# Week 11 : Lecture 63 : Adaptive control: Part 1

So, what we want to do, well look at briefly is what I do in my adaptive control course and this is backstepping adaptive control. So, you already know what is backstepping. So, it turned out that actually the name backstepping I told you right, it came from the KKK Kanalakopoulos Kokotovich Christi, actually Christi I guess named it as backstepping, the method was sort of known before that. So and he is one of the key researchers in the area of adaptive control. So the entire concept and utility of backstepping came from adaptive control. So not the other way around, although it seems like we are doing the other way around.

So we already did for the non-linear system case backstepping ideas, very powerful, you can easily construct sequentially CLFs and do a lot of things. In fact even automate it if you can, I mean you can do symbolic, symbolically keep generating new functions if you do not want to keep track of the complicated terms. So backstepping like I said came from adaptive control. So obviously it makes sense that it is very applicable in adaptive control.

So modern adaptive control uses a lot of backstepping. So what is this? We will not look at all these, I mean what they mean and so on. So I am not going to sort of worry about this, worry about trying to explain this because this is an adaptive control course. So obviously there was more material here. So backstepping obviously is a method for generating strictly Lyapunov functions or the way we said it is control Lyapunov functions.

So control Lyapunov functions and strictly Lyapunov functions sort of the same things. Basically V dot is negative definite. For any CLF you will see V dot is negative definite that is how it is defined in fact. So that is what is called strictly Lyapunov functions. When you take a V which is nice and positive definite and all that and the V dot is negative definite then it is a strictly Lyapunov function.

And we just saw spring mass damper sort of example where even though we know that the system is asymptotically stable the V that we designed or chose did not have a negative definite V dot. So those are not strictly Lyapunov functions. They usually give trouble in adaptive control. They are not nice for adaptive control. So which is why backstepping is so popular because adaptive control requires CLF or strictly Lyapunov functions.

So suppose we have this nonlinear double integrator. We saw double integrator. Now we have a nonlinear double integrator and this is where we have the unknown appearing. You see that this is only one function. This could easily have been summation of theta i star fi plus u.

It could have been very easily that. So that could have been like a function approximator if you may. So it does not matter if there is 1 or 10 or 100 and so on. You can still use adaptive control ideas for this. But this is the unknown.

This could be again inertia or parameters for identifying the general function with these f being the basis functions and so on. But what we want to do is we want to achieve some kind of a tracking for this. We want to achieve some tracking results. So that is what I say. So everything is in real.

So I am keeping the presentation simple of course. So everything is real numbers. f is of course a map from the state and time to the state again or whatever or the vector derivatives again. So what is the objective? Is tracking. What is tracking? It means that I have a signal r and the position follows the r, velocity follows the r dot because I have this matching kind of a requirement.

So the trajectory also has to satisfy this. The velocity is derivative of position. So what is the dynamics for the errors now? This is what we have been doing for the tracking problem. We construct an error. We write the dynamics of the error.

What is the error dynamics? E1 dot is E2 comes by virtue of the matching condition. And E2 dot is just x1 double dot or whatever x2 dot minus r double dot. So x2 dot is this minus r double dot. So this is my new dynamics that I am working with. I still have an unknown.

If I did not have an unknown, I would simply cancel this guy, cancel this guy, introduce the nice terms that I wanted. So I already said that this is a bad Lyapunov function for this system. It leads to what is called detectability obstacle. So we will try to use backstepping to construct Lyapunov functions here because you know that this is a non-strictly Lyapunov function even for the known case. You have seen this spring mass diagram.

What do we do? Standard backstepping. We have E1 dot is E2. We assume that the E2 is the control. So we design an E2 desired. What is the E2 desired in this case? Just a minus k1 E1.

You are just trying to make this go to 0 exponentially. Then this is a good enough system to follow. So if I did that, great. And what would be the corresponding Lyapunov, candidate Lyapunov function? It is half E1 square. Because with half E1 square and this dynamics, I get V1 dot as minus k1 E1 square.

Yeah. This is what, yeah, is for the first states, for the first system, this is the Lyapunov function. Right. Now I will augment it, right, using the backstepping error and so on. So what is it? All of this happens when E2 is exactly equal to E2 desired which is not possible.

Yeah. We do not control E2 itself. Yeah. So we construct the backstepping error. Okay. What is the backstepping error? It is this.

E2 minus E2 desired. Because we cannot make E2 equal to E2 desired, we try to derive E2 to E2 desired. This is the idea of backstepping. Okay. So what is E2 minus E2 desired? It is E2 plus k1 E1.

Yeah. And this is why it is denoted as xi2. Okay. This is denoted as xi2. So now one of the questions that I ask which anyway we also answered in backstepping, I believe, that does this mess with the original control objective. The original objective was to drive E1 and E2 to 0.

But now with the new dynamics, my objective will be to drive E1 and xi2 to 0. So what happens? If that, if indeed E1 and xi2 go to 0, E1 goes to 0 and xi2 goes to 0, but in inside xi2 I also have E1 which is going to 0. Therefore, E1 and xi2 going to 0 is the same as saying E1 and E2 are going to 0.

Okay. And vice versa. You can check. Yeah. Because this transformation is sort of a non-singular transformation.

It is a nice transformation. Yeah. Nice valid transformation of the states. Okay. All right. Great. So you have the first state E1 and you have the backstepping error state xi2 and I take the derivative of xi2 to find the dynamics.

Yeah. And I get this guy. Yeah. It is just E2 dot plus K1 E1 dot. So that is this plus K1 E2.

All right. Clear? Okay. Fair enough. Now what, what do we do? What do we add as the new term is the square of the backstepping error, right? Every time when we do backstepping, all we are doing is taking the original Lyapunov function and adding to it the square of the backstepping error. Always. This is how we come up with the CLF, right? This is what we proved in our backstepping result.

Okay. Great. So what is V2? V2 is half xi2 squared and what is V2 dot? It is this guy. xi2 times the derivative of xi2. Yes? All right. So right now this, the way this is done is we are, we are not looking at the V completely as of now. So as you understand the V for the entire system would be V equal to V1 plus V2, right? We have chosen a control even before we did that analysis.

We are choosing a control right from here. Yeah. And how do we choose it? Basically cancel this guy, cancel this guy, cancel this guy and introduce a good term. Okay. That's all we are doing. This is, this is cancel, this is cancel, this is cancel and a good term is introduced in the xi2.

Okay. This control works. Ideally I would not recommend doing like this. I would say you first do this V equal to V1 plus V2, take its derivative, then guess the control. Okay. So what is, let's see. What is V equal to V1 plus V2? V is, V1 was E1 squared by 2, right? And V2 is xi2 squared by 2, right? So V1 dot is E1 E1 dot, V2 dot is xi2 xi2 dot.

Okay. Alright. So what is E1 E1 dot? E1 dot is E2 and xi2 xi2 dot is, because of this choice, it is minus K2. So I get minus K2 xi2 squared. Yes? Okay. Because I cancelled everything.

All I am left with is minus K2 xi2. Okay. Okay. Great. Now I get this variable E2, which is not my variable anymore, right? Because I did a transformation. So I want to write E2 in terms of the new variable, which is xi2 and E1.

I do that. Right? xi2 is just E2 plus K1 E1. Right? So I have just written E2 in terms of the new variable. Yeah? And once I do that, what do I get? I get this nice negative term in E1 minus K1 E1 squared. I already had the nice negative term in xi2 minus K2 xi2 squared. And I also have a mixed term, E1 xi2.

Right? But I already know what to do with this. We have done this before. We use this, that 2AB less than equal to A squared plus B squared for this mixed term. So this mixed term is basically going to become less than equal to half E1 squared plus half xi2 squared, which is this. Yeah? Okay. So, instead of an equality, now I move to an inequality.

That's it. All right? Because I have, I have less than equal to is always fine. We are doing the Lyapunov analysis. Right? So I have minus K1 E1 squared minus K2 xi2 squared and then this additional term. Okay? But that's pretty straightforward. I mean, I can always use K1 to dominate the half, K2 to dominate the half, and V dot is negative definite.

Right? This is what the entire trick of Lyapunov analysis is all about. All right? And once I do that, what happens? I know that I have V dot negative definite. What does it mean for V dot to be negative definite? It means that both E1 and xi2 are going to 0.

Whatever is in V. Basically V is going to 0. So whatever is in V, that is E1 and xi2 are going to 0. And we have already proved that E1 xi2 going to 0 is equivalent to saying that E1 E2 is going to 0.

Okay? So done. Okay? Now, great. Excellent. Yeah? We have got a strictly Lyapunov function. Why? Because V dot became strictly less than equal to 0. Yeah? So unlike if I had taken E1 square plus E2 square by 2, instead of E1 square plus xi2 square by 2, I would have landed in trouble. E1 square plus E2 square by 2 will only give me V dot negative semi definite.

Okay? So we have something nice here. Now, but there is a problem. In the control, yeah, although here if you see the way this is written, it's written as theta hat. But you see it says theta hat is theta star. So basically this is a known case, what we call as the known case.

If the parameter was known, you can cancel this exactly. Okay? If the parameter was completely known to you, accurately known to you, then obviously you can do this cancellation. Yeah? But now we are in the adaptive framework. We do not know this parameter. Okay? We know nothing about this.

Okay? Then what do we do? Okay? So great. So one nice step is done. We have constructed a CLF. Yeah? And this is critical for adaptive control.

Without a CLF, you cannot do adaptive control. Okay? Remember this. Yeah? For the known case, by the way. I am not saying CLF for the entire system after adaptation and all that. I am saying if you, if the, if even when the parameters are known, you don't have a strictly aponop function or a CLF for your control system, then you have a problem. You will not be able to use that to do adaptation. Okay? So even in the known case, if it is non-strict, then in the unknown case, it becomes even worse.

Okay? That is essentially the idea behind detectability of stepping. So even at least for the known case system, you must have a CLF. And that is why you use backstepping. Okay? If you have a CLF in, by some other means, feel free to use it.

No problem. If you can guess it without doing backstepping, no problem. Yeah? If you can guess the control Lyapunov function without using backstepping, not a problem. Absolutely feel free to use it. Yeah? But more often than not, you will not be able to guess it.

Alright. Great. So what happens when theta star is unknown? Okay? I have already sort of given a glimpse. What we do is, instead of theta star, we use what is called theta hat. Okay? So this is the estimate of theta star. This is called the estimate of theta star.

So basically we try to estimate theta star. Okay? We don't, because we don't know the value. So what's the best thing we can do? We try to get an estimate. This is what control folks do. What is an estimate? It's like if you've seen Kalman filtering, you have a state estimate. Right? You don't have the true state, but you use some sensor data and you feed it into a filter.

Yeah? Probably you don't know what filter you're feeding into. Whatever. But you feed it into a filter and you come up, you get out what is called an estimate of the state. It's not the true state. Yeah? Because you are seeing the world through the sensors. Okay? So there's no real concept of true state.

Yeah? I mean you may have a slightly more accurate state. For example, if I have a bunch of vision sensors with which I'm trying to identify my current location, my current position, x, y, z position. Okay? And I do a pretty good job. Okay? But that is still a sensor data. Right? But real, what a lot of folks would do is they would try to compare this vision process data to say GPS data or very accurate GPS data. If you have say military grade GPS data, it will have some, you know, sub millimeter accuracy and then you can compare that.

Yeah? Your position given by GPS with your vision based data. Okay? So there's no real true, so all the data, everything, you see the world through the data, right? Through the sensors. So therefore there's always an estimate involved. Yeah? In the states also, in fact. Yeah? Although we don't talk about it in this course. Yeah? So the idea is here also what we do is we create an estimate for the parameter, not for the state but for the constant parameter.

This is what is the adaptive control idea. Okay? This part is called adaptive estimation but it combines with the control so it's called the adaptive control. It shows up in the control. Alright? Okay. Great. So we replace it with an estimate, we replace the true value with an estimate and then we try to figure out how to calculate the estimate.

Okay? Because just like in a Kalman filter there is a particular logic by which you update the states of the Kalman filter. Yeah? Comes from some kind of an optimality. Right? Kalman filter is basically coming from an optimality result. Yeah? So similarly here it's not coming from an optimality result I can tell you.

It comes from a stability result. Okay? This is also an estimate typically in deterministic systems that is where there's no probabilistic quantities coming in. Estimates or observed states, sorry, actually I would say estimates come from stability requirements.

Yeah? Not optimality requirements. Alright. Okay. Great. So what is the stability requirement? We want to drive the event into 0. Okay? That is our stability requirement. We want to be able to track even if we don't know the true value of the parameter. Okay? Great.

Now what? We create a slightly modified candidate Lyapunov function for the unknown case. We already have a strict Lyapunov function for the known system V1 plus V2. We add to it a term in the parameter error. I don't know the parameter.

Best I can do is try to drive my estimate to the true parameter. Right? Okay. Again, same logic by which most of control folks will work. Yeah? What is this? I define this theta tilde. This is a notation that we very standardly use in adaptive control.

Tilde denotes the parameter error. Theta star denotes the true value. Theta cap denotes the estimate. Okay? Standard in even in nonlinear control for that matter. Okay? Now we take time derivative just like we were doing earlier.

What is it? V1 dot is E1 E1 E1 dot. So E1 E2. V2 dot is okay. Now I have to write some terms I guess. Sorry. So if you remember this is E1 squared by 2 plus psi 2 squared by 2. Right? So V dot if I make this big that I can write now. V dot is E1 E1 dot plus psi 2 psi 2 dot plus actually minus theta tilde theta hat dot divided by gamma.

So this gamma is some positive point. Gamma is just a positive scaling called the adaptation gain. Okay? So, routinely controls how fast you will update the parameter. That's it. Okay? So if you notice the last term had a negative here.

Why? Why does the last have a negative sign here? Yeah. Theta minus theta hat. And this is the constant. So the derivative value goes to zero.

So this is minus theta hat dot. Okay? Simple. Alright? Nothing too complicated. Now I am going to write the terms. Right? This is E1 dot E, E1 E1 dot is E1 E2.

Just like before. Plus psi 2 psi 2 dot. Psi 2 dot was what? Theta times f x t. Plus u minus r double dot. Yeah? And I keep the last term.

As it is. As it is. Now what did I choose my control as? I chose it to cancel these terms. Tried to cancel this term by saying theta hat f x t. Yeah? So these two terms I can still cancel. Right? If you see the control.

Yeah, it's this right? Right? So this term will still cancel. This term will still cancel. The nice term will still appear. Yeah? But this term will not cancel. Right? So what will I be left with? Correct.

So this is where the theta tilde shows up. This is basically, yeah that's basically how I get from here to here. Yeah? Because the control will have a theta hat. This will bring a minus theta hat f and minus k1, k2 psi 2. So theta minus theta hat f is theta tilde f.

And that's this guy. Okay? Yeah? Can you confuse me? You can just do it on your own and see. Yeah? So that's it. From here I go to this step. Alright? Now what? Now it's not that difficult. Now what do I do? I of course substitute for E2 because I want to write everything in terms of the new variables just like I was doing earlier.

Right? So I'm left with E1 psi 2 minus k1 E1 square minus k2 psi 2 square. And I do one more thing. I you see that now this term theta hat and this term both have theta tilde common. Right? So I take the theta tilde common and I use this theta hat dot to cancel this.

I can. If I choose theta hat dot as gamma times psi 2 f. Yeah? Absolutely. That's our estimate. Right? What I. So I've just written it here if you see. All I've done is I've taken these two terms and I've taken theta tilde common. Right? And this gives me an ek and something in theta hat dot.

Right? What do I do? I cannot make this negative definite. Remember I don't know theta tilde. So I cannot give something like minus k theta tilde or anything like that because theta tilde contains the unknown.

It's a parameter error. I don't know it. Okay? So the best I can do is make this zero. That's the best I can do. I'll try to make this zero because I can't make this negative definite. Right? And how do I make this zero? By this choice. Okay? Yes? Cannot make this negative definite in adaptive control because theta tilde is unknown.

Otherwise you will have to have a minus k theta tilde which is not allowed. Right? It's an unknown. So that's not possible. So the best I do is I make this zero. Okay? So this is gone.

And what do I have here on the right hand side? You see I have all known quantities. Right? Gamma is known to me. Some adaptation gain. This is why I said it's adaptation gain because it sort of gives you a rate at which you are adapting.

And then xi two is the second state. It is just e two plus k one e one. So I'm assuming that you are measuring the state. So you know this. And then f is just some function.

If you are thinking in neural networks, it's just basis functions. Some basis functions. So known to you obviously. Even if it's time varying, it doesn't matter. But it's known to you. Okay? So once you have cancelled everything, you are left with what? k one e one square, k two xi two square, e one xi two. Notice v dot looks exactly like the known case.

Yes? I started, I mean even though I started with a more terms in v, I ended with the same v dot. This is why starting with non-strict Lyapunov functions will land you in trouble. Okay? If only e two was appearing here, then you will have something like this. This will be a big problem for you. Okay? So in adaptive control, the way we do it, the v dot looks exactly like the known case.

Okay? So looks exactly same as known case v dot. Okay? Even if you started with a additional term in v. Okay? What can you say about this v dot now? Definiteness. Of course you can do this a b less than equal to a square b square and all that. What about this definiteness? Negative definite? Is this v dot, is this v, so is v dot negative definite? Many times you guys make the same mistake.

But I added a new state. Theta tilde is now a state. It's not a, it's a new variable there. So v is now a function of three variables. But it contains only two in v dot. It's only negative semi definite. Okay? So even if you started with a nice clf with a negative definite v dot for the known case, as soon as you move to adaptation, your v dot will become negative semi definite.

Why? Because you had more terms in the v. Which are new variables. Okay? Yeah, this should be very clear in your mind. Okay? When you go to the adaptation, you add an adaptive control part, your v dot will become negative semi definite. And that's what I've written here. So the v dot is negative semi definite.

Right? So what does this give you? As always from the Lyapunov theorem, all that you get is uniform stability. Nothing more. Okay? Then of course now we are left with trying to use Babel-Athlema and so on. So of course you get from this step to this step also using this sum of square type of result which we have been using regularly. Alright. Thank you.