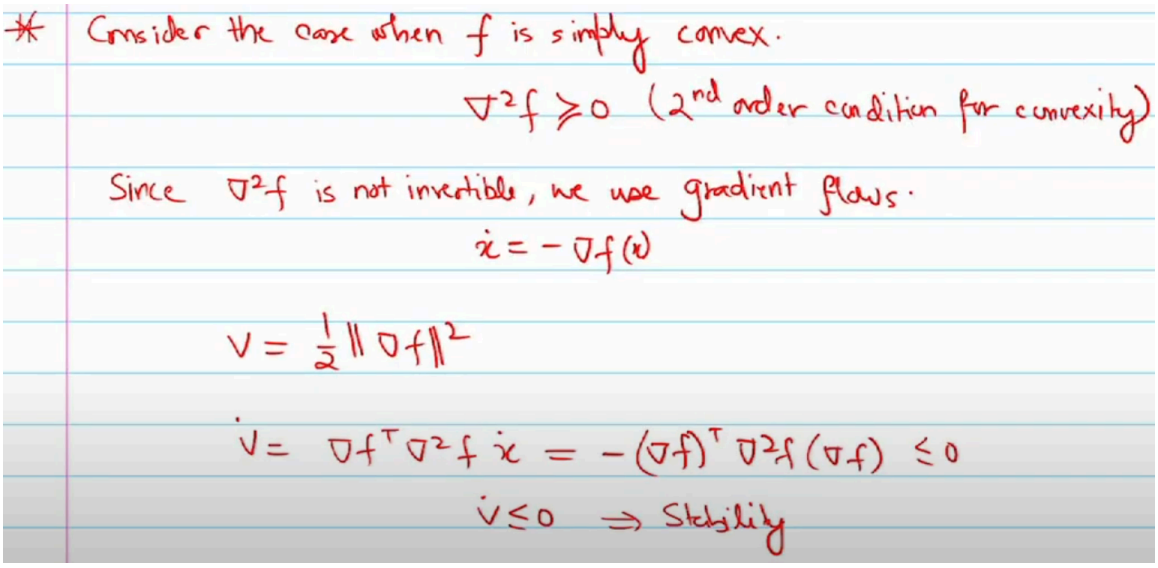
So, again like the kind of progress that you are making it can be exponential or it can be like 1 over t kind of thing which is not exponential right which is just asymptotic. So, but then if you are still making progress with larger rates that is what you desire. So, interestingly enough if we consider the case when f is simply convex. Yeah, in continuous time as I said right. In continuous time x dot is equal to negative x or x dot equal to negative 10 x both have the same equilibrium and both will have exponential convergence.

Yeah, yeah, yeah. So, there is no because in you are always decaying along the trajectory continuously right. Whereas the moment you start discretizing it, if you choose a very large learning rate, that discrete time trajectory is actually going to be very different from the continuous time trajectory. If you choose smaller step sizes, you are going to be very close to that thing. So, there is no concept of learning rate in the context of continuous time. Learning rate sort of kicks in when we talk about discretized algorithms.

So, consider the case when f is simply convex. So, what can we say about these types of functions? So, well if f is simply convex, what is the second order condition for convexity? So, h n is positive semi definite. So, that is all we can say about the, so this is the second order condition for convexity So, if let us say we end up choosing the same Lyapunov function v. So, again in this case first of all Hessian is not invertible right. So, we cannot use a Newton's flow, we will have to use as usual gradient flow.

So, let me first write down the dynamics since it is not invertible. we use gradient flows that means x dot is negative ok. So, now if I choose a Lyapunov function v to be same as half norm f square again as I said there are in much choice many choices. So, this either this or f of x minus f star would work depending on the kind of assumptions that you make on the function. So, v dot turns out to be you can be creative and try to come up with Lyapunov function which are non-intuitive, but at least intuitively these I mean these and I am going to talk about one more type of Lyapunov function would be a good sort of suitable candidates, but not more than that.

So, v dot turns out to be gradient of f transpose hessian x dot and if x dot I write in terms of gradient of f, this is transpose hessian and this thing is less than equal to 0, right because the matrix hessian of f is positive semi definite. So, all we can say is v dot is less than equal to 0. What can we conclude from this? Not even asymptotic, just stability. For asymptotic you want V dot to be strictly less than 0. So, all we can conclude is stability of equilibrium.



So, that means your iterations are not going to go off, but I mean convergence of x to x star is also not guaranteed here at least just from this analysis. So, all we can conclude about all we can conclude is your iterations your iterates x they are going to be. not like they are basically going to be bounded around your x star, but whether or not they converge to x star that at least from this we cannot argue. So, you would have to use

something called LaSalle invariance and I think that would be part of your homework, where you want to show that in fact even in scenarios where v dot is less than equal to 0, there is a way to argue that way to argue asymptotic stability and not just stability. So, we would I would leave this for now.

Any questions on this? So, again the more the assumptions you make on your functions, the better rates you can guarantee, but then that also mistakes the class of function that you can work with. It does not mean that if you use like let us say I mean if I use simple, if I choose a function f which is not let us say strictly convex, but not strongly convex. it is it may or may not be possible to guarantee. So, there are no negative results of the form that if f is strictly convex and not strongly convex you cannot guarantee exponential convergence. That kind of results I mean I mean we do not have that kind of results, but they not like I mean because all of these are anyway sufficient conditions, but if you if you have strongly convex functions and functions that all that are also else smooth those for all I mean for those functions simple gradient flow would also guarantee exponential convergence.

So, that is the sort of main summary for this. For straight convexity you can you would have to use something like if the Hessian is invertible you would have to use something like Newton's method to guarantee at least to guarantee exponential convergence of the Lyapunov function or the gradient. In this one? No, in this one it is great. So, yeah.

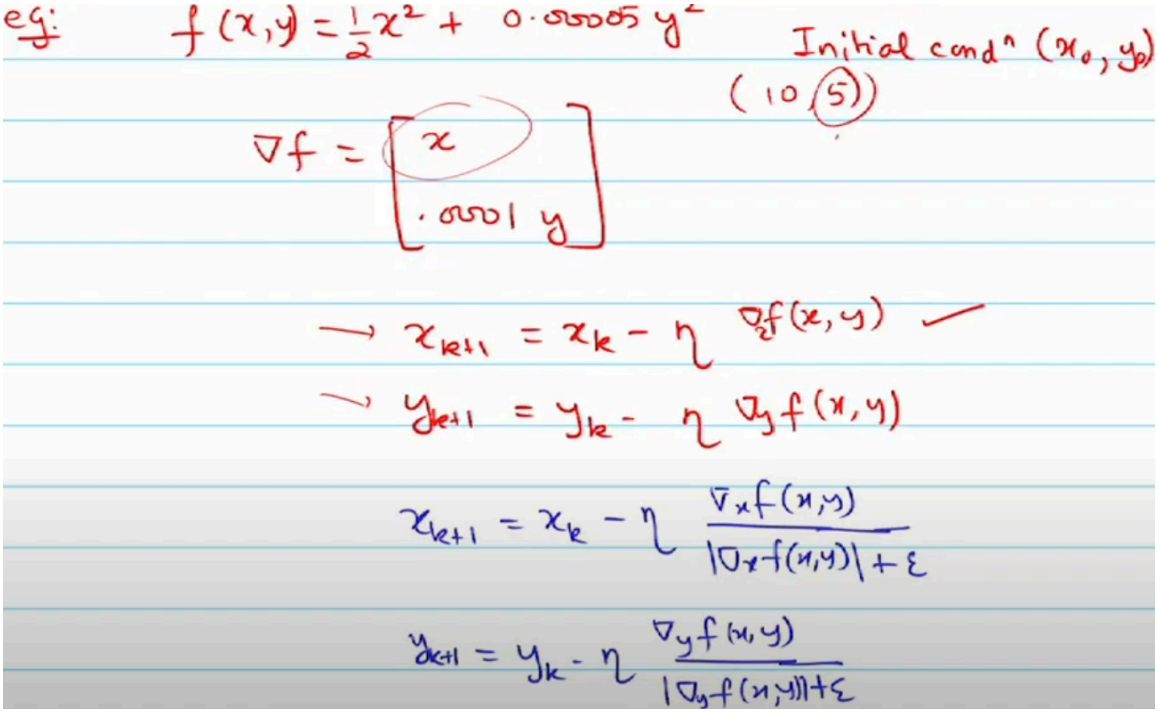
So, that is a good question. So, if in this example here, if we had used simple gradient flow, we would have gotten v dot is negative gradient of gradient f transpose hessian f gradient if right and hessian f we know that it is positive definite. So, this would we would have gotten v dot is strictly less than 0. So, that means asymptotic stability, but in order to guarantee more than asymptotic stability. So, asymptotic stability of x to x star which is anyway that is what we can derive even in this case as well, but at least we can derive the exponentials like exponential convergence of v to 0. Whereas, in the other case it would have been the asymptotic convergence of v to 0 as well.

So, that is a difference. So, if you are using the curvature information, you are likely going to accelerate the algorithm better than if you are not using the curvature information. Sometimes curvature information is useful. So, for instance, let us consider this example. It is a function of two variables x, y and let us say this is x square plus let us say half x square the 0.0000005 or maybe not something like this ok.

So, what is the gradient of this function? x the first term and the second term is 0.0001y. So, that is the gradient. Now, let us say like I want to minimize this function and I start at 10 comma 5 something like this.

That is my initial condition. So, that is your x naught comma y naught. And now I want to optimize this. So, what would be the gradient in that case? So, you can see the gradient is largely dominated by x. So, while I will be making very sort of large updates in the x direction, I am almost making no updates in the y direction ok. So, if I really look at let us say forget the continuous time version for now let us just for the sake of simplicity just look at the discrete time variant.

So, x k plus 1 is x k minus let us say yeah gradient with respect to x. and this would be your ok. So, if I look at these updates, so in the while in the x direction I will be making a huge sort of update, in the y direction that does not change by much right. And if I look at this initial condition phi, if I want to reduce this to, so what is the optimal solution here? So, if I need to reduce this 5 to 0 that will take a lot of iterations right and that is why the curvature information in some sense is useful and that Newton's kind of method is useful. So, instead of let us say using the Newton's law I do a simple hack to it.

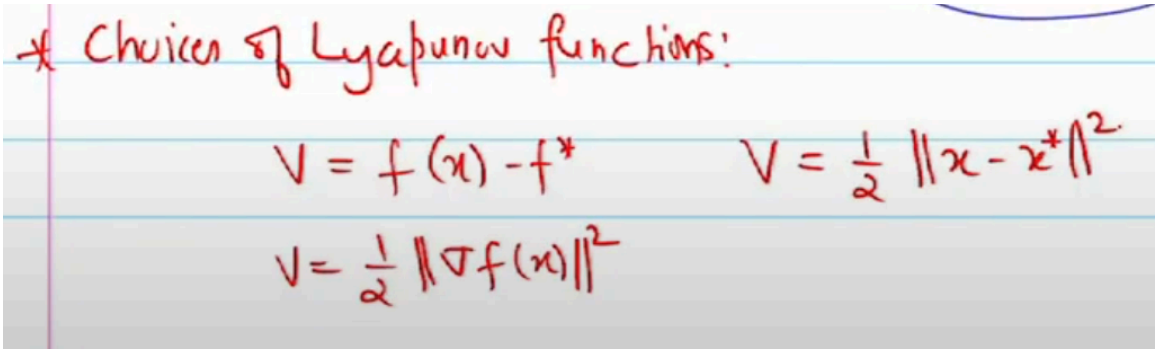


So, I design a new algorithm, I also divide this by the norm of this. and just for numerical stability I add a small epsilon to it. Now, if I look at this particular algorithm In this case, both x and y directions are not normalized right. So, you are making a similar amount of like update in both the directions, just by normalizing it with respect to the inverse of the gradient.

So, this norm of this quantity is 1. and I think what you can view this is you can particularly view this as an adaptive gradient step right. So, at each step at each step at

each iteration your step size changes and it changes with the norm of the gradient. So, I can view this as an adaptive gradient step method, but what you are really doing is you are normalizing your x and y direction and that is what your Hessian inverse in some sense is trying to do here. Because of the Hessian, you may have lost landscape which may look something like this. So while you will be making updates largely in this direction, in order to get from here all the way to this point, it will take a lot of effort.

The moment you make it Hessian inverse, you kind of change this landscape, lost landscape to this circularly looking kind of landscape. And therefore, every direction is sort of equally preferred and you can make faster updates. basically you whatever curvature is there you sort of invert the effect of it by using Hessian inverse and that is that is how you can accelerate convergence whereas in this case if I start somewhere over here it will take me a lot of iterations to get to the optimal ok. So, this gradient normalization This is quite useful and that is something that we are going to formally look at it in today's lecture. So, I was telling that there are not many choices of Lyapunov function that you can work with.



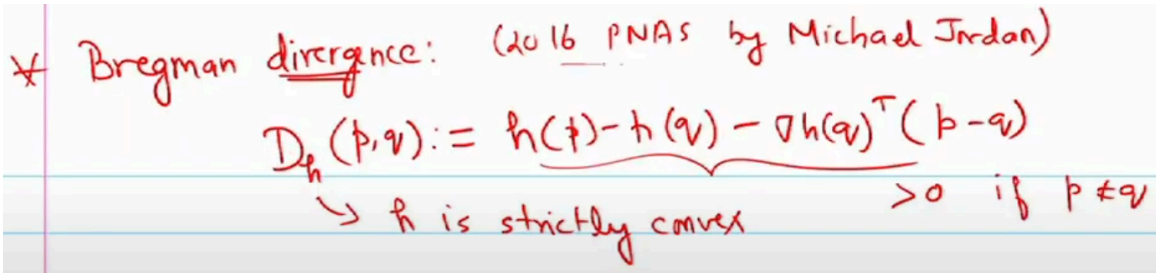
As I said f of x minus f star and half norm gradient f square. So, what are the choices of Lyapunov function? again in the we are looking at it in the context of optimization algorithms being mapped to dynamical system. So, one of them can be f of x minus f star, another that we have looked at extensively is in some text or in some cases you may also find this to be useful. So, this is also a valid choice of Lyapunov function why because only at x equal to x star this is going to be 0 everywhere else it is going to be positive right. So, these are valid Lyapunov function this in certain literature you can find use of something called Bregman divergence.

So, does anyone know what Bregman divergence is? So, Bregman divergence. So, let me write this. So, let me first define Bregman divergence. So, Bregman divergence is defined for a function h which is strictly convex. So, consider f to be like let us say you are working with f which is strongly convex or strictly convex.

So, you can also define something called Bregman divergence and what it is really saying is. So, this particular term. So, if the function is strictly convex, this particular

term is strictly greater than 0 if p not equal to p right and otherwise it is always greater than equal to 0. If h is convex, this term is always greater than equal to 0. If it is strictly convex, it is strictly greater than 0 if p is not equal to q.

And if it is strong, if h is strongly convex, this term happens to be nothing but mu over 2 times norm p minus q square, right, which is anyway the Lyapunov function like this kind of Lyapunov function or maybe this kind of Lyapunov function that you can similar to these kind of Lyapunov functions. But this is another form of another kind of Lyapunov function that one can potentially use and that is called Bregman divergence. ok, but you would require h to be strictly convex in this case, just convexity alone would not help because in that case this inequality or this particular term is just greater than equal to 0 not strictly greater than 0 right.



So, that would not help. So, it would be the function f. Yeah, a function f, your function f or something added to your function. So, again depends on, so we at least in this course we would not be looking at it too much, There is this paper by Michael Jordan which is on variational perspective and optimization, it is a 2016 paper, 2016 PNAS paper by Michael Jordan. So, Jordan actually uses looks at different like many classes of optimization algorithms in continuous time dynamic like in continuous time, but instead of using Lyapunov function like they basically they use Bregman divergences to basically obtain convergence rates. So, if you are interested you can read this particular paper, I think it is titled Variational Perspective in Optimization. I will also post this paper on course teams. Thank you.