Distributed Optimization and Machine Learning

Prof. Mayank Baranwal

Computer Science & Engineering, Electrical Engineering, Mathematics

Indian Institute of Technology Bombay

Week-10

Lecture 37: Continuous-time Distributed Optimization Algorithms

So, since throughout this course parallelly we are also we have also been looking at continuous time versions of these algorithms particularly the ones with fixed time convergence guarantees. So, let us try and develop something for fixed-time convergence as well. So, continuous So we are going to look at continuous time distributed optimization algorithm with potentially fixed time convergence guarantee. Again the setup that we have is we have n agents in the network. So each agent as its own private objective function.

So, we assume that agents can exchange information about basically on their own estimates of the optimal solution x i's and maybe something related to a gradient f i of x i ok. So, agents can exchange this kind of information. So, the problem that we are going to be looking at is it is the same problem that minimize let us say R in i 1 through capital N summation f i x i. subject to x1 equal to x 2 all the way to x n.

$$\min_{\{x_i \in \mathbb{R}^n\}} \sum_{i=1}^{N} f_i(x_i)$$

$$\text{s.t} \quad x_1 = x_2 = \cdots = x_N$$

And we are also additionally going to assume as we as we had already seen that you cannot accelerate optimization of any of any random convex function right? You need certain properties on the convex function be it strongly convex or be it strictly convex and so on. So, we assume this capital F which is capital F of x which is nothing but summation i from 1 n f i of x. So, this is mu strongly convex. You need not have all f i's to be strongly convex.

Let us say all f i's are convex and one of them just happens to be strongly convex and then we have already seen that the sum of convex and strongly convex function is also strongly convex. So, we just want the team objective function or the cumulative objective function to be strongly convex. So, this is the setting that we are going to be working with. So, before we look at the distributed protocol, let us try and see an equivalent centralized protocol. Then centralized protocol, we assume that, so there is an initialization requirement.

So, we assume that every agent, let us say magically, every agent starts at the same initial condition. This is something that I mean let us now we are just assuming for now that every agent starts at the same initial condition x1 0 is x2 0 all the way xn 0 right. Suppose this is true, every agent suppose every agent starts at the same initial condition and agents have access to global information. When I say global information what do we mean by that? Suppose all agents, they have access to this particular global information i 1 through n gradient fi of x, xi let us say, call it xi. Suppose agents have access, so this is a global information right, because agents only know their own gradients, they do not know the gradients of others.

So, they do not have knowledge about this, this is a global information and they do not have access to this global information, but for now let us suppose that agents start at the same initial condition and they have access to same global information. So, every agent is going to run this particular dynamics x i dot is let us say g and where g happens to be or g star rather let us say and g star happens to be minus So this is g star. Now if the agents start at the same initial condition and they have access to same global information. So then their g stars would also be the same starting at time t equal to 0 and they are being driven by the same input. Every agent is being driven by the same input.

$$\dot{x}_i = g^*$$

$$g^* = -\sum_{i=1}^{N} \nabla f_i(x_i) - \mathrm{sgn}^{\nu_1}\left(\sum_{i=1}^{N} \nabla f_i(x_i)\right) - \mathrm{sgn}^{\nu_2}\left(\sum_{i=1}^{N} \nabla f_i(x_i)\right)$$

So their input is same at all times. So therefore at all times x1 t is going to be x2 t is going to be x3 t and so on right. So I can very well write down the dynamics x dot is negative 1 through n gradient fi xi not xi, I mean in fact xi is basically equal to x here. ok. Did everyone follow this? So agents, so every agent start at the same initial condition, they have access to same information and using that information this control input u or g star is derived and if g star basically same for all the agents starting in the same initial condition, so everyone is simulating the same ODE right and therefore they will have the same trajectories.

$$\dot{x} = -\sum_{i=1}^{N} \nabla f_i(x) - \mathrm{sgn}^{\nu_1}\left(\sum_{i=1}^{N} \nabla f_i(x)\right) - \mathrm{sgn}^{\nu_2}\left(\sum_{i=1}^{N} \nabla f_i(x)\right)$$

So I can write x1 equal to x2 equal to x3, I mean I can represent this as x ok. So again we are looking at centralized protocol for now. So these two are big assumptions by the way in a distributed setting. I mean we won't have this agents initializing at the same exercise because I cannot know what some other agent has initialized to right. And likewise agents need not, agents to start with they won't have access to this global information.

But let's say for now you assume that this is true. So claim is centralized protocol converges in a fixed time. And the way we show this is we consider this Lyapunov function which is essentially. So, I am just giving you the proof sketch. So this is the Lyapunov function, essentially it is nothing but gradient of capital F right, it is gradient of capital F and that should be 0 at when x is at the optimal solution, optimal value otherwise it is, if I look at V dot, it is essentially summation right and let us say if if you if you happened if your individual objective functions happen to be strongly convex or so then you can further sort of use a similar argument and you will be able to get you will you will be able to show that V dot is less than equal to minus c1 V to the alpha 1 minus c2V to the alpha 2 and this way you can show that this converges in a fixed time ok.

$$V = \frac{1}{2}\left(\sum_{i=1}^{N} \nabla f_i(x)\right)^T \left(\sum_{i=1}^{N} \nabla f_i(x)\right)$$

$$\dot{V} = \sum_{i=1}^{N} \nabla f_i^{\,T} \nabla^2 f_i \, \dot{x}$$

$$\dot{V} \leq -c_1 V^{\alpha_1} - c_2 V^{\alpha_2} \quad \cancel{\text{to}} \ \text{Fixed-!}$$

So, this would imply that you have fixed time convergence. So, this fixed time this protocol this centralized protocol we can show that this converges in a fixed time, but the two restrictive assumptions that we took was that the agents initialized to the same value which is never going to be true. In fact, no one would no agent would know that what other agents have initialized to and agents would not have access to this global information. So, we need to ensure two things. One that if the agents want to access this global information for all times.

So, that means we should we should develop a fixed time parameter estimation scheme. So essentially this is the parameter that I am trying to, like every agent will try to estimate, right. And if you can show that the, your estimates would converge to the different elements or I mean this is, if f i, if let's say x is n dimensional, this is also going to be n dimensional. So you are going to be estimating an n dimensional vector and if those, basically those estimates they converge in a, they converge to the true value in a fixed time, then that means in a fixed time you can start accessing global information. So, fixed time parameter estimation scheme ok.

So, through by developing this fixed time parameter estimation scheme, we can ensure that the agents have access to this global information. Now, what about this initialization thing? Suppose I have estimated the like parameters in a fixed time. On top of it, if I run a fixed time consensus scheme on x i's. consensus on x i's what happens? There will be a time after which every agent will be in consensus right. That means if I call that to be my initial time.

So from that point onwards agents have access to this group parameter estimation. They are also in consensus. So after that if I run this protocol then every agent is essentially following a centralized protocol. and that is the kind of OD or the dynamical system that we are going to be designing. Is the idea clear? So, we know that the centralized fixed-time protocol converges in a fixed time.

We now want to show that under the assumptions where like this centralized protocol converges in a fixed time one of them is that the agents initialize at this agents are initialized at the same starting point and agents have access to this global information. So, if we can develop a fixed time parameter estimation scheme where we can estimate the parameters of this in a fixed time and on top of it we run the consensus scheme then both these conditions are going to be satisfied. So, there will be a time beyond which both these conditions are going to be true and the centralized protocol and every agent would then start following the centralized protocol. So, we say let theta i be the agent i's estimate of this global information. So, let us say this is the theta i is the agent i's estimate.

So, what these agents run is, so essentially what do we want this theta i to converge to? To some theta c which is 1 over n, basically this is what we wanted to converge to. So, we define this theta tilde i which is the error between theta i minus theta c and theta tilde i dot is theta i dot minus theta c dot. You can see that theta c dot by the way is not 0 right. What is xc? No, this is the global information that you are trying to estimate right. We want theta i to converge to this.

$$\theta_c = \frac{1}{N}\sum_{i=1}^{N} \nabla f_i(x_i)$$

$$\tilde{\theta}_i = \theta_i - \theta_c$$

$$\dot{\tilde{\theta}}_i = \dot{\theta}_i - \dot{\theta}_c$$

So, this is time varying by the way theta c. So, that is why you have theta c dot also appearing here and not just theta i dot ok. So, once you the time varying target essentially once you converge then you stay converge for all future times right. So, that is Average of the grade like yes basically average of the gradients of not average or well let us say I mean average is fine yeah you want to if you can estimate the average then n times that is essentially what this is right. So, you are going to converge to the average of the gradients of all the agents.

some of the basically average of the gradients of the all the agents right. So, the theta i dot scheme. So, what every agent runs is theta i dot is omega i plus h i where h i turns out to be d by dt of and omega i essentially is a fixed time. So, omega i turns out to be So this is your usual fixed time, basically you are trying to run a consensus on theta i's, that's why you see these terms appearing. The reason we add this particular term is, so if you look at the dynamics of theta i dot right, it also has d by dt of gradient of f i and every agent would like have their own h j and so on.

$$\dot{\theta}_i = \omega_i + h_i \qquad h_i = \frac{d}{dt}\nabla f_i(x_i)$$

$$\omega_i = \sum_{j\in N_i}\Big[p\,\mathrm{sign}(\theta_j-\theta_i) + \gamma\,\mathrm{sgn}^{\nu_1}(\theta_j-\theta_i) + \delta\,\mathrm{sgn}^{\nu_2}(\theta_j-\theta_i)\Big]$$

So ideally had there been no h i, this would have been a consensus on theta i, but you also want to track this time varying gradient right. and that is why you also need this particular information to be included. And if you look at, so essentially let's say if you have this particular omega i, so this gives you, if you simulate this particular dynamics you get your theta i. So what is the g i or the equivalent descent like basically distributed protocol that every agent would be applying? Remember g star was this particular term right, which is n times theta i. So every agent would be running minus n times theta i plus signum this this not signum, but the.

So, every agent would be running this, because every like agents do not know what is what is what this global information is. So, they would instead of running g i g star they would be running this right, where theta i would eventually converge to the global information ok. The other thing that agents need to need to ensure is that consensus on x i. So, individual dynamics of this agent would be g i plus u i, u tilde i let us say where this u tilde i happens to be a consensus on signum let us say nu 1 x i minus x j. So this is the final thing that agents would be running, xi dot is gi plus this u tilde i, u tilde i is the consensus step on xi, gi is essentially this particular protocol where theta i is basically obtained by running this.

So this is what every agent would do. Every agent would simulate this and eventually, you can show that, so essentially I can point you to the particular verb, but the result says that if So theta i would converge to theta c in a fixed time if the difference between h i t minus h j t because these are time varying quantities now that is less than equal to rho for every t greater than equal to 0 where ok. So let us recap a bit. So remember in omega i, I mean ideally I would have achieved consensus without this particular term. But the reason we add this term is we want to subsume the effect of this disturbance.

So think of it as a disturbance on your consensus term, right? And you want to get rid of, subsume this term in, essentially you would want to get rid of this disturbance by using a large gain. So this, think of p as a control gain. So if the disturbance is small, so what you are saying is the disturbance is bounded. Then if I choose p to be large enough, my controller the controller the control gain that I am applying if that is large enough then I can subsume the effect of this disturbance. This is the similar to one of the centralized remember in one of the lectures we had this additive epsilon kind of disturbance and we try to get rid of this by adding an additional term that is exactly what we are trying to do.

$$g_i = -\left(N\theta_i + \text{sgn}^{\nu_1}(N\theta_i) + \text{sgn}^{\nu_2}(N\theta_i)\right)$$

$$\boxed{\dot{x}_i = g_i + \tilde{u}_i}$$

$$\dot{\tilde{u}}_i = -\text{sgn}^{\nu_1}(x_i - x_j) - \text{sgn}^{\nu_2}(x_i - x_j)$$

So, every agent runs this dynamics x i dot is gi plus u tilde i, where u tilde i is essentially is a consensus kind of dynamics, g i is defined like this which is similar to centralized protocol, but without the global information and in order to estimate global information in a fixed time theta is updated like this. So, once if every agent runs this you can show that in a fixed time every agent would converge to the optimal solution in a fixed time. So, that this is the equivalent fixed time protocol for the distributed optimization. So, any questions on this? So that's all I wanted to cover in today's lecture. So essentially gave a flavor of different types of distributed optimization algorithms that exist.

<u>Theorem</u>: $\theta_i \to \theta_c$ if $\|h_i(t) - h_j(t)\| \le \rho \quad \forall\, t \ge 0$

where $p > \left(\frac{N-1}{2}\right)\rho$.

As I said the literature is pretty vast. So I mean we have already looked at, in the discrete setting we have only looked at DGD but there are multiple such algorithms. I mean that in fact every year like in every control conferences you would see multiple of papers just on this distributed optimization. I mean there are more advanced algorithm, there is PI consensus, there is PI algorithm, PI consensus algorithm and so on. There is also accelerated Nesterov, distributed Nesterov gradient descent.

So, there are multiple such algorithms. And then we also looked at centralized protocol which has a fixed time convergence guarantees. So, I hope like with these algorithms in like if you understand these algorithms fine you should now be able to solve distributed optimization problem using an algorithm of your choice and you should be able to read papers and follow the literature on this. So, in the next class we are going to slightly shift the gears towards federated learning. So, where well not federated learning to a great deal, but we are also going to look at issues like what happens when you have like the communication kind of issues and when you're trying to train very large neural networks.

So it's a large scale optimization problem. So you may have issues sharing information with your neighbors, right? So there may be issues with regards to synchronicity that not every agent updates us values at the same time. So what happens when you have a synchronicity? So in the next few lectures, we are going to be focusing more on that. So we are going to look at parameter server approach, then ring all reduce algorithm and so on. So that would be. topics of discussion in the next few lectures.