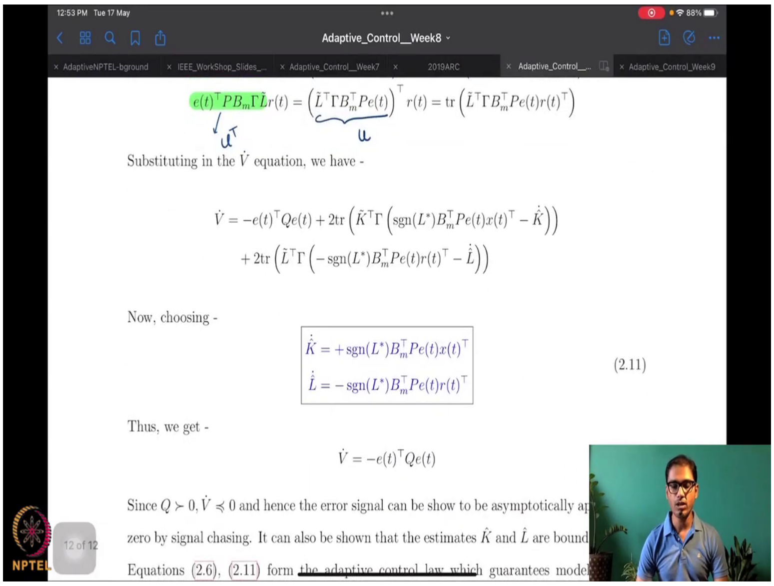
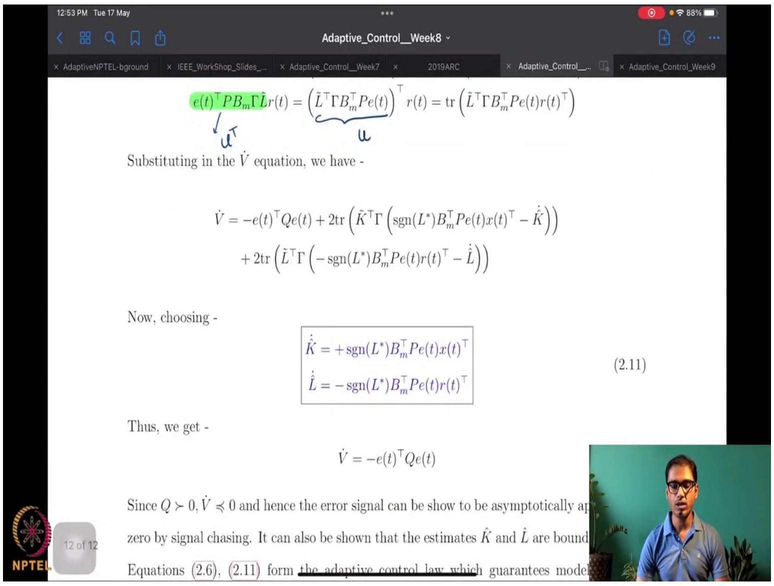
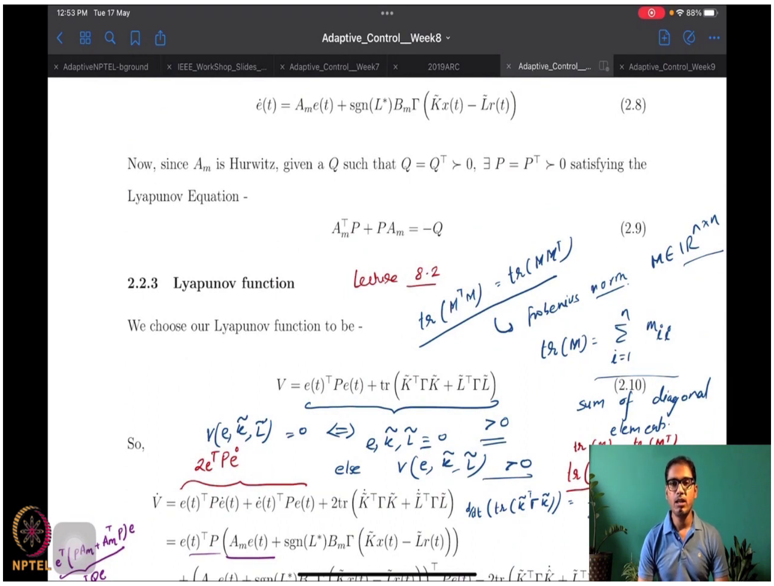
Nonlinear Adaptive Control
Professor Srikant Sukumar
Systems and Control
Indian Institute of Technology, Bombay
Week 8
Lecture No: 45
Generalisation of Adaptive Integrator Backstepping Method (Part 1)

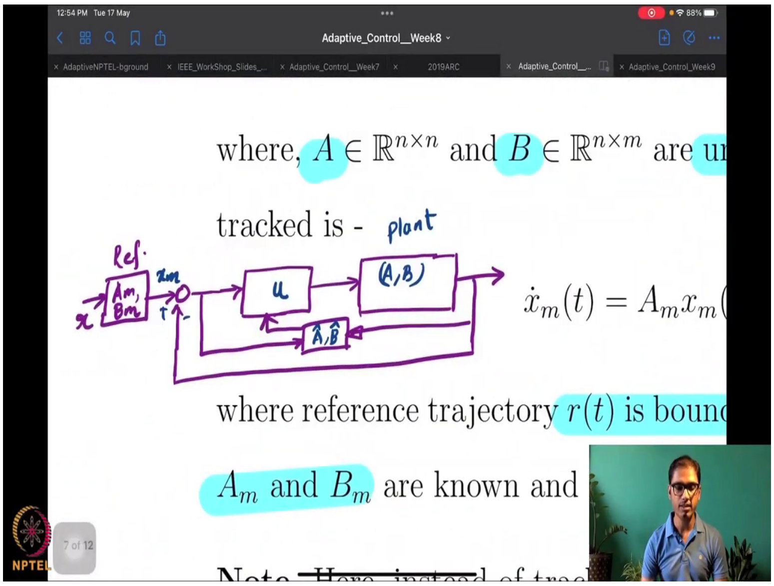
Hello everyone, welcome to yet another session of our NPTEL on Non-Linear and Adaptive Control, I am Srikant Sukumar from Systems and Control, IIT, Bombay. So, we are into the 8th week of this course on non-linear adaptive control. And we are already looking at several algorithms, we have already looked at several algorithms, which will help us to control autonomously systems such as what we see in the background of a satellite orbiting the earth. We are also now able to design algorithms against uncertainties in such systems.

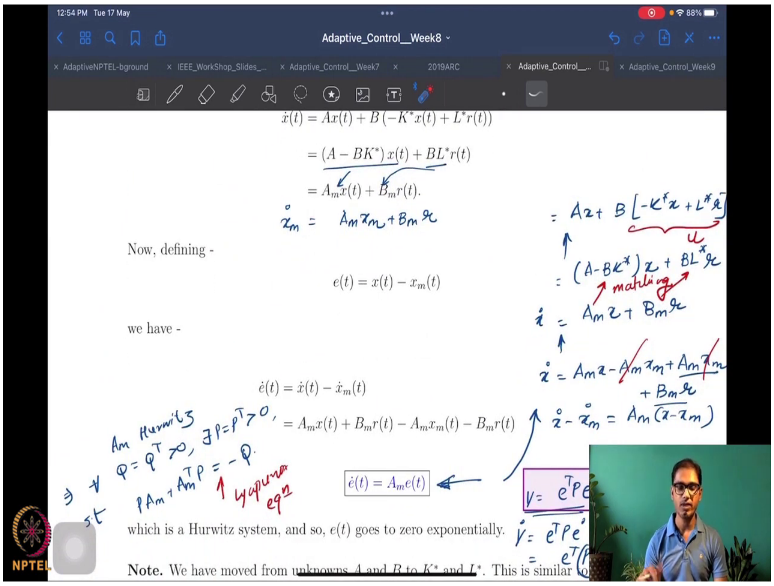
(Refer Slide Time: 1:06)

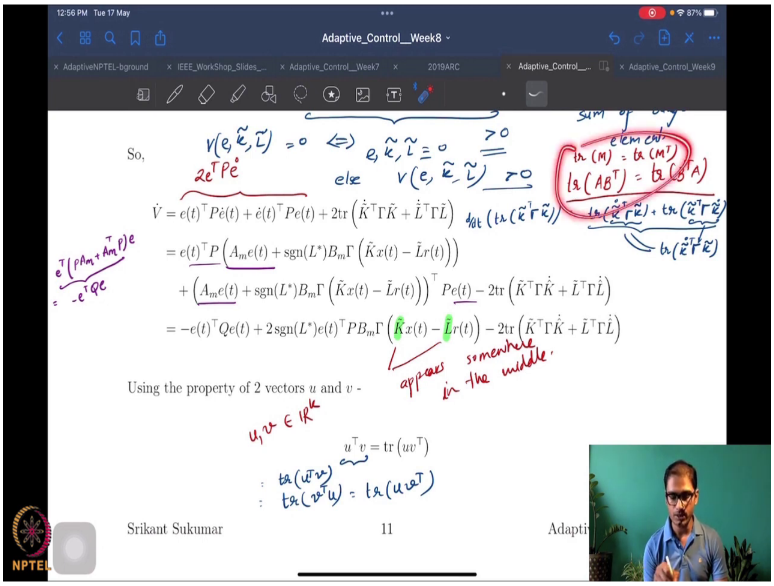



12:53 PM Tue 17 May
Adaptive_Control__Week8
AdaptiveNPTEL-bground
IEEE_WorkShop_Slides_...
Adaptive_Control__Week7
2019ARC
Adaptive_Control_...
Adaptive_Control_Week9
$e(t)^\top PB_m\Gamma\tilde{L}r(t) = \left(\tilde{L}^\top\Gamma B_m^\top Pe(t)\right)^\top r(t) = \mathrm{tr}\left(\tilde{L}^\top\Gamma B_m^\top Pe(t)r(t)^\top\right)$
Substituting in the $\dot{V}$ equation, we have -
$\dot{V} = -e(t)^\top Qe(t) + 2\mathrm{tr}\left(\tilde{K}^\top\Gamma\left(\mathrm{sgn}(L^*)B_m^\top Pe(t)x(t)^\top - \dot{\hat{K}}\right)\right) + 2\mathrm{tr}\left(\tilde{L}^\top\Gamma\left(-\mathrm{sgn}(L^*)B_m^\top Pe(t)r(t)^\top - \dot{\hat{L}}\right)\right)$
Now, choosing -
$\dot{\hat{K}} = +\mathrm{sgn}(L^*)B_m^\top Pe(t)x(t)^\top$
$\dot{\hat{L}} = -\mathrm{sgn}(L^*)B_m^\top Pe(t)r(t)^\top$
(2.11)
Thus, we get -
$\dot{V} = -e(t)^\top Qe(t)$
Since $Q \succ 0$, $\dot{V} \preccurlyeq 0$ and hence the error signal can be show to be asymptotically ap
zero by signal chasing. It can also be shown that the estimates $\hat{K}$ and $\hat{L}$ are bound
Equations (2.6), (2.11) form the adaptive control law which guarantees model
12 of 12
NPTEL


12:53 PM Tue 17 May
Adaptive_Control__Week8
$\dot{e}(t) = A_m e(t) + \mathrm{sgn}(L^*)B_m\Gamma\left(\tilde{K}x(t) - \tilde{L}r(t)\right)$ (2.8)
Now, since A_m is Hurwitz, given a Q such that $Q = Q^\top \succ 0$, $\exists\, P = P^\top \succ 0$ satisfying the
Lyapunov Equation -
$A_m^\top P + PA_m = -Q$ (2.9)
2.2.3 Lyapunov function
We choose our Lyapunov function to be -
$V = e(t)^\top Pe(t) + \mathrm{tr}\left(\tilde{K}^\top\Gamma\tilde{K} + \tilde{L}^\top\Gamma\tilde{L}\right)$ (2.10)
So,
$\dot{V} = e(t)^\top P\dot{e}(t) + \dot{e}(t)^\top Pe(t) + 2\mathrm{tr}\left(\dot{\tilde{K}}^\top\Gamma\tilde{K} + \dot{\tilde{L}}^\top\Gamma\tilde{L}\right)$
$= e(t)^\top P\left(A_m e(t) + \mathrm{sgn}(L^*)B_m\Gamma\left(\tilde{K}x(t) - \tilde{L}r(t)\right)\right)$
lecture 8.2
$\mathrm{tr}(M^\top M) = \mathrm{tr}(MM^\top)$
Frobenius norm
$M \in \mathbb{R}^{n\times n}$
$\mathrm{tr}(M) = \sum_{i=1}^{n} m_{ii}$
sum of diagonal elements
$V(e,\tilde{K},\tilde{L}) = 0 \iff e,\tilde{K},\tilde{L} = 0$
else $V(e,\tilde{K},\tilde{L}) > 0$
$2e^\top P\dot{e}$
NPTEL


12:54 PM Tue 17 May
Adaptive_Control__Week8
where, $A \in \mathbb{R}^{n\times n}$ and $B \in \mathbb{R}^{n\times m}$ are u
tracked is - plant
Ref.
A_m, B_m
x_m
u
(A,B)
$\hat{A},\hat{B}$
$\dot{x}_m(t) = A_m x_m($
where reference trajectory $r(t)$ is bound
A_m and B_m are known and
7 of 12
NPTEL





So, what we were doing until last time is basically completing the proof for model reference adaptive control. So, model reference adaptive control was a paradigm for adaptive control of linear systems, which is a very very popular, and very very well-known, and very well cited and even used sort of set of methods in adaptive control.

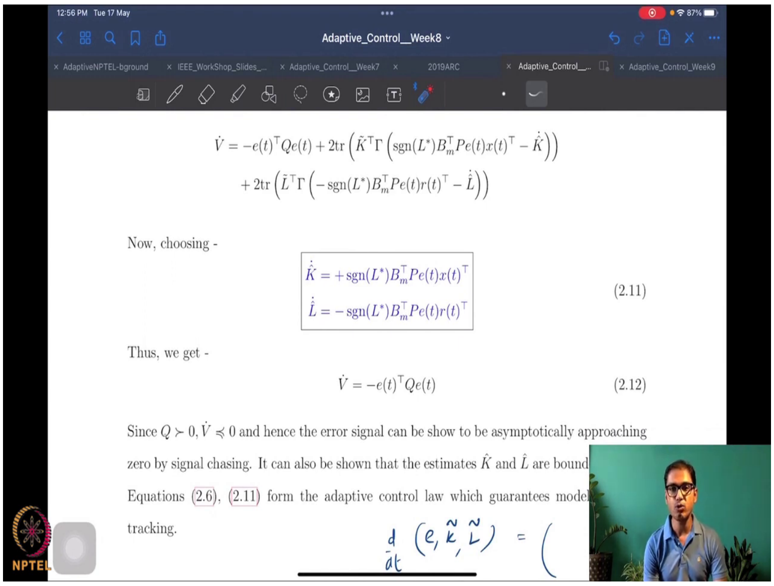
And the basic difference from what we have been doing before this is that, here we actually track a state that comes out of a reference model than just a reference signal. So, I believe the picture looked somewhat like this. So, that is the big difference here that you have a reference model instead of just a reference state and everything else is sort of similar in terms of the analysis the big difference that we saw was that we had unknowns which were actually matrices.

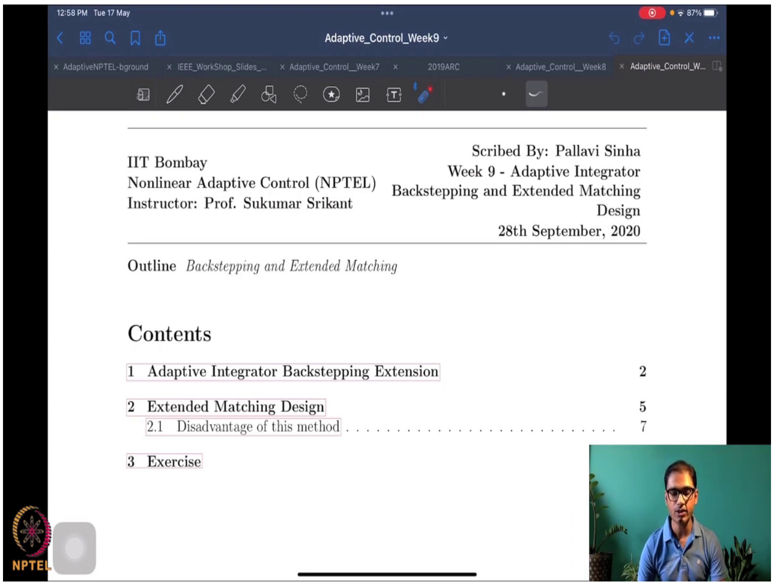
So, we started with A star and sorry A and B being unknowns and then of course we moved it to sort of redesigned it. So, that the unknowns became these K star and L star which were still matrices. And because we were dealing with matrices and matrix unknowns, we also learned a rather novel way of designing Lyapunov candidate functions for matrix states or matrix unknowns.

So, here the parameter error term instead of just being a quadratic or a square of the vector norm is in fact a kind of a matrix Frobenius norm weighted matrix Frobenius norm and it is defined in terms of the trace function. So, this is was the big difference and in order to sort of complete the proof, we also needed to use some very very interesting and useful trace properties.

So, this is something that I hope all of you will remember always because these are very very useful in manipulating any kind of Lyapunov candidates or any in any other context if you see trace functions then it is very very useful to manipulate these trace functions and we can use these equalities in order to do that all.

(Refer Slide Time: 3:44)





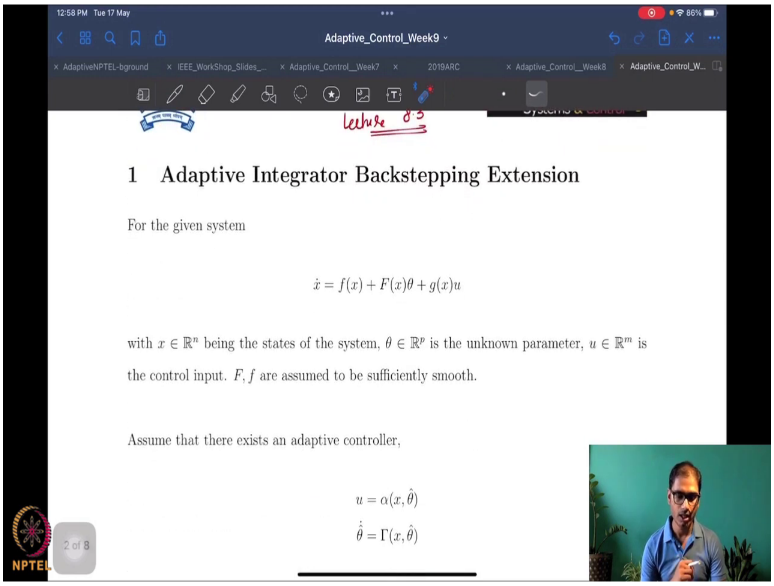
And of course then we designed our update law as usual it was still a certainty equivalence adaptation. So, all was as usual other than these particular features. So, now we are sort of ready to move into the next week's lectures. Again please do not get confused the sort of classification of the week is also in a sense to provide homeworks and just to organize material, as and when we do complete the material for a particular week we will move on to what we call next week lectures.

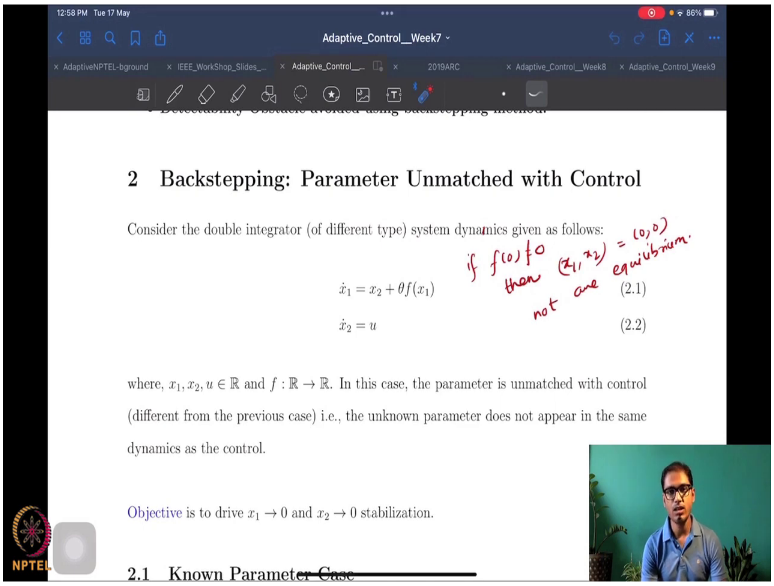
So, although we are in week number 8, we are already going to start looking at week 9 lectures. So, I really hope that this does not cause any confusion should be pretty straightforward this is just for us to you know organize the homeworks in a proper way. So, this week 9 is titled adaptive integrator backstepping and extended matching design.

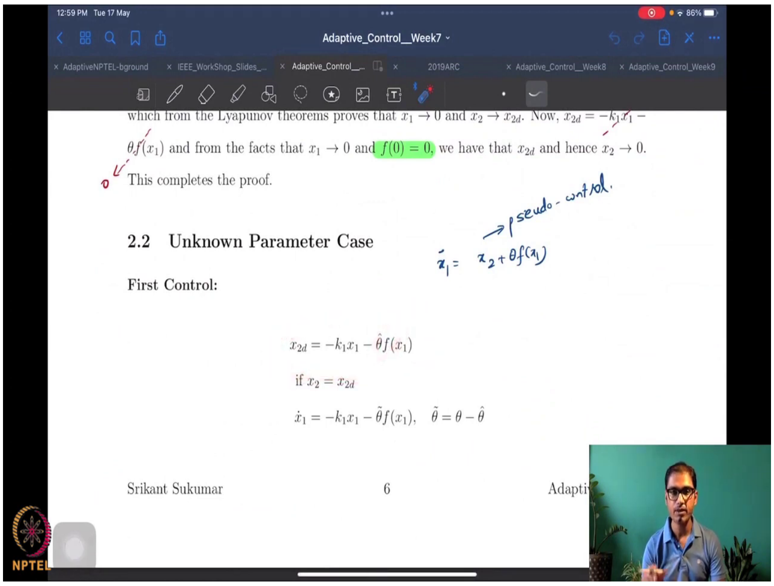
So, we will do a couple of things in this set of lectures the first is, we sort of generalize this adaptive integrator backstepping that we did a couple of weeks ago. So, not the week number 8 but week number 7 lectures, in week number 7 lectures we saw integrator backstepping for the first time, we also saw how to do a backstepping design for the unmatched case.

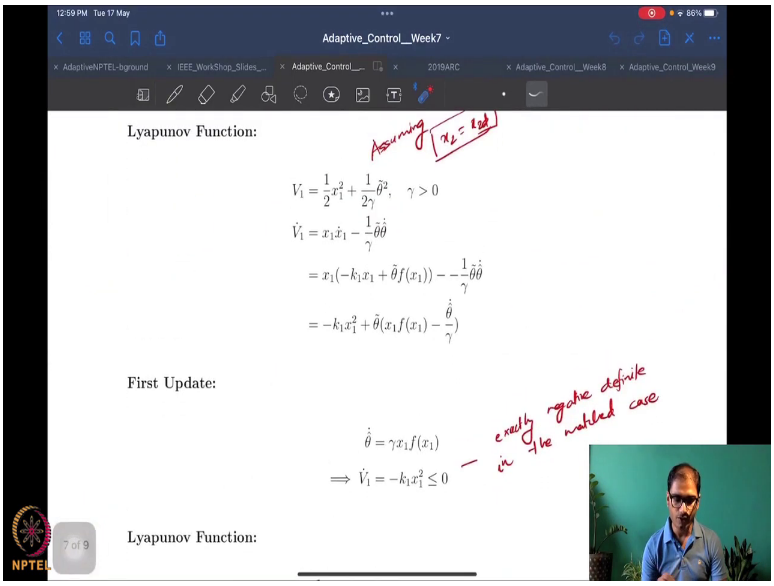
So, what we will do here is to generalize that idea. And then we look at a different method of doing adaptive integrator backstepping. So, that we do not have to resort to over parameterization. So, that is the agenda for week 9 lecture notes. So, let us continue.

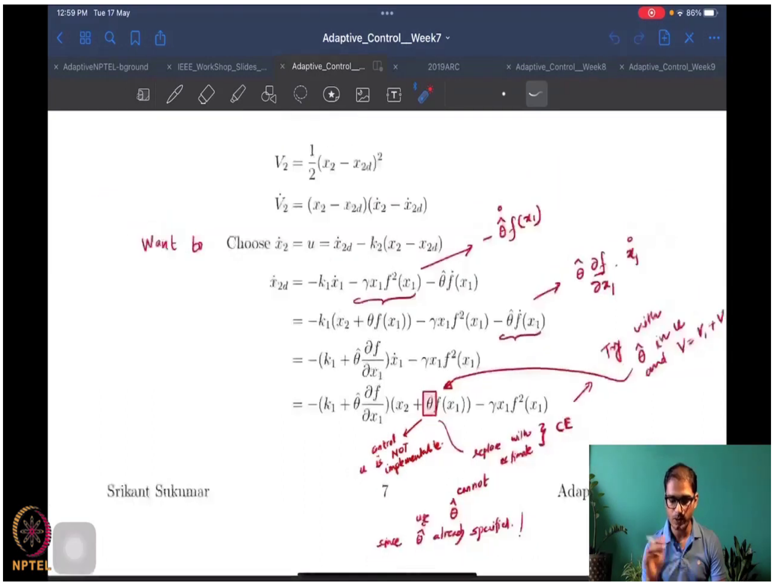
(Refer Slide Time: 5:40)

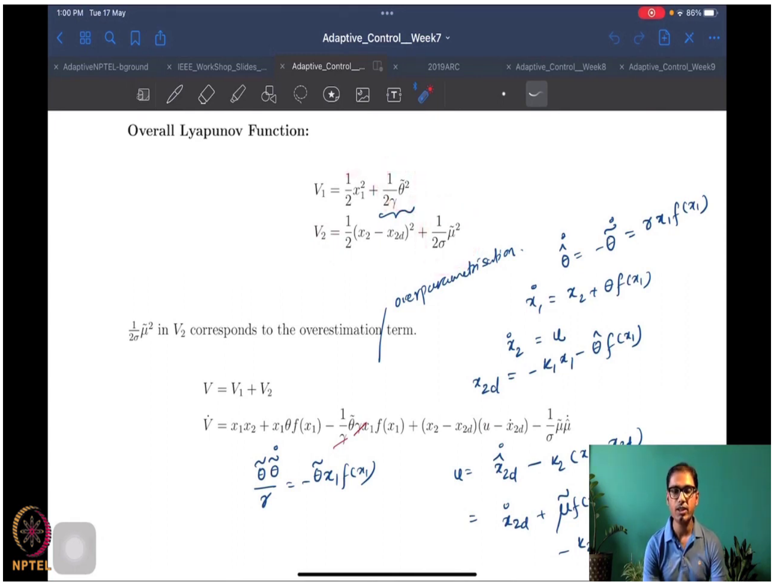


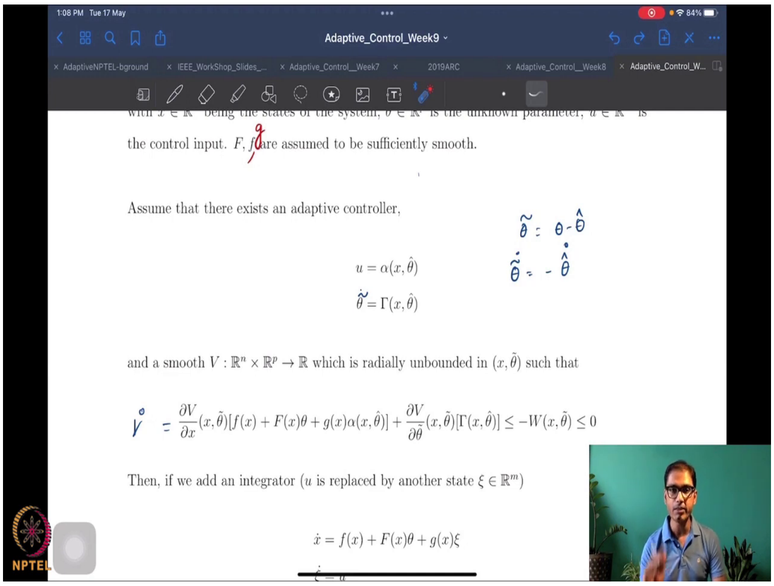


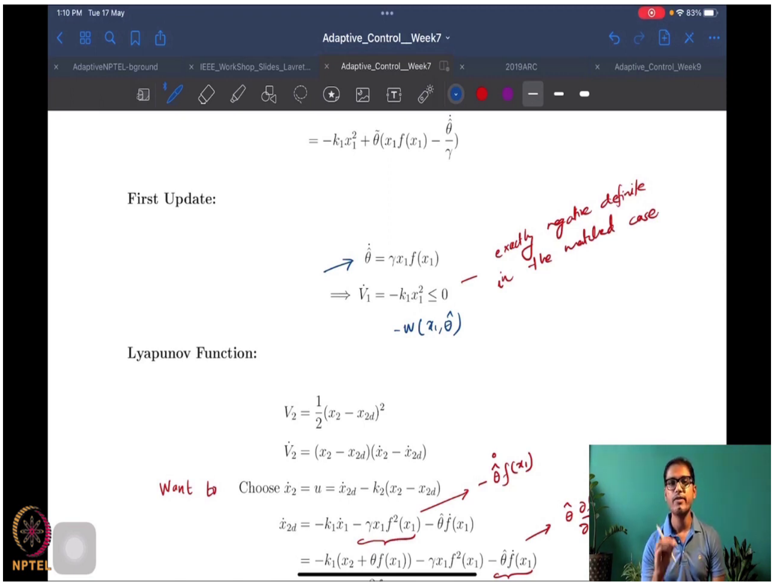

Adaptive_Control_Week7
which from the Lyapunov theorems proves that $x_1 \to 0$ and $x_2 \to x_{2d}$. Now, $x_{2d} = -k_1 x_1 - \theta f(x_1)$ and from the facts that $x_1 \to 0$ and $f(0) = 0$, we have that x_{2d} and hence $x_2 \to 0$.
This completes the proof.
2.2 Unknown Parameter Case
First Control:
$x_{2d} = -k_1 x_1 - \hat{\theta} f(x_1)$
if $x_2 = x_{2d}$
$\dot{x}_1 = -k_1 x_1 - \tilde{\theta} f(x_1), \quad \tilde{\theta} = \theta - \hat{\theta}$
pseudo-control
$\dot{x}_1 = x_2 + \theta f(x_1)$
Srikant Sukumar
6
NPTEL


Adaptive_Control_Week7
Lyapunov Function:
Assuming $x_2 = x_{2d}$
$V_1 = \frac{1}{2}x_1^2 + \frac{1}{2\gamma}\tilde{\theta}^2, \quad \gamma > 0$
$\dot{V}_1 = x_1\dot{x}_1 - \frac{1}{\gamma}\tilde{\theta}\dot{\hat{\theta}}$
$= x_1(-k_1x_1 + \tilde{\theta}f(x_1)) - -\frac{1}{\gamma}\tilde{\theta}\dot{\hat{\theta}}$
$= -k_1x_1^2 + \tilde{\theta}(x_1 f(x_1) - \frac{\dot{\hat{\theta}}}{\gamma})$
First Update:
$\dot{\hat{\theta}} = \gamma x_1 f(x_1)$
$\implies \dot{V}_1 = -k_1 x_1^2 \leq 0$
exactly negative definite in the matched case
Lyapunov Function:
7 of 9
NPTEL


Adaptive_Control_Week7
$V_2 = \frac{1}{2}(x_2 - x_{2d})^2$
$\dot{V}_2 = (x_2 - x_{2d})(\dot{x}_2 - \dot{x}_{2d})$
Want to
Choose $\dot{x}_2 = u = \dot{x}_{2d} - k_2(x_2 - x_{2d})$
$\dot{x}_{2d} = -k_1\dot{x}_1 - \gamma x_1 f^2(x_1) - \hat{\theta}\dot{f}(x_1)$
$= -k_1(x_2 + \theta f(x_1)) - \gamma x_1 f^2(x_1) - \hat{\theta}\dot{f}(x_1)$
$= -(k_1 + \hat{\theta}\frac{\partial f}{\partial x_1})\dot{x}_1 - \gamma x_1 f^2(x_1)$
$= -(k_1 + \hat{\theta}\frac{\partial f}{\partial x_1})(x_2 + \theta f(x_1)) - \gamma x_1 f^2(x_1)$
$-\dot{\hat{\theta}} f(x_1)$
$\hat{\theta}\frac{\partial f}{\partial x_1}\cdot\dot{x}_1$
control u is NOT implementable
CE
Srikant Sukumar
7
NPTEL


Adaptive_Control__Week7
Overall Lyapunov Function:
$V_1 = \frac{1}{2}x_1^2 + \frac{1}{2\gamma}\tilde{\theta}^2$
$V_2 = \frac{1}{2}(x_2 - x_{2d})^2 + \frac{1}{2\sigma}\tilde{\mu}^2$
$\frac{1}{2\sigma}\tilde{\mu}^2$ in V_2 corresponds to the overestimation term.
$V = V_1 + V_2$
$\dot{V} = x_1x_2 + x_1\theta f(x_1) - \frac{1}{\gamma}\tilde{\theta}\gamma x_1 f(x_1) + (x_2 - x_{2d})(u - \dot{x}_{2d}) - \frac{1}{\sigma}\tilde{\mu}\dot{\hat{\mu}}$
NPTEL


Adaptive_Control_Week9
the control input. F, f are assumed to be sufficiently smooth.
Assume that there exists an adaptive controller,
$u = \alpha(x, \hat{\theta})$
$\dot{\hat{\theta}} = \Gamma(x, \hat{\theta})$
and a smooth $V : \mathbb{R}^n \times \mathbb{R}^p \to \mathbb{R}$ which is radially unbounded in $(x, \tilde{\theta})$ such that
$\frac{\partial V}{\partial x}(x,\hat{\theta})[f(x) + F(x)\theta + g(x)\alpha(x,\hat{\theta})] + \frac{\partial V}{\partial \theta}(x,\hat{\theta})[\Gamma(x,\hat{\theta})] \le -W(x,\hat{\theta}) \le 0$
Then, if we add an integrator (u is replaced by another state $\xi \in \mathbb{R}^m$)
$\dot{x} = f(x) + F(x)\theta + g(x)\xi$
NPTEL


Adaptive_Control__Week7
$= -k_1x_1^2 + \tilde{\theta}(x_1 f(x_1) - \frac{\dot{\hat{\theta}}}{\gamma})$
First Update:
$\dot{\hat{\theta}} = \gamma x_1 f(x_1)$
$\implies \dot{V}_1 = -k_1x_1^2 \le 0$
Lyapunov Function:
$V_2 = \frac{1}{2}(x_2 - x_{2d})^2$
$\dot{V}_2 = (x_2 - x_{2d})(\dot{x}_2 - \dot{x}_{2d})$
Choose $\dot{x}_2 = u = \dot{x}_{2d} - k_2(x_2 - x_{2d})$
$\dot{x}_{2d} = -k_1\dot{x}_1 - \gamma x_1 f^2(x_1) - \hat{\theta}\dot{f}(x_1)$
$= -k_1(x_2 + \theta f(x_1)) - \gamma x_1 f^2(x_1) - \hat{\theta}\dot{f}(x_1)$
NPTEL

So, let us suppose that for this system. So, now suppose we are since we are generalizing. So, we are essentially taking vector states and all that. So, well I mean to be honest let me mark this first before we start. So, this is actually lecture 8.3, this is lecture 8.3. So, we will recall is this week 7, I mean I am going to go here to week 7, we had a system like this, where we learned how to do backstepping for unmatched parameters.

And we know how we did it we started with the first system then assume that there is an unknown parameter. So, the desired quantity was designed with a theta cap and then we of course had a Lyapunov candidate design but then when we went to the second Lyapunov function and we tried to choose a control there was a problem because the control is not implementable due to the presence of this theta here.

So, control contains an x 2 desired dot which can which brings up a theta which is not implementable. So, obviously we do not want to use a theta hat again because we already have specified an update law and we cannot do it, it will create an error in the analysis. So, instead of theta hat we put in a new estimate mu hat. So, we put in a new estimate mu hat for the same quantity theta, we have two estimates theta hat and mu hat.

And so with that we declare this new variable as x 2 hat x 2 desired hat dot instead of x 2 desired dot and then we continue this analysis with this new mu hat and then of course you have a candidate Lyapunov function, which also contains a mu tilde on top of the theta tilde. So, essentially because we have two parameters well two estimates for the same parameter, we have two terms corresponding to the parameter errors.

So, there are two different parameter errors. So, that essentially was what we were doing I mean we then specify mu dot and all that in the standard way if you do not remember please go back and revise what we sort of did in this lecture. So, what we want to do today at least in the beginning is that we want to look at a generalization of this then the states are not scalar states because most in most problems more often than not your state should be vectors.

So, it does not make any sense to just restrict ourselves to scalar states. And doing this will also give you a very fair idea that the analysis methods do not change significantly, when you have a vector state instead of a scalar state. So, it is not like things become too complicated. So, that is rather nice.

So, I this is another thing that I want all of you to get used to because suppose I think of any robotic application say a two arm manipulator or think a quad rotor quadrotor has six degrees of freedom. So, three rotational and three translational. So, corresponding to that there are 12 states because three position states three linear translational velocity states three angular position states three angular velocity states. So, there are 12 states. So, it is vectors.

So, we describe the equations in terms of vector equations. Similarly, for a two joint manipulator for a two joint manipulator. So, you can think of the shoulder in elbow if I have a two joint manipulator. So, the shoulder has two states corresponding to the shoulder angle and the shoulder angular velocity and the elbow has two states corresponding to the elbow angle and elbow angular velocity. So, therefore this is again a vector.

So, it does not make sense always to look at scalar states because we might end up having vector states more often than not more often than not. So, that is the purpose two things, one you want to generalize to vector steps and of course this particular method to vector states and second we also want to sort of get a feel for how to deal with vectors in Lyapunov analysis because in many cases the states are going to be vectors and you will see they are not significantly different.

Now that we have this preamble. So, suppose we have this system x dot is f x which is like a drift plus f x theta which is the term containing the unknown parameter plus g x u. Now the states are in some R n the unknown is in some vector R is a vector R p. So, p dimensional vector state is n dimension and you can find the corresponding dimensions of f and f and all that and g and the control input u is assumed to be an m dimensional vector.

So, there are three different dimensions the states are n dimensional, theta is p dimensional, u is m dimension and of course the f and small f and whatever and also g actually also g are assumed to be sufficiently smooth and have all the nice properties. So, that you know we can take differentials if we want and so on and so forth.

So, suppose for this system there exists an adaptive controller that is what is an adaptive controller there are two pieces to an adaptive control first is a control law u specification for what the actuator has to generate and this depends on x and an estimate of the state theta hat and further there is an update law for theta hand that is a theta hat dot which again depends on the state and theta possibly.

So, this is what constitutes an adaptive controller two pieces, a control law u and a parameter update law theta hat. So, suppose that there exists such an adaptive controller for this system and a smooth V function that is a which is essentially Lyapunov function smooth V which takes the state and the parameters, it is readily unbounded in x and theta tilde.

Such that if I take V dot, so what is this this quantity is just, this quantity is just V dot this quantity is just V dot I am just taking because V is a function of x and theta tilde. So, first I take partial with respect to x then I multiply it with x dot because x dot now contains the control this gets substituted here. So, this is the closed loop system this gets substituted here.

So, del V del x f x plus capital F x theta plus g x alpha. So, this is del V del x x dot plus del V del theta tilde theta tilde dot. So, let us see we have to be careful here, we have to define let us see let me see if we have done this correctly. I think I think this is this should be actually, this should be theta tilde dot this is not theta hat dot but a theta tilde dot it does not matter because I mean we usually are defining theta tilde using theta hat.

So, theta tilde is defined as theta minus theta hat typically. So, theta tilde dot is equal to minus theta hat dot because theta is assumed to be a constant. So, basically this does not change anything I mean if you, I mean it just change the sign but that is important because that is the sign used here. So, this is theta tilde dot not theta hat dot.

So, it is the derivative of the parameter error but remember the side cannot depend on theta tilde because theta delta is unknown we keep that in mind but then this theta tilde dot is just minus theta hat dot. So, if you know theta tilde dot you know minus theta hat dot also. So, this is just the derivative of V along the closed loop trajectories.

So, V is a function of x and theta tilde. So, partial of V with respect to x times x dot and partial of V with respect to theta tilde times theta tilde dot and the assumption is that for this correspond, for this particular V, V dot along the system trajectories is less than equal to minus W x theta tilde which is a negative semi definite function. So, minus of W is negative semi definite.

So, this is the assumption. So, this is the assumption we start with this kind of a system and we have an adaptive controller and a corresponding V. And the corresponding V such that V dot turns out to be less than equal to a negative definite function. So, that is the idea then we go on to add an integrator but before we do that I wanted to compare this with our sorry, our system here.

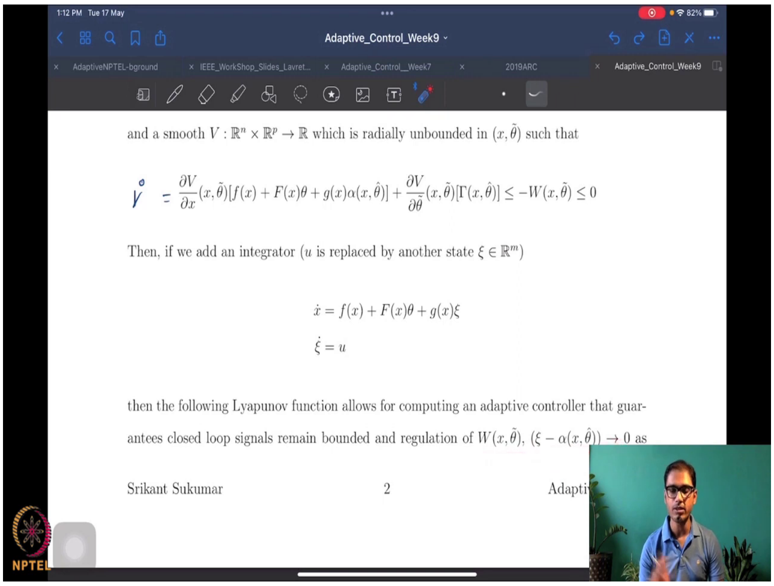
If you look at this guy the first piece where you have this system just look at this, just look at this system and if you think of x 2 as the control just like we did in backstepping if you think of x 2 as the control then you do have this control law which contains the theta hat and an update law with the corresponding V such that V dot is negative semi-definite. So, we exactly have this.

So, what is the situation then we our first system was is of the form this in fact I have written it here. Where this is actually the control and this is the control law which is a function of x1 and theta hat and of course there is an update law, which is theta hat dot there is a V function which is this and V dot equals or less than equals minus W would also depend on theta hat but the point is it is negative semi-definite. The key point to remember is that it is negative semi-definite.

So, we exactly have the same setup already available for the scalar state case, scalar state scalar control case. So, we are exactly generalizing that setup we are exactly generalizing that particular setup. So, we assume that we have now a vector system we have an adaptive controller for this vector system such that there also exists a smooth V such that V dot is negative semi definite.

So, now if we add an integrator which is essentially what is been done here, if you look at this we think of this as the control but actually you have an additional state x 2 dot is u essentially we have an integrator. So, that is what we say now if we add an integrator. So, the control here.

(Refer Slide Time: 18:33)



$$\implies \dot{V} < 0 \quad \forall k_1, k_2 > \frac{1}{2}$$

which from the Lyapunov theorems proves that $x_1 \to 0$ and $x_2 \to x_{2d}$. Now, $x_{2d} = -k_1 x_1 - \theta f(x_1)$ and from the facts that $x_1 \to 0$ and $f(0) = 0$, we have that x_{2d} and hence $x_2 \to 0$.

This completes the proof.

2.2 Unknown Parameter Case

First Control:

$$x_{2d} = -k_1 x_1 - \hat{\theta} f(x_1)$$

$$\text{if } x_2 = x_{2d}$$

$$\dot{x}_1 = -k_1 x_1 - \tilde{\theta} f(x_1), \quad \tilde{\theta} = \theta - \hat{\theta}$$

$$\dot{\hat{\theta}} = \gamma x_1 f(x_1)$$

$$\implies \dot{V}_1 = -k_1 x_1^2 \leq 0$$

Lyapunov Function:

$$V_2 = \frac{1}{2}(x_2 - x_{2d})^2$$

$$\dot{V}_2 = (x_2 - x_{2d})(\dot{x}_2 - \dot{x}_{2d})$$

Choose $\dot{x}_2 = u = \dot{x}_{2d} - k_2(x_2 - x_{2d})$

$$\begin{aligned}
\dot{x}_{2d} &= -k_1 \dot{x}_1 - \gamma x_1 f^2(x_1) - \hat{\theta}\dot{f}(x_1) \\
&= -k_1(x_2 + \theta f(x_1)) - \gamma x_1 f^2(x_1) - \hat{\theta}\dot{f}(x_1) \\
&= -(k_1 + \hat{\theta}\frac{\partial f}{\partial x_1})\dot{x}_1 - \gamma x_1 f^2(x_1) \\
&= -(k_1 + \hat{\theta}\frac{\partial f}{\partial x_1})(x_2 + \theta f(x_1)) - \gamma x_1 f^2(x_1)
\end{aligned}$$

Assume that there exists an adaptive controller,

$$u = \alpha(x, \hat{\theta})$$

$$\dot{\tilde{\theta}} = \Gamma(x, \hat{\theta})$$

and a smooth $V : \mathbb{R}^n \times \mathbb{R}^p \to \mathbb{R}$ which is radially unbounded in $(x, \tilde{\theta})$ such that

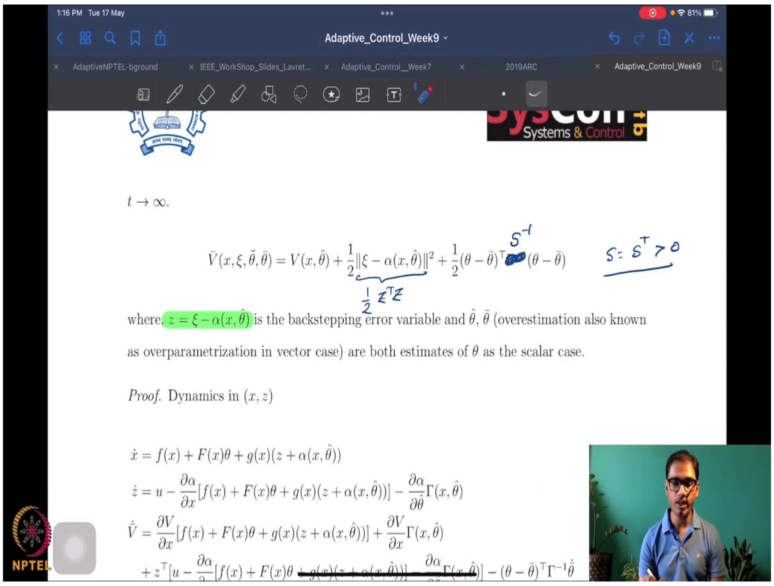
$$\frac{\partial V}{\partial x}(x, \tilde{\theta})[f(x) + F(x)\theta + g(x)\alpha(x, \hat{\theta})] + \frac{\partial V}{\partial \tilde{\theta}}(x, \tilde{\theta})[\Gamma(x, \hat{\theta})] \leq -W(x, \tilde{\theta}) \leq 0$$

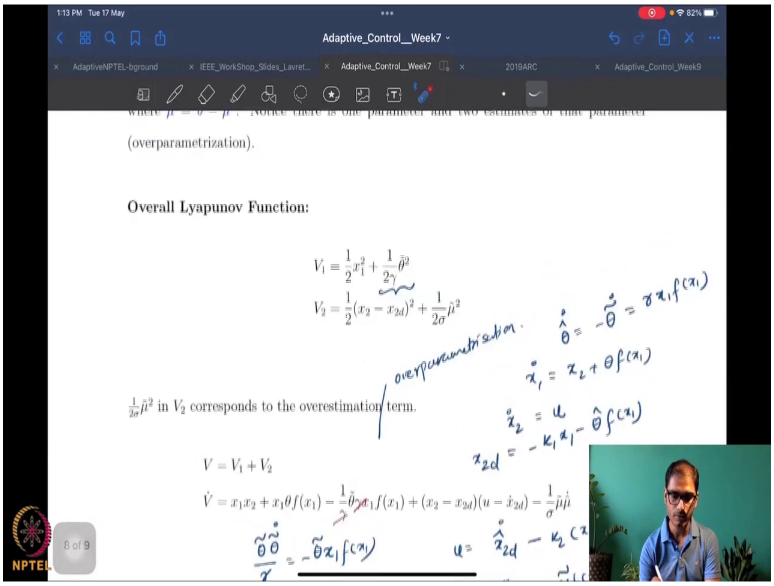
Then, if we add an integrator (u is replaced by another state $\xi \in \mathbb{R}^m$)

$$\dot{x} = f(x) + F(x)\theta + g(x)\xi$$

$$\dot{\xi} = u$$

then the following Lyapunov function allows for computing an adaptive controller





So, the structure is slightly more general of course but still we do the same thing if we add an integrator that is the control is replaced by this psi and then the psi dot is u. So, the control is at the next level. Then what do we do? So, this is where the integrator backstepping was used earlier and we want to of course generalize it to this vector case.

So, one thing should be evident that this psi is also in R m it has to be the same dimension as the control because otherwise this is not a feasible situation. So, if psi is a different dimension then there can be problems. So, we assume that psi is the same dimension as the u. So, then the claim is that we can still construct a Lyapunov function, which allows us to compute an adaptive controller which is going to guarantee closed loop signals remain bounded and also that somehow the backstepping error and W go to 0 as t goes to infinity.

Because this is the best we can do anyway but even in the scalar case, we prove that W goes to 0. What was W? In this case W is this. So, we prove x1 goes to 0 that is the best we can do. And of course we also prove that the backstepping error goes to 0. So, we prove these two things that is what we want even in this scalar case.

So, that is what we claim here also, we want to claim here at least that this W goes to 0 and the backstepping error which is psi minus the because the psi is the state now and cannot always be made exactly equal to alpha but in steady state it can be. So, psi has to follow alpha and that is the backstepping error and that is also going to be driven to be 0.

So, what is this magical Lyapunov function nothing very magical to be honest. So, we call this V bar now and it has several states it has x and psi, which are now the new states, it has the earlier parameter error theta tilde and it has the new parameter error new parameter theta bar. Remember then we use a mu here of course we used a mu let us see here we used a mu the mu hat. So, here we are using theta bar just a difference in notation. So, it is a function of four quantities now x which was always there, theta tilde which is already there for the one state system, and now we have psi and theta bar corresponding to the integrator, and the new over parameterization.

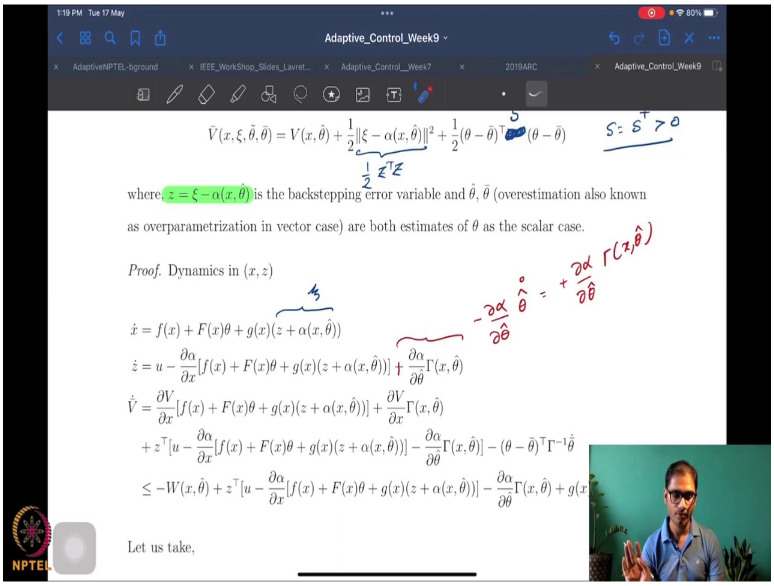
How do we construct it? Take the same V which was just a function of x and theta hat then we add a backstepping error term standard then we add a term corresponding to the new parameter error. So, theta bar is a new parameter estimate. So, I just add a term corresponding to that, now notice that because this is a vector I have taken care that this is written as a norm.

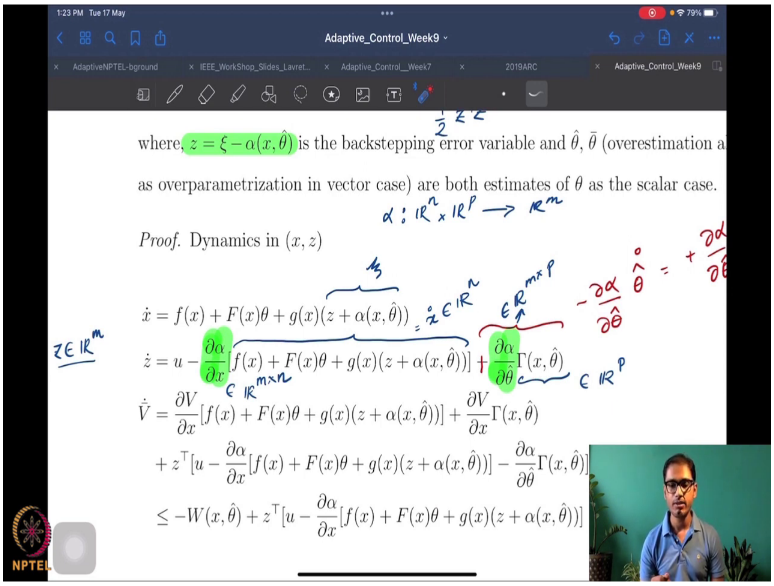
So, this is written as a norm this is no longer just square of psi minus alpha, it is in fact norm squared of psi minus alpha and it is very standard to use the Euclidean norm or the two norms. So, this is denoted as z of course like is mentioned here. So, this term is actually one half z transpose z and this is the Euclidean norm squared and similarly I construct a very similar looking term for the theta minus theta bar that is a new parameter error also just that I add a adaptation gain I always add an adaptation gain this is the adaptation gain.

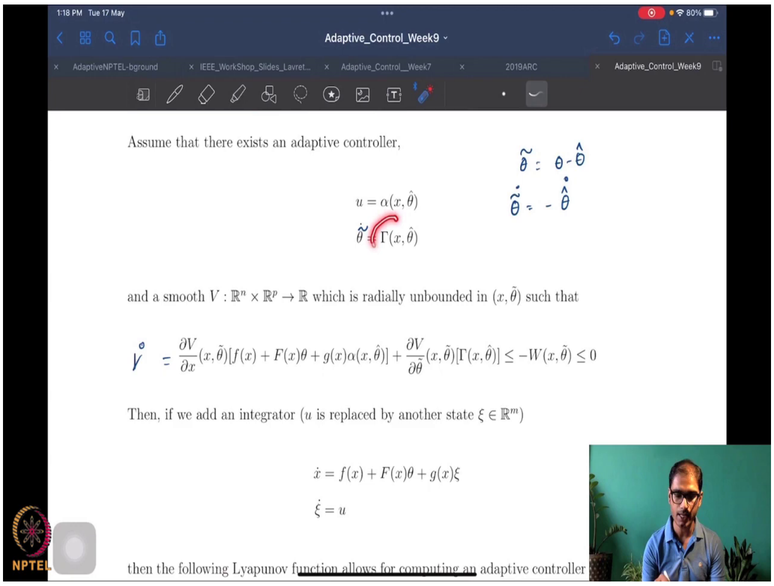
So, let me see unfortunately we have used the same gamma here this is not the same let us be clear on that this is like, let me call it an S inverse. So, S is some constant positive definite matrix. So, this is not the same as that gamma that gamma was the this gamma is actually the update law for the parameter theta hat. This is the different parameter, different matrix S and this S is simply like you all know is the adaptation gain it actually controls how fast or slow your adaptation will happen.

So, now that we have this sort of a new kind of Lyapunov function for this integrator system with vector states. So, therefore we have norms appearing here we have transposes appearing here all we are going to of course take the derivative I mean our claim is that this is a good Lyapunov function, this is our claim. Now of course we want to verify this claim. So, let us see let us see.

(Refer Slide Time: 24:02)







So, the dynamics first we want to do the first thing we want to do is write the dynamics in the new variables, which is x and z now because we introduce the backstepping variable it is not x and psi but it is a x and z. So, we just do that. So, first. So, x dot is f x plus F x theta plus g x times a u and u can be written as, what I am sorry this is not u, this is in fact I am sorry, this is my bad, it is not u but this is psi. So, this is what is the dynamics.

So, and psi can be written as z plus alpha that is what I do and what is z dot z dot is just the derivative of this which is psi dot which is u and then minus alpha dot then I have minus of alpha dot. So, all of this is minus of alpha dot alpha is a function of two variables x and theta hat. So, first I take the derivative with respect to x. So, del alpha del x times n x dot and x dot is just plugged from here.

So, this guy is right here and then minus del alpha del theta hat times theta hat dot times theta hat dot. So, what is theta hat dot. So, this where we have to be careful. So, this is actually now let me be careful here this term should be minus del alpha del theta hat theta hat dot remember alpha cannot depend on theta tilde and because it is was the first stage control it can depend only on theta hat because theta tilde is unknown.

So, and remember from the previous derivation theta hat dot is actually equal to minus theta tilde dot. So, it is equal to minus gamma minus gamma. So, I have to be careful it is actually going to be equal to plus del alpha del theta hat gamma x theta hat there is a plus sign and not a minus sign. So, I need to be careful here this is actually going to be a plus sign this is actually going to be a plus sign I hope that is clear.

So, what have I done I have simply written the dynamics it is just a lot of bookkeeping again I have simply written the dynamics in terms of the new variables x and z this is exactly what I was doing for the scalar case also. All I am doing here is being a little bit more careful because I have vectors that are involved and that is it. So, here I have x dot which was f x plus cap f x theta plus g x times psi and psi is just replaced in terms of the backstepping error variable as z plus alpha, then I have z dot which is psi dot minus alpha dot.

So, when I say dot it is a derivative with respect to time. So, I take psi dot which is u and minus alpha dot. So, there are two terms del alpha del x minus del alpha del x x dot minus del alpha del theta hat theta hat dot. So, there are two terms and then the first term is therefore del alpha del x and x dot which is substituted from here and then I have del alpha del theta hat theta hat dot which brings in a negative gamma.

So, this becomes a positive. So, I have fixed this sign. So, when I write it in this form again remember that what is the dimension of z. So, it is important that we remember also said, so x is in. So, x dot belongs or well I will write it here. So, z x is in R n of course I do not need to write it but z is the same dimension as the control. So, it is in R m. So, this is R m now what is the dimension of this quantity? This is R n.

So, if I look at thing in the bracket this is simply x dot. So, this is equal to x dot and this of course belongs to R n I need to be careful here. Now so what is the dimension of del alpha del x, therefore what does what has to be the dimension of del alpha del x, it has to be let us… has to be R m cross n to be consistent with this expression because the right hand side has to be R n. And this is in R n, therefore this has to be an m cross n matrix.

Now how does this happen similarly, similarly before we go on, gamma here is from theta tilde dot or theta hat dot whatever and theta was in R p, R p. There are p unknown parameters. Therefore, this guy has to belong to R m cross p. So, how does this happen? So, look at what is alpha, alpha is a function of x and theta hat what do I know about dimension of alpha I know that alpha if I may write it carefully alpha is in fact a map from x states.

So, R n and p states which is R p. So, that is theta hat states to what it is the same dimension as psi. So, it is m R m. So, now if I take partial of alpha with respect to only these states, it is obvious that it will give I will get m by n matrix when I take partial of an m vector with respect to only n of the states I will get an m by n matrix similarly if I take the partial of m vector with respect to p states then I will get m by p matrix. So, this is all consistent.

So, just keep this in mind that all these partials are now Jacobians, what we call Jacobians and they are all matrices. So, what is it that we did today, we sort of started to generalize the adaptive integrator backstepping that we looked in looked at in week number 7 to the vector case. And we are sort of trying to understand the differences when there is the vector case. So, the norms appear the transposes appear and then there is the Jacobian and all these notions.

So, we have to do very careful bookkeeping but as of now I hope you have seen that the methods are not in distinctly different just because the states become vectors or something like that. So, of course we continue working on this generalization of adaptive integrator back stepping to the vector case in the subsequent session also. So, this is where we stop and see you again next time. Thanks.