STOCHASTIC APPROXIMATION: THEORY AND APPLICATIONS

Dr. Gugan Thope

Department of Computer Science and Engineering

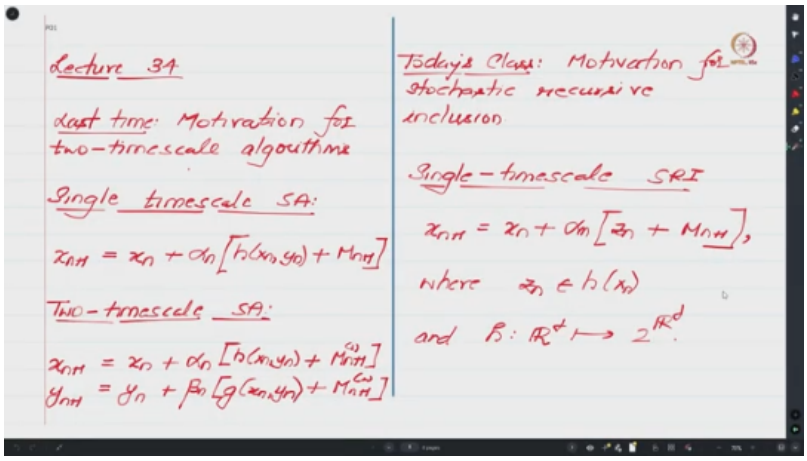
Indian Institute of Science, Bangalore

Week 9

Lecture 34

Motivation for Stochastic Recursive Inclusion in Distributed Mean Estimation

Hello and Namaste everyone. Welcome to lecture 34 of this NPTEL course on Stochastic Approximation. If you recall, during this week and the next week, we are going to discuss the motivations and utility of some variants of Stochastic Approximation Algorithms, namely, two-timescale Stochastic Approximation Algorithm and Stochastic Recursive Inclusion. Right. So, in particular, in the previous two classes, we sort of set the stage that motivated the use of two-timescale stochastic approximation algorithms.

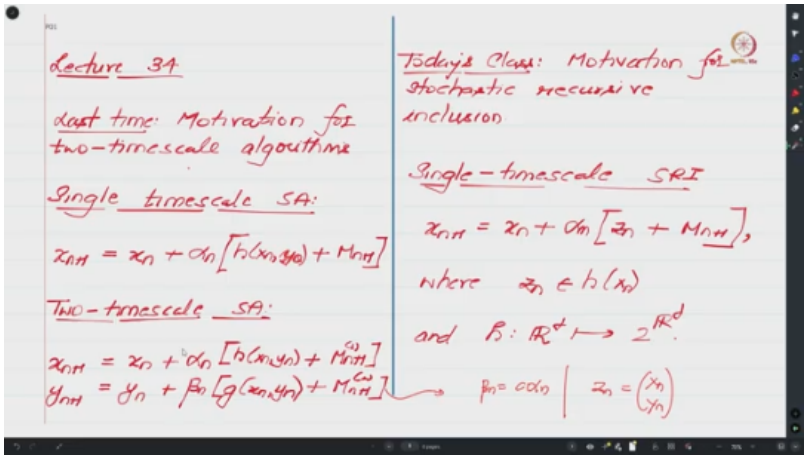


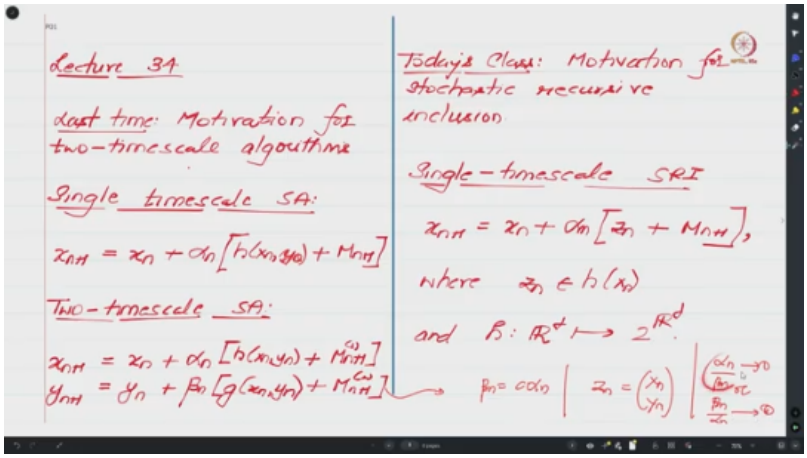
So, just to give you a quick recap, um, Recall that a single-timescale stochastic approximation algorithm is an update rule of the form. So, I should perhaps remove this: it is an update rule of the form xn plus 1 equals xn plus alpha n times h of xn plus mn plus 1. So, you have like one variable which you are updating using one step-size sequence. So, such algorithms are referred to as single-timescale stochastic approximation.

On the other hand, the general form for a two-timescale stochastic approximation is as follows. Here, you sort of have two different variables that you are updating, and the first variable update rule could depend on both Xn and Yn, and similarly, Yn's update rule could depend on Xn and Yn, right. Now, you know, this algorithm itself could be viewed as one-timescale when alpha n and beta n are of the same order. By that, I mean suppose your alpha n and beta n had the relation that, you know, Your beta n is, let us say, c times alpha n, right?

So, then you could, you know, club these two variables together. That is, Zn equals, you know, Xn stacked over Yn, right? And then you can see that you can write an update rule for Zn based on these two update rules, and that will have a similar form like this. So, whenever alpha n and beta n are of the same order, then even though we may write it this way, that algorithm can still be viewed as a single time-scale algorithm. So, what makes

Such an algorithm two time-scale is the fact that alpha n and beta n, right, they are of different orders, right. In particular, the different choice of orders will imply that either your alpha n over beta n goes to 0 or beta n over alpha n goes to 0, okay. So, either this goes to 0 or, you know, the inverse ratio actually goes to 0. So, in this case, your alpha n decays much faster than beta n. On the other hand, in this scenario, your beta n decays much faster than alpha n. So, in the previous class, we said that sometimes the function that we are interested in, whose zero we are interested in finding, may not have the linearity property, which implies that the expectation and certain things need not get reversed, in which case we have to work with two time-scale algorithms.





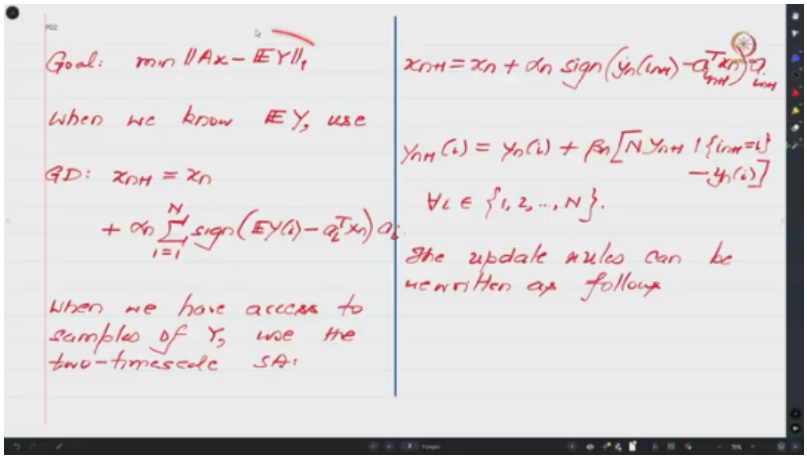
Right, and in today's class, what we will do is we will come up with a motivation for working with stochastic recursive inclusions. Is this okay? In particular, you know, to begin with, let us first look at an update rule for single time-scale stochastic recursive inclusion, and the same idea can also be extended to discuss two time-scale stochastic recursive inclusion. So, you know, the form for single time-scale stochastic recursive inclusion is given over here. So, observe that here the update rule has the form Xn plus 1 equals Xn plus some step size alpha n times Zn plus Mn plus 1.

So, notice that unlike a single time-scale stochastic approximation where we had H of Xn, we do not have H of Xn here. Instead, right, we have Zn, and we will presume that this function H, right, is actually set-valued. Set-valued meaning for every X that comes as input to H, the output H of X is actually a subset of, you know, Rd. So, this notation over here represents the power set. So, you know, which basically implies that for every input, right,

X in Rd, H of X is actually a subset of Rd, right? So, in that sense, you know, your stochastic recursive inclusion allows us to study set-valued dynamics, right? And in today's class, we will somehow try to understand why we would want to study set-valued dynamics in a very specific context, right? So, coming back to this update rule, Okay, here instead of having H of Xn, which would have been a vector in a single time-scale stochastic approximation, here little h of Xn is actually a set-valued map.

So, which means Zn is some d-dimensional vector in H of Xn. Is this okay? So, whenever you have such an update rule, we will refer to it as a single time-scale stochastic recursive

inclusion. Right? So, you know, coming back to this problem that we were interested in solving, that is, we wanted to minimize this objective function.



Keep in mind that you have the L1 norm over here, right. So, I will try to use this objective function again to illustrate why we may want to minimize work with set value dynamics, right. So, let us recall, you know, the design of our algorithm, and from there on, try to see why we may want to work with set value dynamics, right. So, recall that, you know, if you knew the expected value of y, right, then, you know, and there were

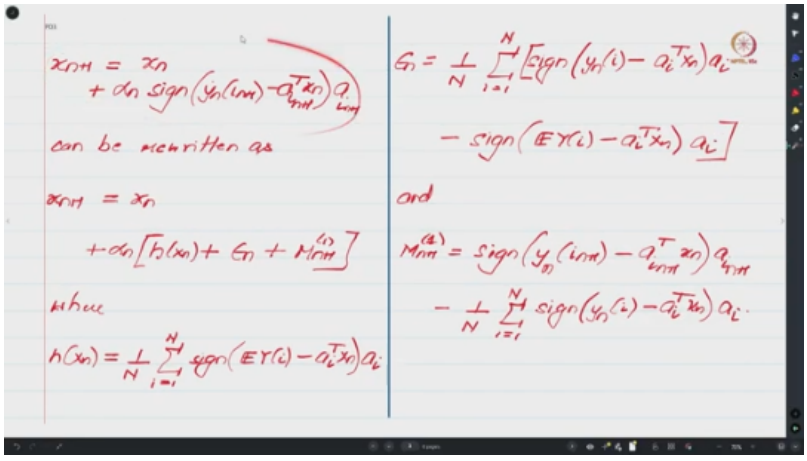
then we could use this gradient descent algorithm. However, often in practice, we do not know the expected value of Y. Instead, we have access to samples of Y, and using these samples of Y, we want to nevertheless solve this optimization problem. So, when there are no adversaries, this is the update rule that we came up with. For Xn plus 1, we came up with this update rule, and here notice that we have little yn instead of directly plugging in calligraphic yn. Calligraphic yn, recall, is an appropriate sample of—or I should say, calligraphic yn plus 1 is a sample of

a suitable coordinate of the vector y. In particular, in this case, it is the sample of y i n plus 1. So, we do not plug in directly this calligraphic yn plus 1 because, as I said, the expectation of a sign is not the sign of expectation, which holds because the sign function is not—you know, the expectation and sign operators do not interchange with each other. So, because of that reason, instead of plugging in calligraphic yn plus 1, we actually plug in little yn plus 1. And the idea is that, you know, as n becomes larger, right, this yn will

start getting closer and closer to the expected value of y, right, in which case, at least in an asymptotic sense, this update rule will start looking like this. Okay, so that is the hope.

So, you can see that in this scenario, okay, estimating your yn on one time scale and using that estimate and plugging it here sort of makes sense. Because if we had directly plugged in here, we would not have managed to have such an update rule. However, by going via this two time scale route, at least in an asymptotic sense, we are ensuring that this update rule starts mimicking what we would have in the gradient descent case. Where recall that gradient descent is the noiseless version of what we want to do. Now, in some sense I mean please keep in mind that there are no adversaries so far and I sort of used this recap to recall for you the motivation for working with two time scale algorithms.

And what I will now do is I will sort of tell you how we can rewrite this update rules that sort of enables our analysis. So, I hope you agree that this update rule over here, which is the Xn update rule can be written in this form. So, here you can see you have H of Xn. So, I think I should have perhaps used capital H here because I am going to use capital H for the set value dynamics henceforth. So, presume that capital H is a set value dynamics here or you know as a set valued map and this is basically capital H goes from rd to 2rd and here when I write little h I will continue to imply that it is a map from rd to rd that is when I give x as input to h the output h of x is also d dimensional in nature.



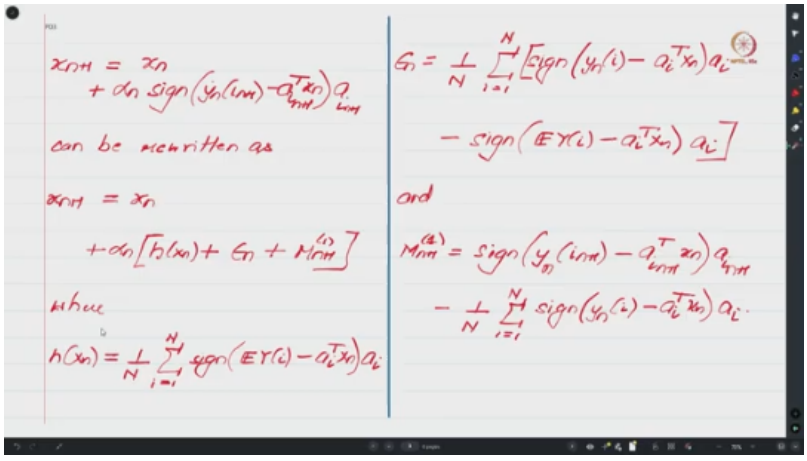
So coming back to our goal, we want to rewrite it in a form that sort of enables our analysis. So I hope you agree that this update rule can be written in this form where H is given in this fashion. So notice that here what we have done is we have first of all taken

the conditional expectation of this expression with respect to IN plus 1. So that is why we have 1 over N. Because recall that we had presumed this i n plus 1 is being sampled uniformly randomly.

On top of that, wherever I have y n, I have replaced it with the expected value of y of i. So, in some sense, this expression matches the gradient of this objective function that we are interested in. Is this okay? And so now, epsilon n is basically whatever we have over here. So, whatever we have over here, subtract it with what we really have. So, yni is basically what we work with.

So, yni is this expression, and this expression we are going to compare against this idealized expression. So, this and this cancel off. So, what we are left with is this expression and this Mn plus 1, 1. So, the 1 here denotes that this is the noise sequence associated with this Xn update. So, this expression is basically what we have in the update rule in reality minus whatever is the expression over here.

So, in this sense, you can see that if you add h of xn, epsilon n, and mn plus 1 of 1, you would indeed get back this expression over here. So, let us try to understand why I have expressed this update rule in this form. So, this expression, whatever you see over here, is like an idealized update. It is the update that we would have used if we had known the expected value of y. So, this is the update that we would have used. Unfortunately, we don't know this.



Instead, what we have access to is yn. So you can think of this term as as a perturbation term that is being added, right? So this is like a perturbation or a bias that is being added because of the fact that we don't know expected value of y. Instead, we have access only to little yn. But nevertheless, since yn, you know, is hope to go to expected value of y for every i, recall that we don't have adversaries here.

Hence, the hope is that for every i, this expression will go to expected value of y of i. So, in that case you know the sign of this expression and so I think I should be careful here this is the sign of this expression and similarly here also it is the sign of this expression and maybe I should have a big bracket over here. So, you know, sorry, sorry, I think I made a mistake. What was there before is correct. Sorry about that.

So, this is correct. So, this is a scalar. So, we look at the sign of this expression. This is a scalar. We look at the sign of this expression and these ai's are vector.

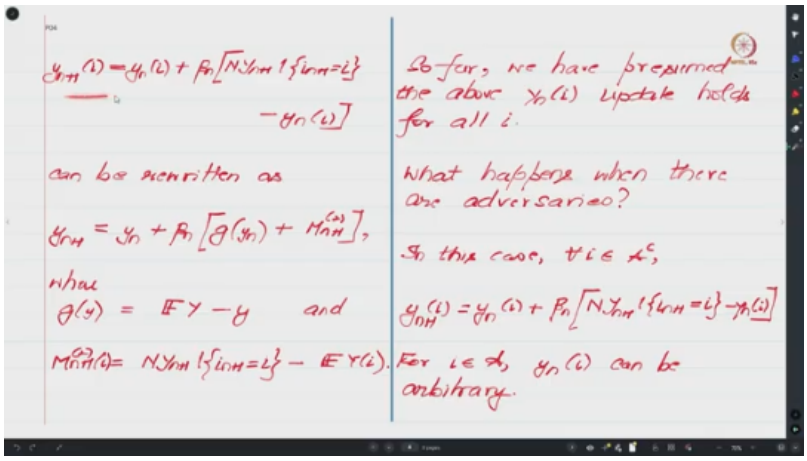
I apologize for that mistake. So, since Yn goes to expected value of Y because we are in the situation where there are no adversaries, one would hope that the sign of this expression also matches the sign of this expression. So, I am saying one should also hope. Because the sine expression or the sine function is not a continuous function. So, sine is a discontinuous function.

So, even though the argument to the sine functions one would hope and one can probably show that this expression converges to this expression. It is not obvious that the sine of this expression should converge to the sine of this expression. But nevertheless... you know I mean we will soon see how to address this and we will actually you know invoke the fact that this Yn actually goes to expected value of Y to tackle this gap and you know we will show that in some sense the effect of epsilon n in the behavior of Xn right is negligible. So, you know I will sort of quantify what this some senses and so on and so forth.

And Mn plus 1 is basically what we had here minus its conditional expectation with regards to the information that we have at time n. And in that sense one can see that this expression and this expression if you take its conditional expectation that conditional expectation will be 0 thereby telling us that this Mn plus 1 is actually a martingale

difference sequence. So, this is a nice term. This is in some sense an additional perturbation term that we have not encountered so far. And this H of Xn is what we have over here. In some sense, this is like the idealized update we would like to have.

And similarly we can rewrite this yn update. So recall that the yn update has this form. So we can rewrite this in the form yn plus 1 equals yn plus beta n times g of yn plus mn plus 1 superscript 2. So the superscript 2 means that this is the noise in the update rule for yn. Where g of y is basically expected value of y minus y and this is a vector whose ith coordinate equals whatever we have over here which is this minus the ith coordinate of this expression in particular at yn.



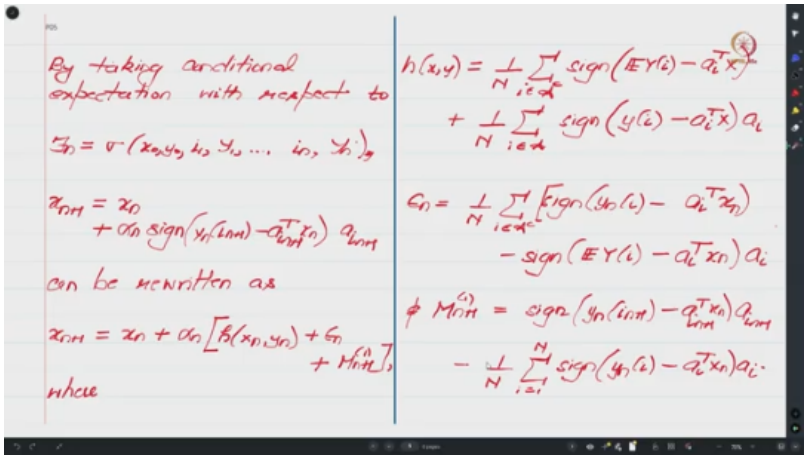
So if you see this expression and this expression will cancel off and we would end up with this. Again, one can see that if you take the conditional expectation of this with respect to the information that you have at time n, right, and invoking the fact that both calligraphic yn plus 1 and in plus 1 are generated independently and also independently of each other. Using that fact, one can see that if you take the conditional expectation of this, then this would again be 0, again telling us that this is a martingale difference sequence. So, in other words your Yn plus 1 has this update rule where again one can view this G as providing an idealized direction. So, idealized direction is basically determined by this expression over here.

So, so far we have presumed that there are no adversaries which implies that whatever is the update rule that we have mentioned over here is what is being done at all values of I. Now, we want to go to the scenario that we were studying which is when a small subset

of these I values are adversarial in nature which means that Let us, you know, denote these adversarial indices by calligraphy K. So, here I have the complement of this. So, for any i in a complement, which means that this index i is now not adversarial. So, for such non-adversarial or honest, you know, coordinates, indeed the update rule matches what we have over here, right?

So, the update rule matches what we have over here. However, if I is adversarial in nature, then this expression can be arbitrary because this need not follow this update rule, as the adversary need not do what we are expecting it to do. However, the honest workers will indeed do what we are expecting them to do. So, the difference between this and this is that in the honest case, we presume that this update rule that we have over here is followed for all I. Whereas, in the case where we have adversaries, this update rule is only followed by honest workers, and for adversarial coordinates, we will presume that this Yn of I is some arbitrary value.

Is this okay? So now let us see how we can rewrite this update rule. So now we have to keep in mind that your Yn for adversarial coordinates need not go to the expected value of Y. So what we do now is, while we work with the same update rule as in the non-adversarial case, For our analysis, we rewrite it in a certain form.



So, let us try to understand what that form is. So, this is the update rule that we had before, but now we write it as h of xn, yn plus epsilon n plus mn plus 1 of 1, right? So here, notice that instead of having H of Xn—observe that here we had H of Xn in the

non-adversarial case. Now I am presuming that this little h is actually taking two quantities as input. One is Xn and the other is Yn.

right and what is this function this function right again I would like to highlight that when I begin this lecture I had you know said that this little h is set valued but you know I take back that thing so this little h you know basically takes as input to d-dimensional vectors and spits out a d-dimensional vector and for that set valued map we will use the notation of capital H So, what is little h of x, y here? Well, here observe that I split the sum into two parts. One is I in A complement where you call A complement means it is the set of indices which are honest or non-adversarial in nature. On the other hand, calligraphic A means adversarial.

So since your honest workers are indeed doing what we expect them to do or the honest coordinates. So I am using the word coordinates and workers interchangeably. So the honest coordinates are indeed doing what we expect them to do. Hence here we have written an expression similar to what we had in the all honest workers case. So observe that here we have written the idealized update direction.

But this is only for I in A complement. For I in A, we basically write this sign expression based on whatever is the input Y that is provided. So Y is D dimensional. You look at I in calligraphic A and for all those I's, you write basically this sign expression. So, in some sense, the adversaries do not affect this part of your update direction, but they can control this part of the update direction.

Right? So, this is the definition of H of XY. Now, this epsilon n. As before it is the difference between what is the idealized update direction when we you know like for the case where we hypothetically know this expected value of Y and the realistic scenario where we know Yn. So, epsilon n is the difference between these two expressions.

This matches what we had in the previous noiseless scenario also. However, the key difference is previously this I went from 1 to capital N. Here, on the other hand, we restrict the values of I to lie in calligraphic A complement. So, that is what is this epsilon n and I think I should put a bracket appropriately. So, the bracket should be something

like this. So again, since I is in A complement, this is like a measurement obtained at an honest coordinate.

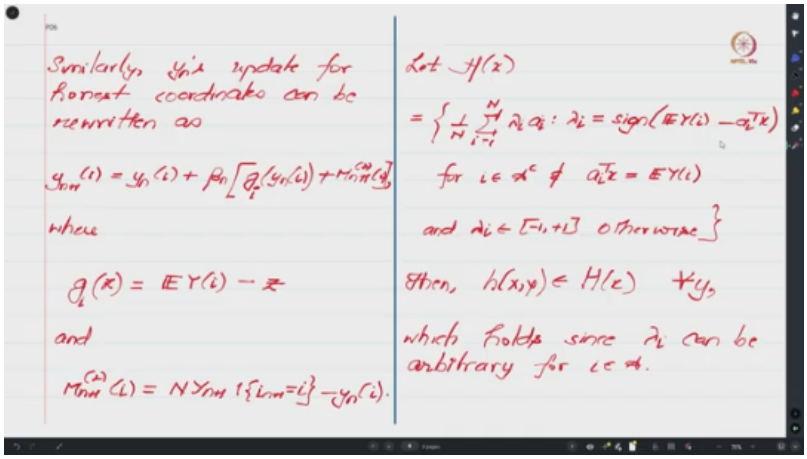
So this one would expect will go to this and while the sine function is in some sense discontinuous, we will nevertheless show that because Yn goes to expected value of Y, the effect of this epsilon n in some sense is negligible. So I will talk about those things later on. And this noise term is exactly as before. Now, if you take the conditional expectation of this with respect to the information that you have at time n, then the only thing that is random here is i n plus 1 and hence if you take the conditional expectation of this minus this expression, one can see that indeed its conditional expectation will be 0 and hence this will continue to be your martingale difference noise. So just to summarize, the difference between the adversarial case and the non-adversarial case is what we view as the idealized update direction.

So in the noiseless case, our idealized update direction was given by this expression. On the other hand, in the case where we have a small subset of adversaries, the idealized update direction is what is given over here. So, again I would like to highlight that the effect of adversaries in some sense is captured over here and let us try to ponder over it for a few seconds. So, observe that this is the quantity that the adversary can provide. But since this quantity is sitting inside the sign expression, what the adversary is

at worst can do is flip the sign of this expression meaning if we had expected value of y of i here we would have you know gotten the right sign right. So right sign is basically whatever is there in the true gradient expression right. However since we do not have expected value of y and this coordinate is controlled by an adversary right the adversary could flip this sign. So in some sense because of this presence of sign, the impact of the adversary is restricted in the sense that In the worst case scenario, the adversary can flip the sign.

Now, I would like the reader to think about what would have happened if instead of working with L1 minimization, we would have worked with L2 minimization of this AX minus expected value of Y's update rule. So, then I would like to ask you what do you think would have been the impact of this adversary. And similarly, as in the

non-adversarial case, Yn's update can be written in this form. However, one needs to now be careful. In the no-adversary case, Yn plus 1 had the following update rule.



In other words, if you look at the ith coordinate of this, Okay, you know there exists an update rule where you know you looked at the ith coordinate of this and that update rule held for every i. However, in the case where we have adversaries, right, this update rule holds true only for the coordinates i which are non-adversarial in nature, right. And for all other i's, these yn i's can be arbitrary. Is this okay? Alright.

Now, the question is, okay, you know, we have done some algebra, rewritten things in certain form. Where does the stochastic recursive inclusion or the set value dynamics kick in, right? So, towards that, we are going to define a set valued map. So, you know, I am using this capital H notation to come up with this set valued map. So, what is this map?

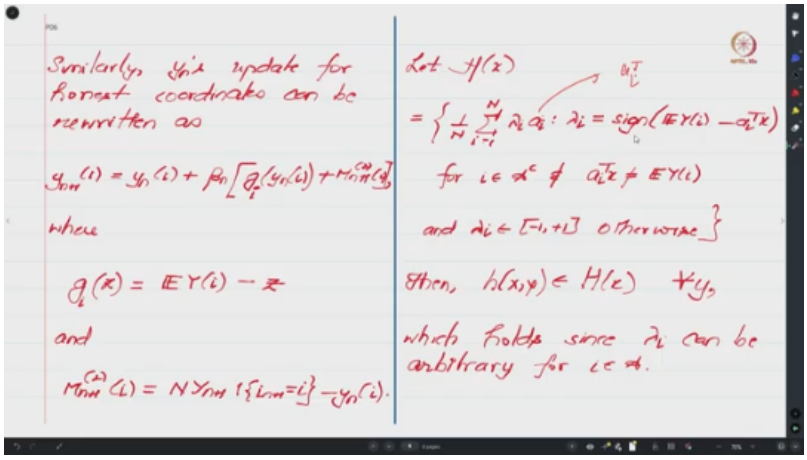
This capital H of X is basically containing all vectors of this form. So, notice that when I give X as input to this capital H function, the output of this function is not one element or not one vector in RD. Instead, it is a collection of vectors of this form. So what is the special property of this collection? Well it is you know of this form meaning it is some you know convex combination of AIs where so I should not say convex combination sorry about that some combination of these vectors AIs.

Where these lambda i's are scalars and ai's are your vectors. So recall that ai transpose was the i'th row of your matrix A. So this ai that we have over here is the transpose of

this vector. So this was a row vector and ai is a column vector. So, we have some combination of these A1 to An in particular it is a linear combination. So, we have a linear combination where lambda 1 to lambda n are scalars meaning they are some numbers and these lambda 1 to lambda n should satisfy some properties in particular

Whenever I is not in the set of adversarial coordinates, this A I transpose, so let me state that again. For every I that is not in the set of adversaries and A I transpose X is not equals expected value of y of i, right? So, if they were equal, then the argument to this sine function would be 0. So, let us try to avoid that case for the time being, right?

So, for every i that is not in the set of, you know, adversaries and a i transpose x not equals to expected value of i, right? For such an I we require that this lambda I equal the sign of this expression. Now if this quantity is not 0 which holds because of this insistence that we have. So this either will be positive or negative. So keep in mind that this is a scalar and AI transpose X is also a scalar.



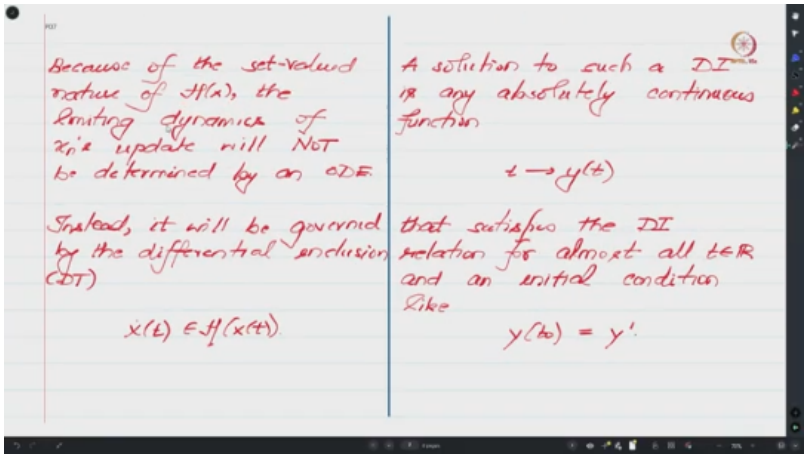
So this is a scalar. So since this is not 0, either it will be positive or negative. If it is positive, the sign expression will output plus 1 and if this argument is negative, the output of the sign function will be minus 1. So this lambda i should either be plus 1 or minus 1 depending on this expression for every i that is not an adversary and ai transpose x not equals expected value of i. And for every other i, this lambda i can be any number between minus 1 to plus 1. Is this okay?

So in other words... For the i's which satisfy both these conditions, the choice of lambda i is unique. It has to be this. But for every other i, your lambda i can be any number between minus 1 and plus 1. So, you know, by choosing different values of lambda i's for such i's which don't satisfy this condition, I hope you can see that we can cook up multiple such vectors here, right?

And because of these multiple vectors, we would end up with a set over here. And consequently one can see that this H X, Y which is one vector. So this is an element in RD whereas this is a subset of RD. So one can see that for whatever value of X and Y. we have right from the definition of H of X, Y that is given over here right one can see that you know this expression or this output will actually lie in capital H of X right so that is something that can be verified right and you know this actually holds because

you know we uh you know as i told you whatever be the value of y right this can be some arbitrary sign expression depending on the value of y but whatever be this value right now this sine function uh you know can only take values between minus 1 and plus 1 in particular if this quantity is negative it will take the value minus 1 and if this quantity is positive it takes the value plus 1 and if this is 0 then the sign expression can take any value. Is this okay? So keeping that in mind and since we allow lambda i to be any value between minus 1 to plus 1, one can check that h of xy belongs to capital H of x. Is this okay?

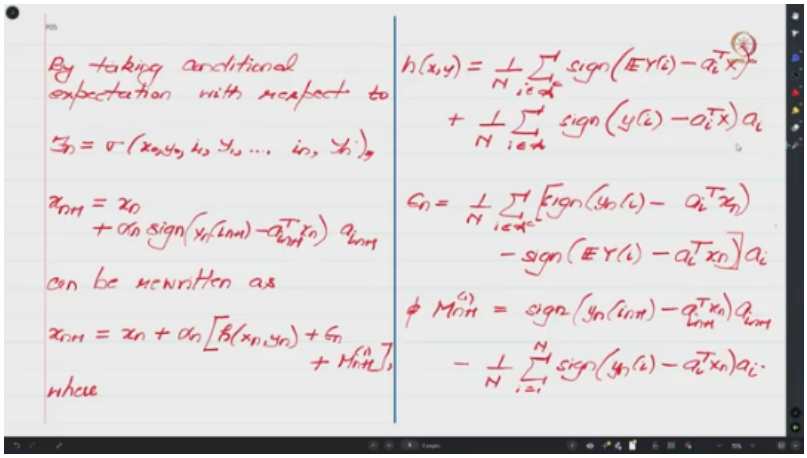
So because of the set valued nature of this capital H of X, the dynamics of Xn is not determined by a vector valued function. Instead, it is determined by this set valued function. And consequently, the limiting dynamics of Xn's update will no longer be determined by a differential equation. Instead, it will be governed by a set-valued generalization of a differential equation, which is a differential inclusion. And often, I will abbreviate differential inclusion by these two letters DI.



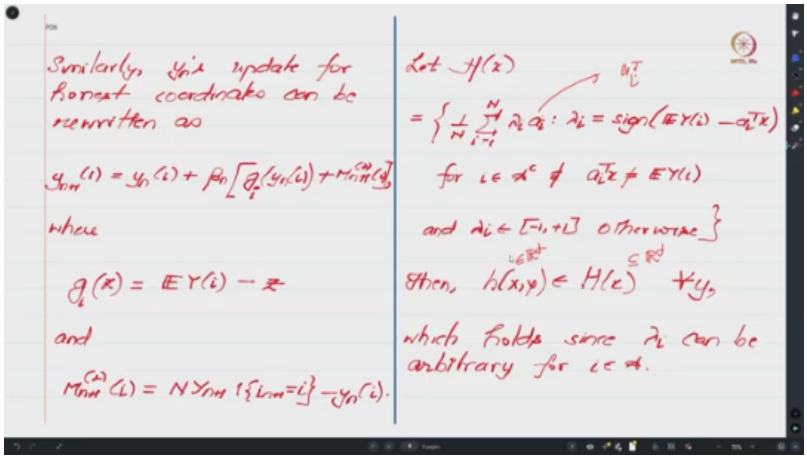
So DI stands for differential inclusion. In particular, the limiting differential inclusion that governs the update rule of Xn is basically this. So now let us just try to understand what is this set value dynamics of help or what help does this provide. So as I told you, The adversary, see this is the update rule.

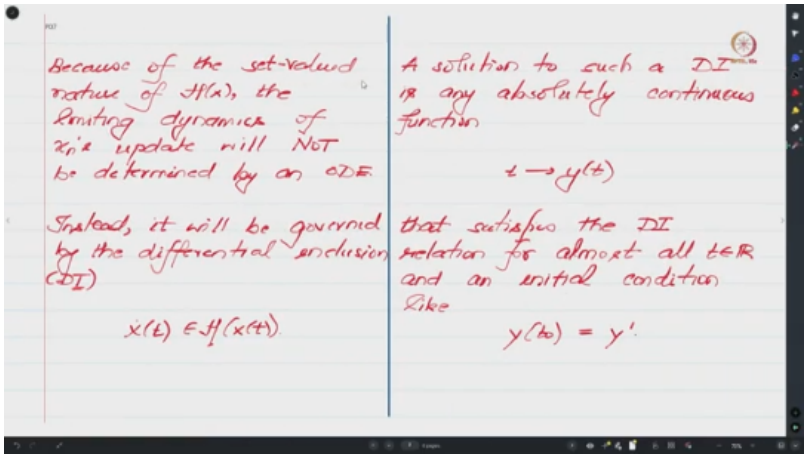
But as I told you, when we have adversaries, we can reimagine the update rule to be of this form. And we can imagine that this, you know, some coordinates of this Y are coming from adversaries. In particular, Y of I for I in A comes from adversaries. So the adversary can basically do whatever they want. So, when I am at X, your H of XY could take different values depending on what your value of Y is.

And in some sense, what we are saying is that whenever we are at X, This is the nice part of the update we would like to go. However, this expression is controlled by adversary. Now, whatever be the choice of adversary depend on this value of Y of I's, this will take some value, right? So, either this could be plus 1, minus 1 or maybe something between minus 1 and plus 1, right?

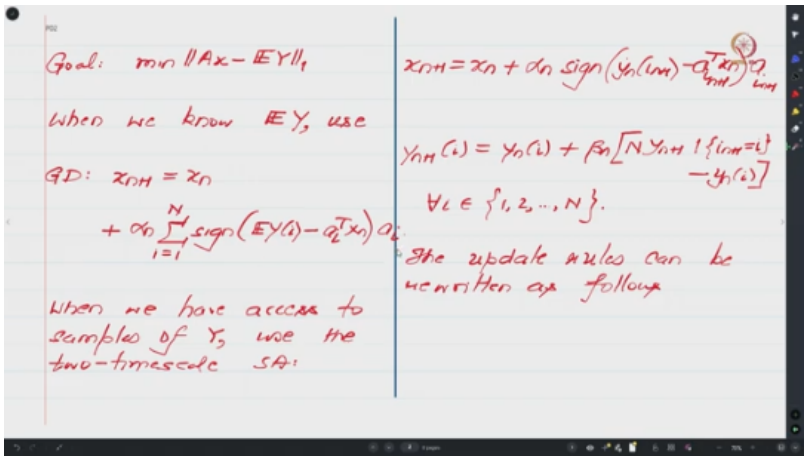


So, whatever be this value, right? This H of XY will belong to capital H of X. So, in this sense, what we are saying is that the adversary can, you know, pick a sine function each time we visit x, right? And hence, although we are visiting x or visiting the same x at different points in time, the adversary could, you know, change the value of y. But whatever the adversary does, it will be an element of this sine. So, in this sense one can see that the set value dynamics is quite natural in the presence of adversaries.





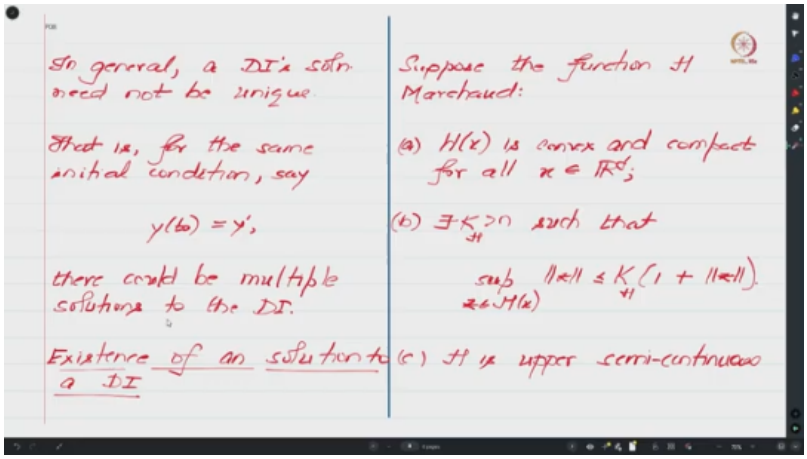
I mean, it is also natural in the case of sub-differentials and so on and so forth. And you know, the idea of sub-differential arises over here because this absolute value or the L1 norm is not differentiable everywhere. And that sort of gives rise to sub-differentials when the arguments to this absolute value are present in this expression. So, when the argument to this norm is actually 0, then this expression that we have over here is not actually a derivative but rather a sub-differential. So, the sub-differential also gives rise to, you know, set-valued dynamics, but I would not like to go into those details at this point in time. Is this okay?



Right. So, that is how it is. So, now one can ask, you know, okay, what is a differential inclusion? In particular, what can we say about its solutions and so on. Right. Now, first, let us ask: what is a solution of a DI?

Right. A solution of a DI is any function of time. So, you take T in R and Y of T in R^D. So, Y of T is an element in R^D, and you have a map like this, and we require that this mapping be absolutely continuous.

And for almost all t in R, because this is absolutely continuous, one can guarantee that its derivative exists for almost all t in R. And for almost all t in R, this derivative should satisfy this relation that we have given over here. And in addition, this y of t should satisfy some initial condition of this form. So, let me again repeat: any map of this form which is absolutely continuous, which satisfies this differential inclusion relation for almost all t in R and an initial condition like this, will be said to be a solution of such a differential inclusion. And I would like to highlight that, unlike in the ODE case where we insisted that this vector field be Lipschitz continuous, that ensures the uniqueness and existence of solutions. In general, the solutions of a DI for an initial value problem need not be unique.



Which means that for the same initial condition, you could have several y of t trajectories, right? Which will pass through this initial condition, but they will be different, right? So, in other words, there could be multiple solutions to this DI, right? And the, you know, the next question is, okay, if the solutions are not unique, at least can we guarantee the existence of a solution to a DI? And one of the sufficient conditions that guarantees the existence of solutions of the DI is the fact that this function H be Marchaud.

So, when is a set-valued function H Marchaud? Well, it is Marchaud if it satisfies the three properties that are listed over here. In particular, the first property requires that for every x, H of x, which is a set, should be convex and compact. Furthermore, we should have a constant K_H, right, which is greater than 0, and for every element z in this set H of x, we require that its norm be upper-bounded by this constant K_H times 1 plus the norm of x. Sorry about this. This should be the norm of x.

So, which means that when x is very large, this quantity can be very large, but of course, it has to have a linear growth. And when x is very, very small—like very close to the origin—then we require that this expression be bounded by this constant. So, this is like a linear growth condition, you know, in terms of x that we insist on—you know—on every element z in H of x. And finally, we require that this function H be upper semi-continuous. So, I will sort of give the definition of upper semi-continuous in the next class, but we also require that this function H be upper semi-continuous. So, this brings us to the end of this class. Let me quickly summarize what we did.

In the previous two classes, we sort of set the stage for why we need two-time-scale stochastic approximation. In particular, we could have some functions, you know, which do not allow interchange of expectation on those functions, right? In particular, functions which are not linear, right? And in that case, you know, we need this notion of two-time-scale stochastic approximation. On the other hand, today we discussed the motivation for working with set-valued dynamics.

In particular, when we have adversaries and so on and so forth, it is often the case that for a particular input, the update direction could lie in a set. And when it lies in a set, that is what gives rise to set-valued dynamics. Such dynamics need to be studied from a stochastic recursive inclusion perspective, which implies that the limiting dynamics will actually be governed by a differential inclusion. And in the last few slides of today's talk, we looked at sufficient conditions that guarantee the existence of solutions of a DI. With that, let me end today's lecture. Until next lecture, goodbye and namaste.