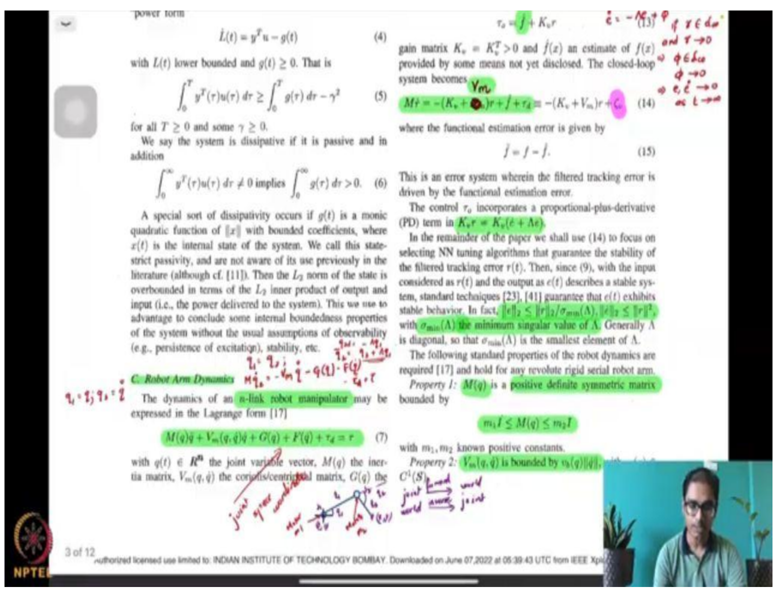
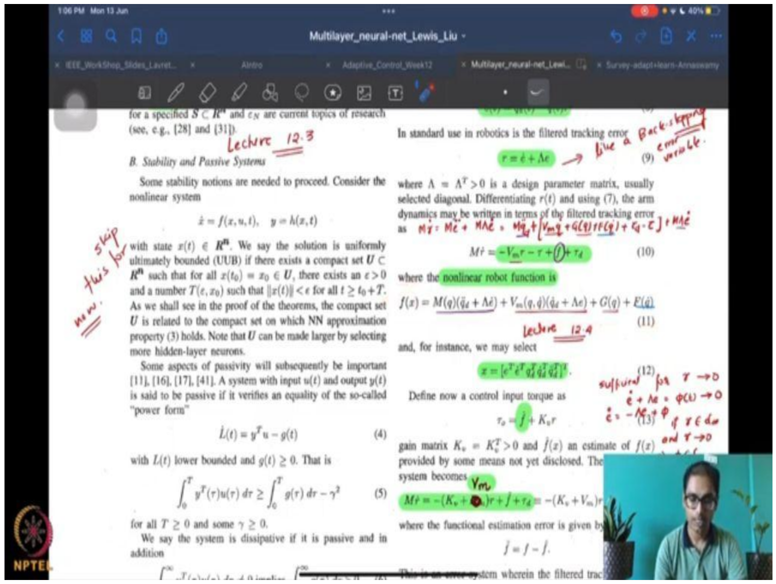
Nonlinear Adaptive Control
Professor Srikant Sukumar
Systems and Control
Indian Institute of Technology, Bombay
Week 12
Lecture No: 70
Real Time Neural Network Based Control of a Robotic Manipulator (Part 4)

Hello everyone. Welcome to yet another session of our NPTEL on Nonlinear Adaptive Control, I am Srikant Sukumar from systems and control, IIT Bombay. So, I warmly welcome you to this last week of lectures on this nonlinear adaptive control. And we have already completed four lectures in this set of lectures this week. And we started off in this week looking at connections between adaptive control and learning. And we are now focusing on a specific problem of looking at a multi-layer neural network, which is being tuned, which is going to be tuned using an adaptive controller.

And this neural network, this three-layer neural network will be specific is being used to design approximation-based controller for a robotic manipulator. So, we do hope that the algorithms that we have been looking at will help you design robust adaptive control algorithms for systems like the aircraft, drone, spacecraft et-cetera that you see in the background.

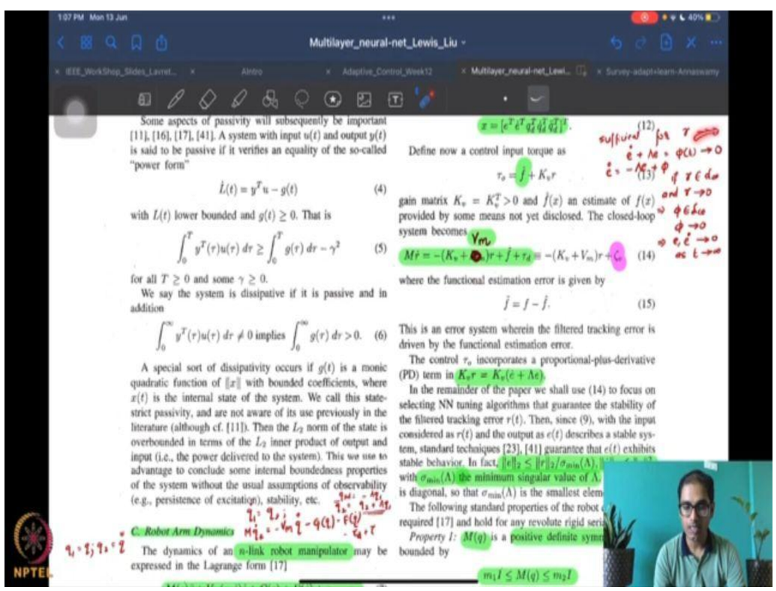
(Refer Slide Time: 01:50)

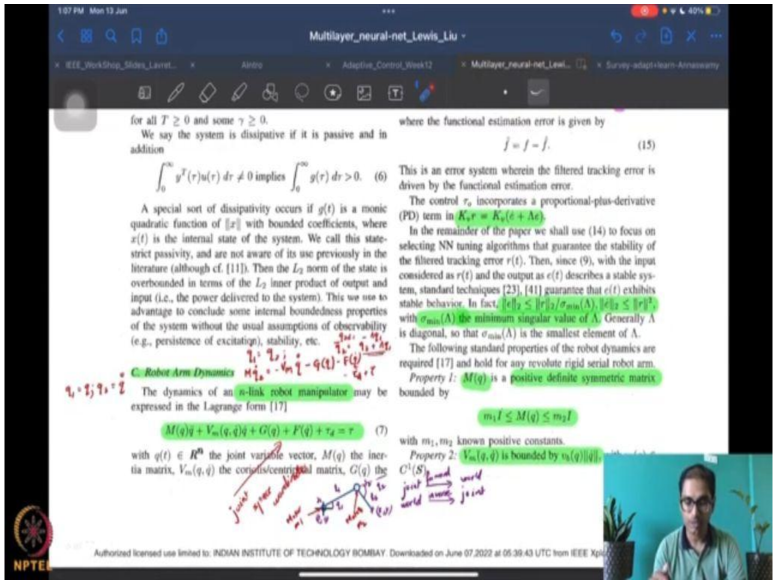


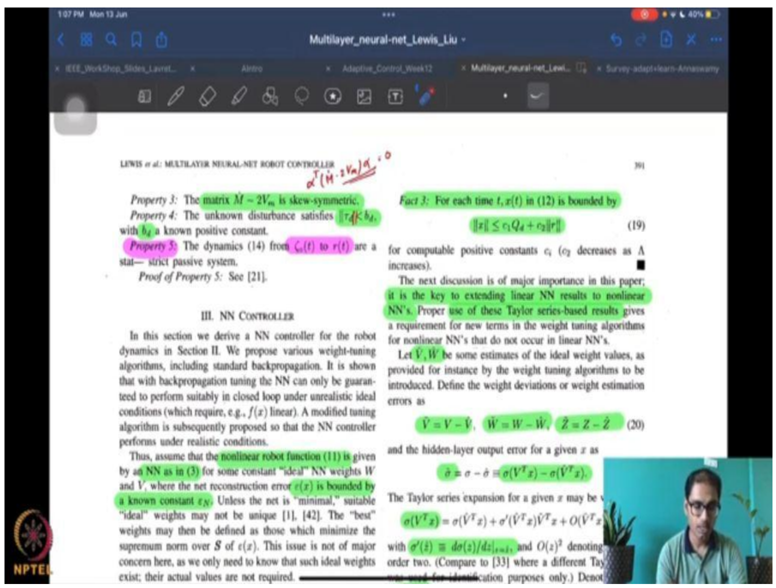


Now, there we there, until last time was basically, we had already seen what the robot dynamics consists of in the joint space coordinates. And then we sort of also you define this nice backstepping variable, which is used for analysis. And this is in terms of, of course, the tracking error. And the entire dynamics was, of course, written in terms of this backstepping error variable. And these dynamics is what contains the nonlinear robot function. And this is essentially what the neural network is used to approximate.

(Refer Slide Time: 02:38)



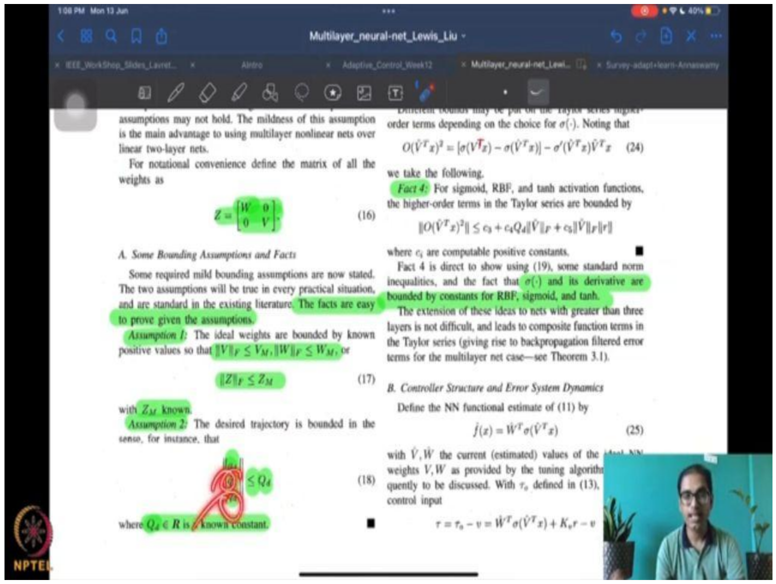


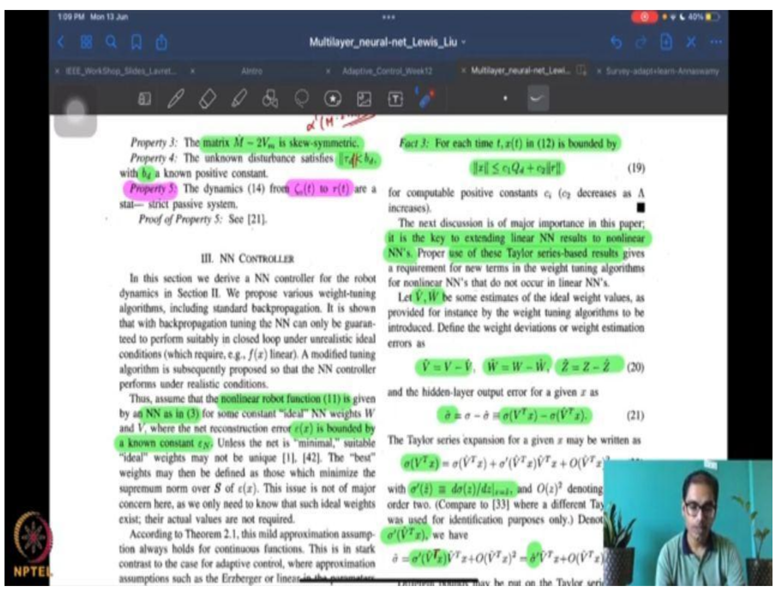


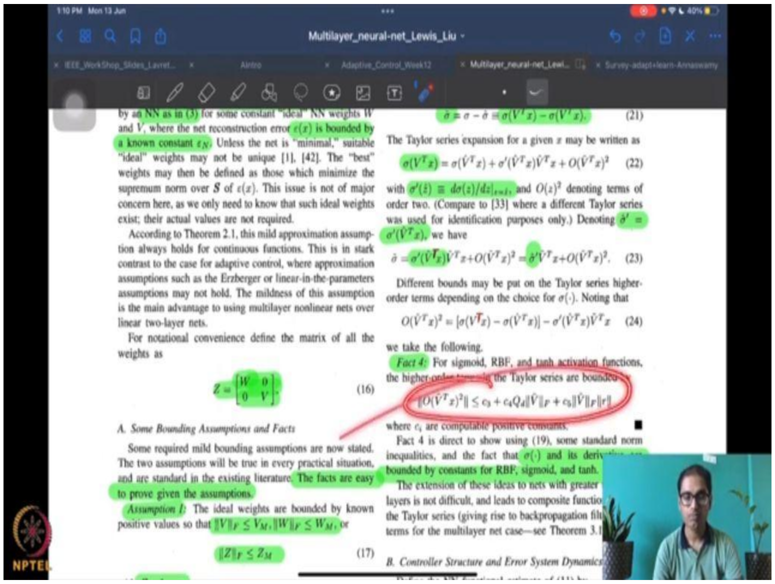
This function is of course, depending on some variables like e, e dot, qd, qe dot, qd double dot, which are collected together in this x, and using estimation of this function using a neural network, which we will see subsequently, we define a controller. And, of course, it is not difficult to see, I mean we are not we are not doing that here. It is not difficult to see that if f tilde and tau d are in fact 0. If these are not there, then this is a nice stable system, asymptotically stable system in r. And you can show that r goes to 0, which means that you can also show that e and e dot go to zero.

We of course showed some nice relationships using this backstepping error variable design. We show that that that basically the e and e dot are bounded by the backstabbing error variable norm as you would expect. Then, we have talked about a few properties like the positive definiteness of M, the boundedness of VM dot, skew symmetry of M dot minus 2 VM. And the fact that the disturbances are bounded, we did not look at the passivity properties.

(Refer Slide Time: 04:00)







So, the idea is to estimate this nonlinear robot function, this was the sort of plan using the neural network, the three stage or the three-layer neural network. We clubbed the unknowns weights, unknown weights W and V inside this new variable Z. And then of course, we made a few assumptions, the first one being that the ideal value of weights are bounded, which is a very reasonable assumption, we do that even in projection based adaptive control design. Then, we also assume that the trajectories are bounded, this again trajectory and its derivatives are bounded.

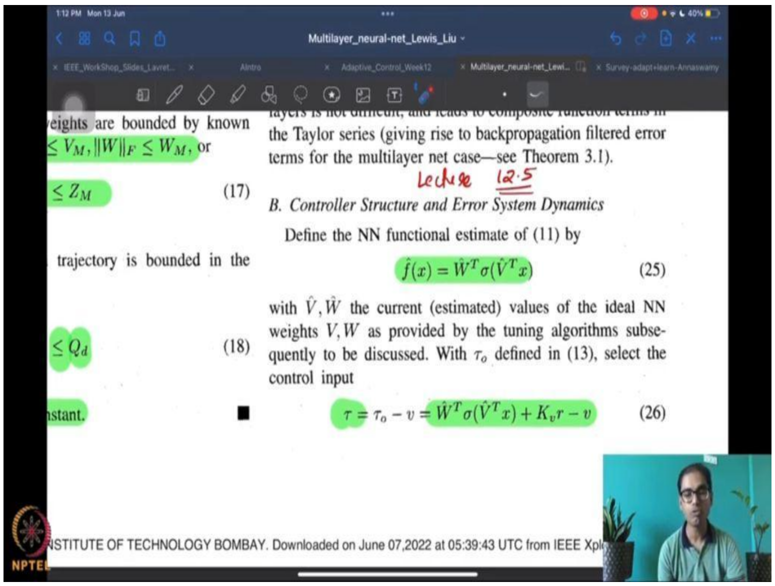
This is again another assumption we did make while doing our standard adaptive control. And finally, using this we could come up with the fact that the norm of x that is where x is this e, e dot, qd, qd dot and qd double dot is also bounded by qd and the norm of r. Again, not difficult to verify once you have made these assumptions.

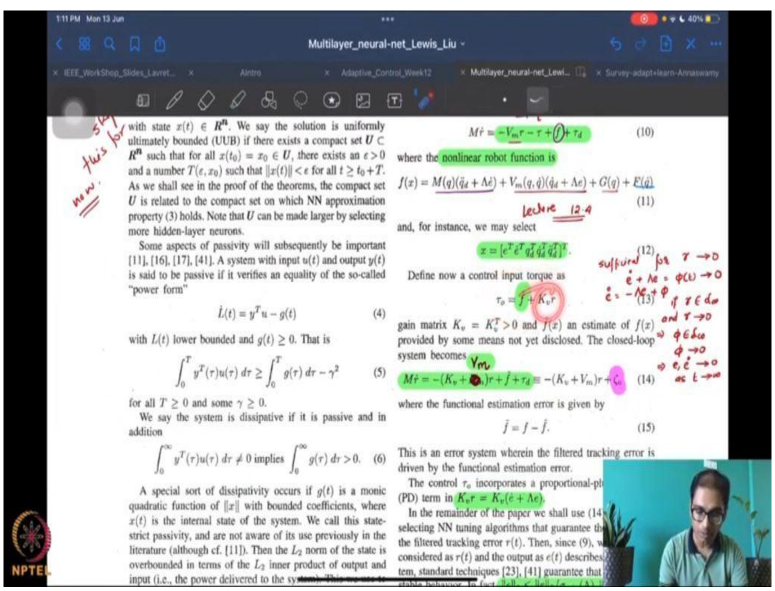
So, then we of course, talk about linearizing the neural network. And because what happens is that you have this nonlinear regressor parameter structure, which you cannot define adaptive laws for. And even if you do define some kind of adaptation law, you will not be able to prove any stability, which is why we, we will we looked at some Taylor series approximations.

So, V hat and W hat were the estimates and correspondingly we define the tildes, which are the estimation errors, and also the error in the activation function values. And using that we sort of gave an estimate for this this sigma tilde, so, sigma tilde had this kind of an estimate. And further we also state the fact which said that this square term is bounded.

So, bound on the square term is of course, obtained using this kind of an expression. This this expression is simply obtained from 23. So, from 23, you obtain this, and from here you can actually prove this kind of a fact. So, this fact essentially says that for sigmoidal, radial basis function, and tan hyperbolic functions. So, these are the activation function examples we looked at, you do have this kind of nice relationship.

(Refer Slide Time: 06:38)





Now, this is where we start today's lecture. So, so, let us mark it first lecture number 12.5, which is the fifth lecture of this last week. So, here, if you see the way we define a f hat is again using just the estimates, we replace sorry. We replace W by its estimate W cap, and V by its estimate V cap and that is it. So, V cap and W cap are the current estimated value. So, this is what is your neural network functional estimate. So, with this, you have this tau 0 which was the control we defined, and the (con) the true control.

So, until now we had only the tau 0 if you remember, which was just these two terms, contained the functional estimate and a damping nice pd type of term, proportional derivative type of term. But now, we also add some kind of sort of a robustification input v, and this is what we look at what this v can be later on. But for now, we introduced this robust certification sort of input v, so, this becomes your true control tau.

(Refer Slide Time: 08:06)

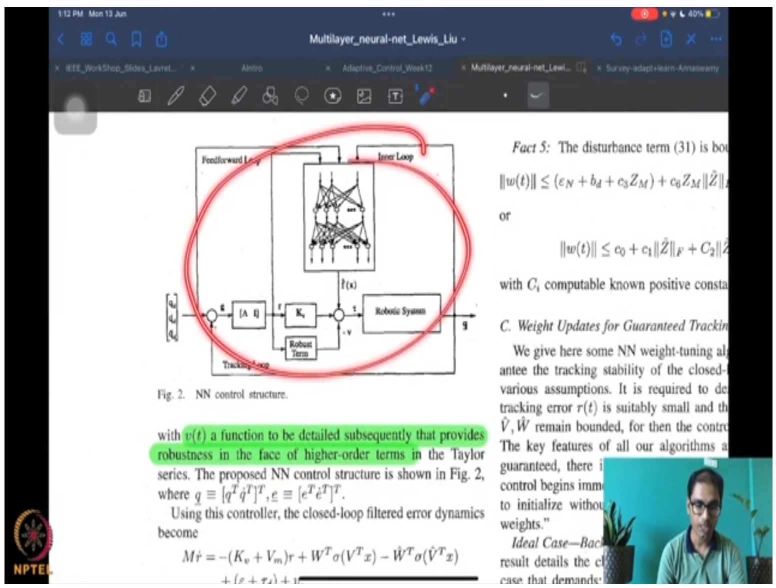


Fig. 2. NN control structure.

with $v(t)$ a function to be detailed subsequently that provides robustness in the face of higher-order terms in the Taylor series. The proposed NN control structure is shown in Fig. 2, where $\underline{q} \equiv [q^T \dot{q}^T]^T, \underline{e} \equiv [e^T \dot{e}^T]^T$.

Using this controller, the closed-loop filtered error dynamics become

$$M\dot{r} = -(K_v + V_m)r + W^T\sigma(V^T x) - \underbrace{\hat{W}^T\sigma(\hat{V}^T x)}_{= \hat{W}^T\hat{\sigma}} + (\varepsilon + \tau_d) + v.$$

Adding and subtracting $W^T\hat{\sigma}$ yields

$$M\dot{r} = -(K_v + V_m)r + \tilde{W}^T\hat{\sigma} + W^T\tilde{\sigma} + (\varepsilon + \tau_d) + v$$

with $\hat{\sigma}$ and σ defined in (21). Adding and subtracting now $\hat{W}^T\tilde{\sigma}$ yields

$$M\dot{r} = -(K_v + V_m)r + \tilde{W}^T\hat{\sigma} + \hat{W}^T\tilde{\sigma} + \tilde{W}^T\tilde{\sigma} + (\varepsilon + \tau_d) + v. \quad (27)$$

The key step is the use now of the Taylor series approximation (23) for $\tilde{\sigma}$, according to which the closed-loop error system is

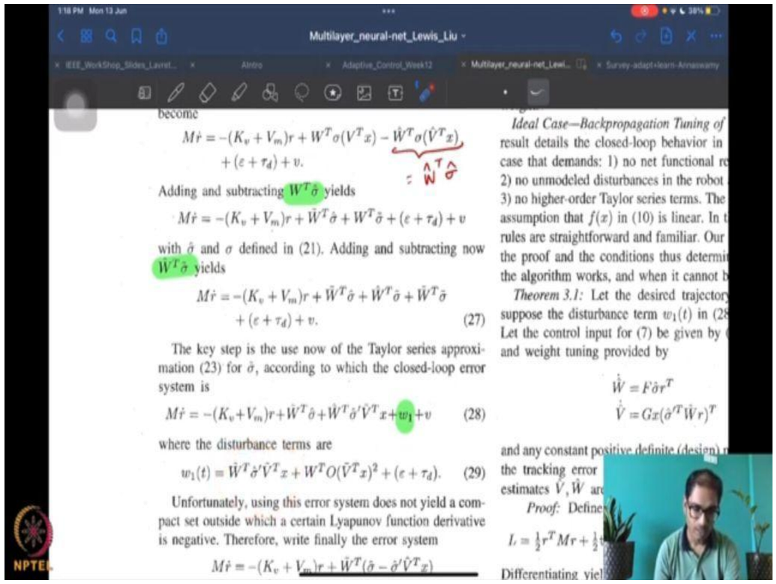
$$M\dot{r} = -(K_v + V_m)r + \tilde{W}^T\hat{\sigma} + \hat{W}^T\hat{\sigma}'\tilde{V}^T x + w_1 + v \quad (28)$$

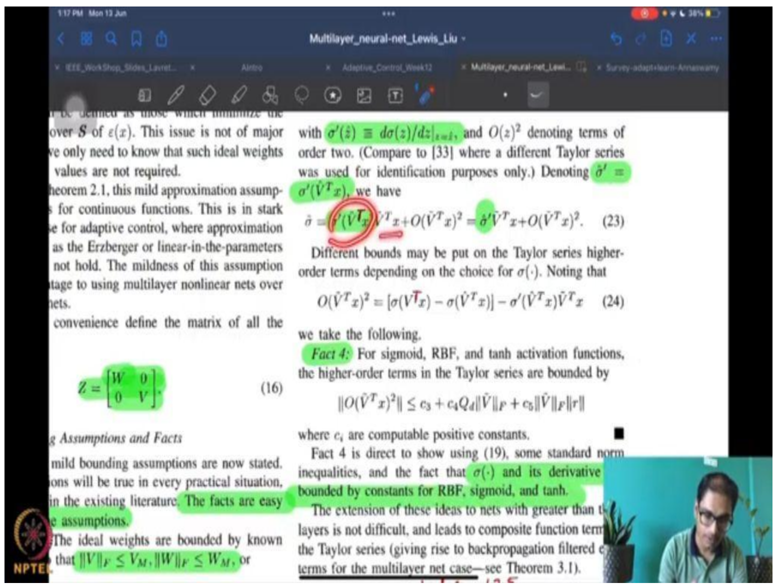
where the disturbance terms are

tracking error $r(t)$ is suitably small and th
$\hat{V}, \hat{W}$ remain bounded, for then the contr
The key features of all our algorithms a
guaranteed, there is no off-line learning
control begins immediately, and the NN we
to initialize without the requirement for
weights."

Ideal Case—Backpropagation Tuning of result details the closed-loop behavior in case that demands: 1) no net functional re 2) no unmodeled disturbances in the robot 3) no higher-order Taylor series terms. The assumption that $f(x)$ in (10) is linear. In t rules are straightforward and familiar. Our the proof and the conditions thus determi the algorithm works, and when it cannot b

Theorem 3.1: Let the desired trajector suppose the disturbance term $w_1(t)$ in (28 Let the control inp and weight tuning





So, so v is basically like I said is a function to be detailed subsequently that provides robustness in the face of higher order terms in the Taylor series. So, so this is sort of the block diagram for this whole implementation, you can verify that this block diagram matches with the system equations, not too complicated. And then of course, you have defined q under bar and e under bar as this e, e dot and q, q dot sort of vectors. So, once you have this modified control tau, we started with the tau 0, and we added the robustification, or we subtracted robustification control v, so, you have the tau.

And you substitute for that and you get this sort of a closed loop system. You you already had this term, this is not a new term. And then this was the function and the true value of the function if you mean, where you have the ideal value of the weight W and V. And then you have the estimate W hat and V hat, and even the ideal value is somehow epsilon off from the true function. So, the function is approximated as W transpose, sigma V transpose x, plus an epsilon. And you have this estimated value, then you have the disturbance, and then you have a this robustification term.

So, then we do a little bit of manipulations. What is this manipulation? First, we add and subtract a W transpose sigma hat, so that sort of gets combined. So, this you notice is W transpose sigma hat, so, we already defined this notation. So, this is actually where W hat transpose sigma hat in the notation we defined. So, if you combine so, so if you add and subtract W transpose sigma hat, one of it combines with this, the other one combined with this guy. So, the combination with this will will yield a W tilde transpose sigma hat, because this is W hat transpose sigma hat, and you have W transpose sigma hat. So, if you subtract one from the other, you get W tilde transpose sigma hat.

And then this guy combines with this again to give you a W transpose sigma tilde, and so you get two terms by adding and subtracting the W transpose sigma. So, then of course, these three terms remain exactly as it is and these three terms show no change, great great. So, now again we do another layer of manipulation and we now.

So, first we added and subtracted W transpose sigma hat, and now we add and subtract W hat transpose sigma tilde. Why? Because this term and this term is what we are looking to sort of deal with this. So, if you look at this term and this term together, these two will combine to give you a W tilde transpose sigma tilde, and you will of course be left with the W hat transpose sigma tilde.

So, that I mean that term remains as it is here and these two combined to give you this term. So, now you have two terms here and you now have three terms, because of this addition and subtraction, it is just a rewriting of these terms, if you remember. Now, we start using all Taylor series approximations that we did. So, this Taylor series approximation is for this V tilde term, sorry, it is for the sigma tilde term. So, this is where we use the Taylor series approximation. So, so if you look at this guy, it is retained as it is, and then you have the W hat transpose sigma tilde type term.

And then you write some terms as what we call the disturbance term or what the authors call as the disturbance term. So, here if you notice in the sigma tilde approximation, you had several terms, so, sigma tilde looked like this. I mean, I mean, of course you have the this term, which had some bound, but then the key approximation was something like this.

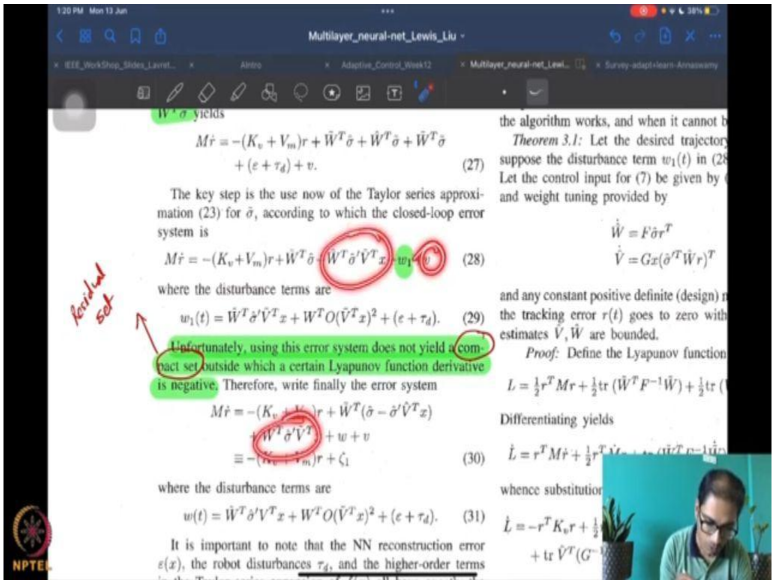
So, this this did we define any further notation? Let me see. This was defined as sigma hat transpose, I believe this is called sigma hat transpose this term, and then you have V tilde transpose x, so, that is what is written here. The sigma tilde here is written as sigma hat transpose, V tilde transpose x.

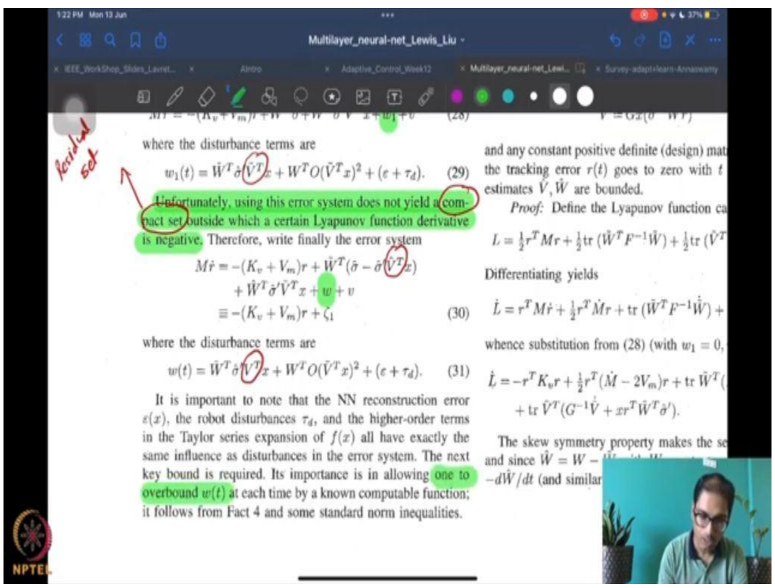
And then everything else is sort of written as a disturbance including this guy. So, this also has this, this comes this comes to this is the approximation, and then you have the second order terms, which combined to give you this. And then you have epsilon plus tau d and the V is kept as it is, so, if you notice a lot of terms have been.

So, the second order terms of corresponding to this guy and this guy combined to give you this, and the first order term of this guy is written here. And the first order term of this guy is also clubbed into the disturbance. So, rather interesting, just I would say more or less arbitrary choice of what should be disturbance.

I mean, here this w1, I mean it makes sense for these to be disturbance. And of course, also this because if you think of the second order terms to be small enough, fair enough, this is also a disturbance. But, not thinking of this term as the disturbance is not very well justified. So, anyway, so so we, so they of course make some changes.

(Refer Slide Time: 14:57)





So, unfortunately using this error system does not yield compact set outside which you have a negative Lyapunov function derivative, so, which is what you needed. You need the Lyapunov function derivative to be negative, at least outside a compact set. So, what is this compact set? This is what is the residual set.

So, this is what what they call this compact set is what is the how we have talked about the residual set, and this is what is the residual set. So, so then they of course, rewrite things a little bit, I guess. What is it that they do? So, this term is the same, then this term is again the same, this term is again the same. Then, this guy is brought back in, and this guy is not put into the disturbance, so, so this term, I believe is brought back in.

So, because, because, so this term is the same here, then you have W tilde transpose sigma hat which is the same here, and then this W tilde transpose sigma hat prime V transpose x is brought back. So, this this term is put back, which is what I also said does not make sense to keep it out, and then this term remains as it is, and then a V.

And now the w1 becomes a W, so this makes a little bit more sense. And here, I believe what is left? Let us see carefully. I see the whole term is not brought back, let us be careful, this what is brought back is only the. So, here you had the W tilde transpose sigma hat prime V tilde transpose.

And what is brought back is only the V hat transpose term, and what is left is the V transpose term. So, because V tilde transpose is V minus V hat, so, the V hat term is brought in here with the negative sign and the v term is left, interesting. I guess the purpose of this is a primarily to make sure the Lyapunov analysis goes through. And of course, if W tildes are small, then this is sort of justified. So, if W tilde is smaller than keeping this as disturbance is sort of justified in some sense. And this is anyway a second order term. So, so these three being here was never big a question.

But this this being here, and this being here does pose some questions, but then you have to sort of think of W tilde having some kind of a nice bound. And then you can say some nice things because x is bounded and all these quantities are also possibly bounded. So of course, then we need to get a bound on w, one to over bound w, so this is what is shown in this next fact.

(Refer Slide Time: 18:33)

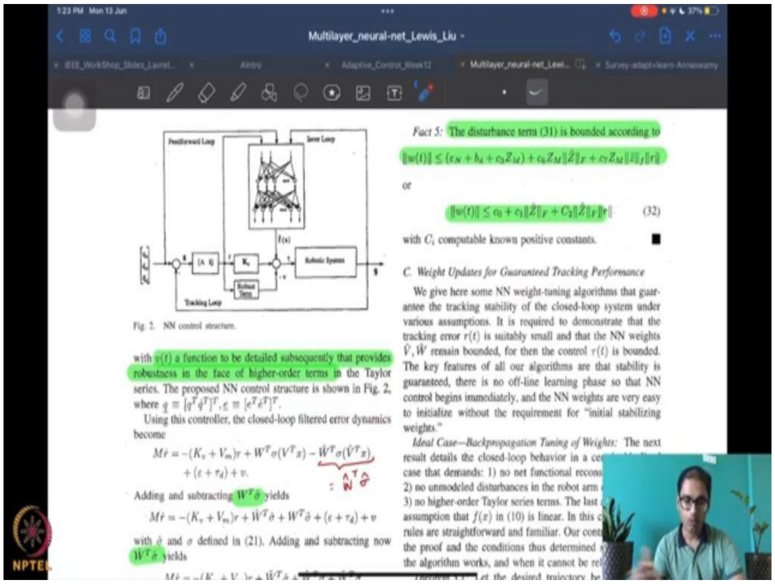

Multilayer_neural-net_Lewis_Liu

Fig. 2. NN control structure.

with $v(t)$ a function to be detailed subsequently that provides robustness in the face of higher-order terms in the Taylor series. The proposed NN control structure is shown in Fig. 2, where $\underline{q} \equiv [q^T \dot{q}^T]^T, \underline{e} \equiv [e^T \dot{e}^T]^T$.

Using this controller, the closed-loop filtered error dynamics become

$$M\dot{r} = -(K_v + V_m)r + W^T\sigma(V^T x) - \hat{W}^T\sigma(\hat{V}^T x) + (\varepsilon + \tau_d) + v.$$

Adding and subtracting $W^T\hat{\sigma}$ yields

$$M\dot{r} = -(K_v + V_m)r + \tilde{W}^T\hat{\sigma} + W^T\tilde{\sigma} + (\varepsilon + \tau_d) + v$$

with $\hat{\sigma}$ and σ defined in (21). Adding and subtracting now $\hat{W}^T\tilde{\sigma}$ yields

Fact 5: The disturbance term (31) is bounded according to

$$\|w(t)\| \le (\varepsilon_N + b_d + c_3 Z_M) + c_6 Z_M \|\tilde{Z}\|_F + c_7 Z_M \|\tilde{Z}\|_F \|r\|$$

or

$$\|w(t)\| \le c_0 + c_1 \|\tilde{Z}\|_F + C_2 \|\tilde{Z}\|_F \|r\| \quad (32)$$

with C_i computable known positive constants. ■

C. Weight Updates for Guaranteed Tracking Performance

We give here some NN weight-tuning algorithms that guarantee the tracking stability of the closed-loop system under various assumptions. It is required to demonstrate that the tracking error $r(t)$ is suitably small and that the NN weights $\hat{V}, \hat{W}$ remain bounded, for then the control $\tau(t)$ is bounded. The key features of all our algorithms are that stability is guaranteed, there is no off-line learning phase so that NN control begins immediately, and the NN weights are very easy to initialize without the requirement for "initial stabilizing weights."

Ideal Case—Backpropagation Tuning of Weights: The next result details the closed-loop behavior in a ce... case that demands: 1) no net functional recons... 2) no unmodeled disturbances in the robot arm ... 3) no higher-order Taylor series terms. The last ... assumption that $f(x)$ in (10) is linear. In this c... rules are straightforward and familiar. Our cont... the proof and the conditions thus determined ... the algorithm works, and when it cannot be rel...

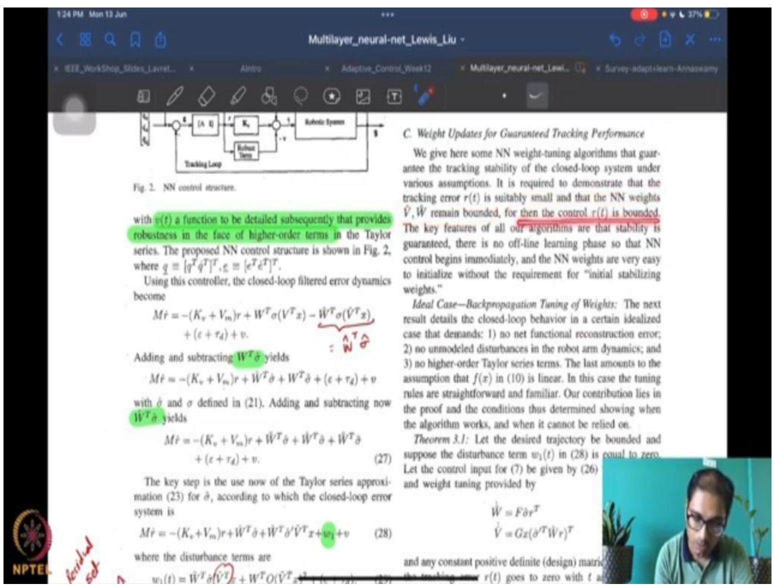

Multilayer_neural-net_Lewis_Liu

Fig. 2. NN control structure.

with $v(t)$ a function to be detailed subsequently that provides robustness in the face of higher-order terms in the Taylor series. The proposed NN control structure is shown in Fig. 2, where $\underline{q} \equiv [q^T \dot{q}^T]^T, \underline{e} \equiv [e^T \dot{e}^T]^T$.

Using this controller, the closed-loop filtered error dynamics become

$$M\dot{r} = -(K_v + V_m)r + W^T\sigma(V^T x) - \hat{W}^T\sigma(\hat{V}^T x) + (\varepsilon + \tau_d) + v.$$

Adding and subtracting $W^T\hat{\sigma}$ yields

$$M\dot{r} = -(K_v + V_m)r + \tilde{W}^T\hat{\sigma} + W^T\tilde{\sigma} + (\varepsilon + \tau_d) + v$$

with $\hat{\sigma}$ and σ defined in (21). Adding and subtracting now $\hat{W}^T\tilde{\sigma}$ yields

$$M\dot{r} = -(K_v + V_m)r + \tilde{W}^T\hat{\sigma} + \hat{W}^T\tilde{\sigma} + \tilde{W}^T\tilde{\sigma} + (\varepsilon + \tau_d) + v. \quad (27)$$

The key step is the use now of the Taylor series approximation (23) for $\tilde{\sigma}$, according to which the closed-loop error system is

$$M\dot{r} = -(K_v+V_m)r+\tilde{W}^T\hat{\sigma}+\hat{W}^T\hat{\sigma}'\tilde{V}^T x+w_1+v \quad (28)$$

where the disturbance terms are

C. Weight Updates for Guaranteed Tracking Performance

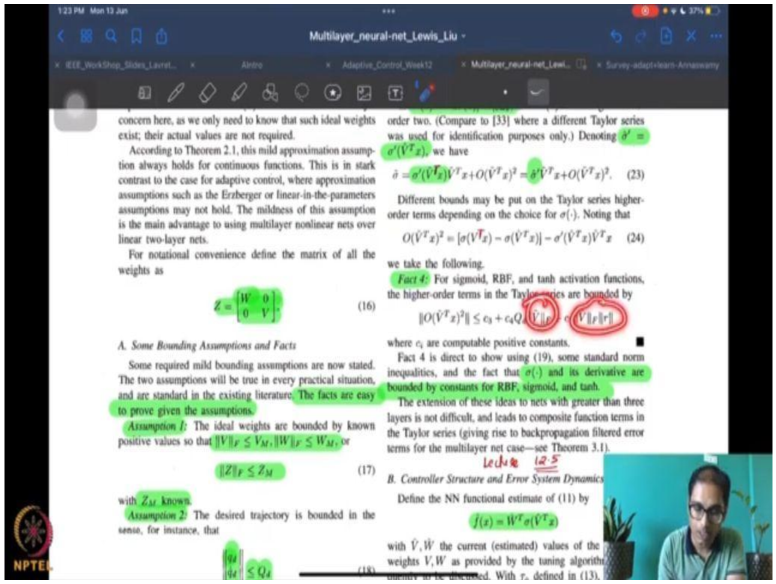
We give here some NN weight-tuning algorithms that guarantee the tracking stability of the closed-loop system under various assumptions. It is required to demonstrate that the tracking error $r(t)$ is suitably small and that the NN weights $\hat{V}, \hat{W}$ remain bounded, for then the control $\tau(t)$ is bounded. The key features of all our algorithms are that stability is guaranteed, there is no off-line learning phase so that NN control begins immediately, and the NN weights are very easy to initialize without the requirement for "initial stabilizing weights."

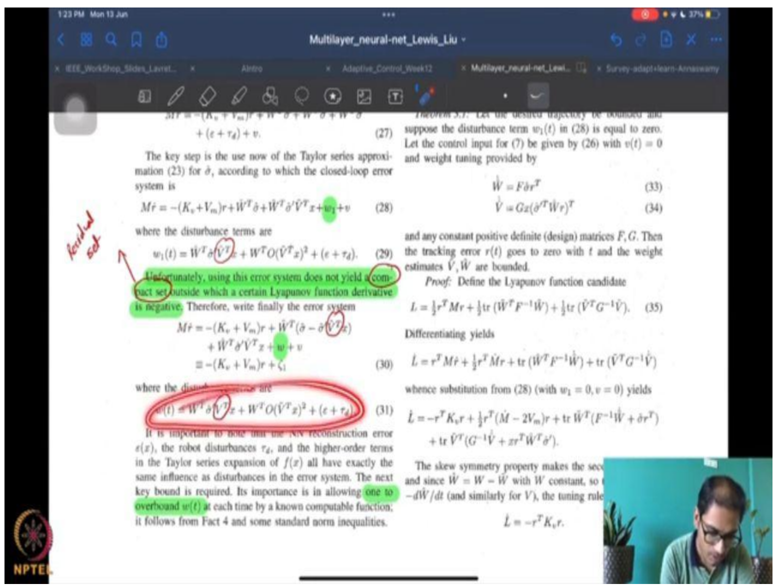
Ideal Case—Backpropagation Tuning of Weights: The next result details the closed-loop behavior in a certain idealized case that demands: 1) no net functional reconstruction error; 2) no unmodeled disturbances in the robot arm dynamics; and 3) no higher-order Taylor series terms. The last amounts to the assumption that $f(x)$ in (10) is linear. In this case the tuning rules are straightforward and familiar. Our contribution lies in the proof and the conditions thus determined showing when the algorithm works, and when it cannot be relied on.

Theorem 3.1: Let the desired trajectory be bounded and suppose the disturbance term $w_1(t)$ in (28) is equal to zero. Let the control input for (7) be given by (26) and weight tuning provided by

$$\dot{\hat{W}} = F\hat{\sigma}r^T$$
$$\dot{\hat{V}} = Gx(\hat{\sigma}'^T\hat{W}r)^T$$

and any constant positive definite (design) matri... $r(t)$ goes to zero with t ...

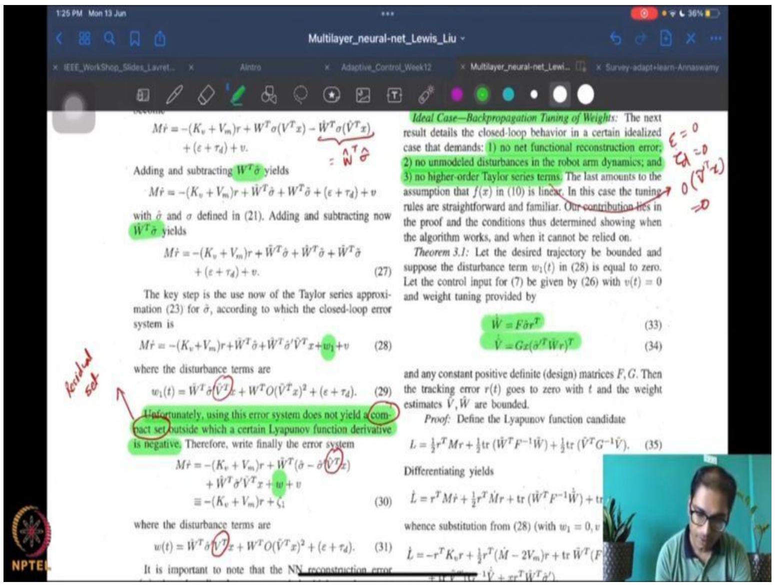




So the disturbance term 31, which is this guy w is bounded in this way. Or, of course, I mean, it is probably nicer similar type of expression here. So, here you have some constant, let us not worry about it. Then, there is dependence on the Z tilde, which is expected, because well, W tilde appears here. Obviously, as we bounded in Z tilde, and then there is bound correspond, there is terms that have multiplication of Z tilde and r. And this also comes I believe from this guy, because this second order term is bounded by V tilde times r. So therefore, the whole thing is bounded by Z tilde times r, norm of Z tilde times r. So, Z tilde is just V tilde and W tilde stacked.

So, this is again another bound with respect to the state and estimate bounds. So, now, once we have all these bound, we have this sort of a closed loop system with whatever we have declared as as some kind of a disturbance. So this is our final system, with with this kind of a disturbance term. And so, what we now want to do is how to do the update propagation. So, this is what is the important thing, we want to see how the updates are propagated. So, it is required to demonstrate that the tracking error is suitably small, and then the neural network weights remain bounded, for only then the control is bounded, so, this is important.

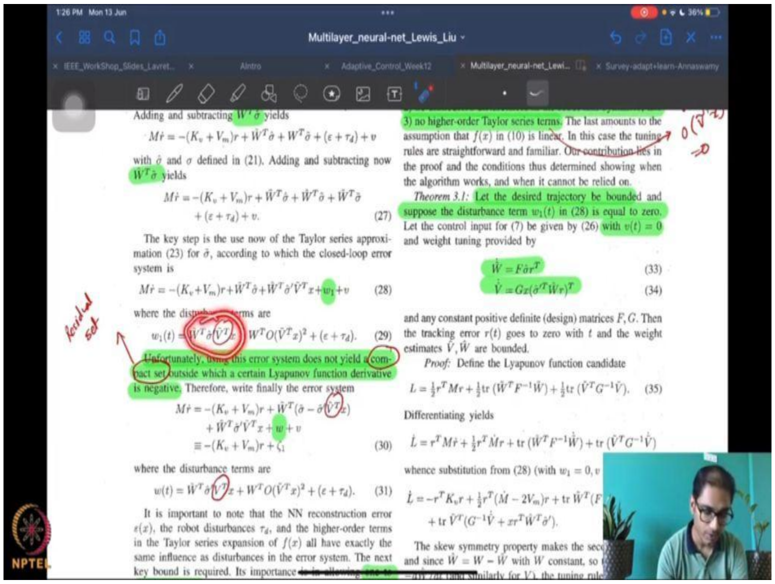
(Refer Slide Time: 20:27)

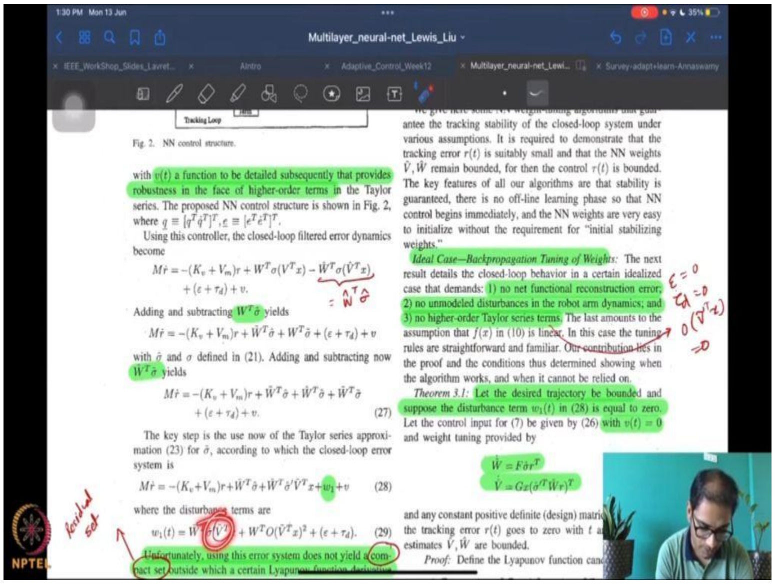


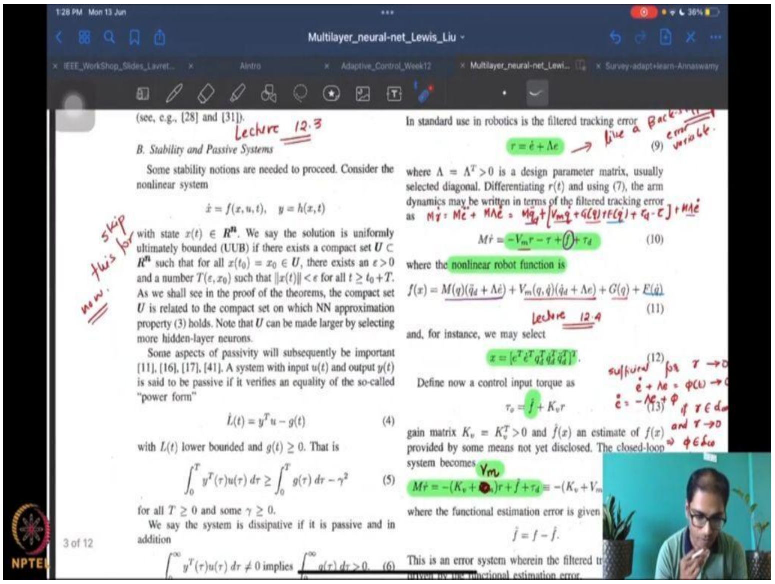
So, if you assume a few things, this is where the ideal case the back-propagation tuning, which is like the basic neural network tuning method. Here there is a lot of idealization. So, the assumption is there is no net functional reconstruction error, this sort of means that epsilon is 0. What is the next one? There is no unmodeled disturbances and no higher order terms. So, essentially, this second one means d is 0, sorry tau d is 0. And the third one means that your unmodeled terms are gone, so, you have whatever, not unmodeled the higher order terms are gone.

So, o V tilde transpose x equal to 0, so, there are no higher order terms. So, if you make these kinds of assumption, so it is as, I mean it is as good as assuming that fx is linear function, as good as assuming that fx is a linear one. Then, of course, things are much more simpler, you go back to your typical adaptive control type domain. So, in that case, this theorem proposes a nice update law, which is this guy and that guy. So, so in fact in fact, this theorem is rather nice, because it looks directly at the w1 disturbance, let the desired trajectory be bounded. So, these are a lot of assumptions actually.

(Refer Slide Time: 21:50)







(see, e.g., [28] and [31]).

B. Stability and Passive Systems

Some stability notions are needed to proceed. Consider the nonlinear system

$$\dot{x} = f(x,u,t), \quad y = h(x,t)$$

with state $x(t) \in R^n$. We say the solution is uniformly ultimately bounded (UUB) if there exists a compact set $U \subset R^n$ such that for all $x(t_0) = x_0 \in U$, there exists an $\varepsilon > 0$ and a number $T(\varepsilon, x_0)$ such that $\|x(t)\| < \varepsilon$ for all $t \geq t_0 + T$. As we shall see in the proof of the theorems, the compact set U is related to the compact set on which NN approximation property (3) holds. Note that U can be made larger by selecting more hidden-layer neurons.

Some aspects of passivity will subsequently be important [11], [16], [17], [41]. A system with input $u(t)$ and output $y(t)$ is said to be passive if it verifies an equality of the so-called "power form"

$$\dot{L}(t) = y^T u - g(t) \tag{4}$$

with $L(t)$ lower bounded and $g(t) \geq 0$. That is

$$\int_0^T y^T(\tau)u(\tau)\,d\tau \geq \int_0^T g(\tau)\,d\tau - \gamma^2 \tag{5}$$

for all $T \geq 0$ and some $\gamma \geq 0$.

We say the system is dissipative if it is passive and in addition

$$\int_0^\infty y^T(\tau)u(\tau)\,d\tau \neq 0 \text{ implies } \int_0^\infty g(\tau)\,d\tau > 0. \tag{6}$$

In standard use in robotics is the filtered tracking error

$$r = \dot{e} + \Lambda e \tag{9}$$

where $\Lambda = \Lambda^T > 0$ is a design parameter matrix, usually selected diagonal. Differentiating $r(t)$ and using (7), the arm dynamics may be written in terms of the filtered tracking error as

$$M\dot{r} = -V_m r - \tau + f + \tau_d \tag{10}$$

where the nonlinear robot function is

$$f(x) = M(q)(\ddot{q}_d + \Lambda\dot{e}) + V_m(q,\dot{q})(\dot{q}_d + \Lambda e) + G(q) + F(\dot{q}) \tag{11}$$

and, for instance, we may select

$$x = [e^T \dot{e}^T q_d^T \dot{q}_d^T \ddot{q}_d^T]^T. \tag{12}$$

Define now a control input torque as

$$\tau_0 = \hat{f} + K_v r \tag{13}$$

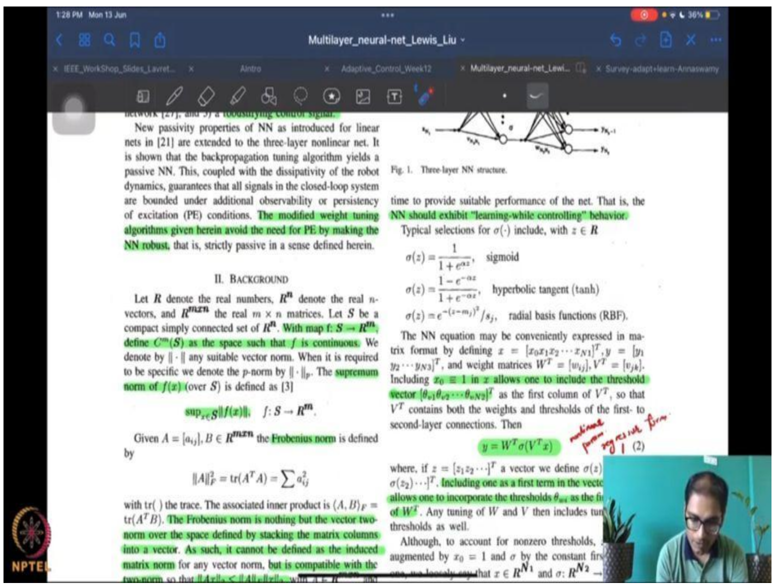
gain matrix $K_v = K_v^T > 0$ and $\hat{f}(x)$ an estimate of $f(x)$ provided by some means not yet disclosed. The closed-loop system becomes

$$M\dot{r} = -(K_v + V_m)r + \tilde{f} + \tau_d \equiv -(K_v + V_m \ldots$$

where the functional estimation error is given

$$\tilde{f} = f - \hat{f}.$$

This is an error system wherein the filtered tr... driven by the functional estimation error.



New passivity properties of NN as introduced for linear nets in [21] are extended to the three-layer nonlinear net. It is shown that the backpropagation tuning algorithm yields a passive NN. This, coupled with the dissipativity of the robot dynamics, guarantees that all signals in the closed-loop system are bounded under additional observability or persistency of excitation (PE) conditions. The modified weight tuning algorithms given herein avoid the need for PE by making the NN robust, that is, strictly passive in a sense defined herein.

II. BACKGROUND

Let R denote the real numbers, R^n denote the real n-vectors, and $R^{m \times n}$ the real $m \times n$ matrices. Let S be a compact simply connected set of R^n. With map f: $S \to R^m$, define $C^m(S)$ as the space such that f is continuous. We denote by $\|\cdot\|$ any suitable vector norm. When it is required to be specific we denote the p-norm by $\|\cdot\|_p$. The supremum norm of $f(x)$ (over S) is defined as [3]

$$\sup_{x \in S} \|f(x)\|, \quad f: S \to R^m.$$

Given $A = [a_{ij}], B \in R^{m \times n}$ the Frobenius norm is defined by

$$\|A\|_F^2 = \mathrm{tr}(A^T A) = \sum a_{ij}^2$$

with tr( ) the trace. The associated inner product is $\langle A, B\rangle_F = \mathrm{tr}(A^T B)$. The Frobenius norm is nothing but the vector two-norm over the space defined by stacking the matrix columns into a vector. As such, it cannot be defined as the induced matrix norm for any vector norm, but is compatible with the two-norm so that ...

Fig. 1. Three-layer NN structure.

time to provide suitable performance of the net. That is, the NN should exhibit "learning-while controlling" behavior.

Typical selections for $\sigma(\cdot)$ include, with $z \in R$

$$\sigma(z) = \frac{1}{1+e^{\alpha z}}, \quad \text{sigmoid}$$

$$\sigma(z) = \frac{1-e^{-\alpha z}}{1+e^{-\alpha z}}, \quad \text{hyperbolic tangent (tanh)}$$

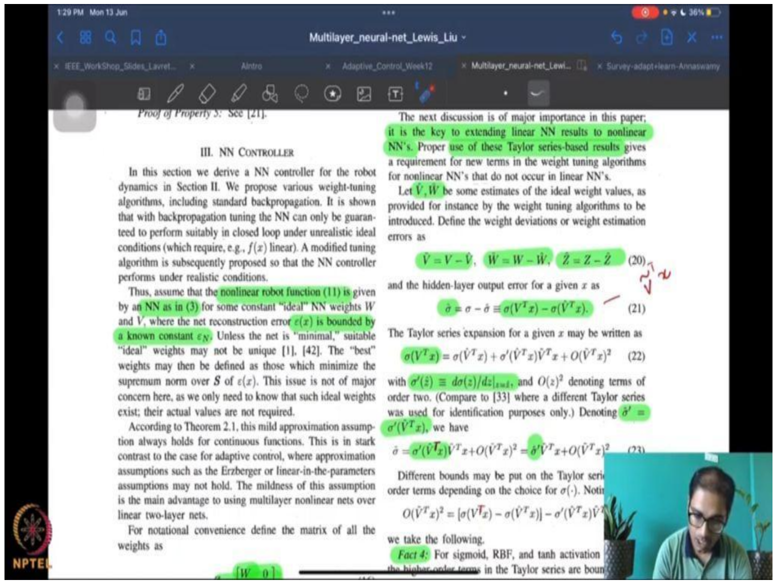
$$\sigma(z) = e^{-(z-m_j)^2}/s_j, \quad \text{radial basis functions (RBF)}.$$

The NN equation may be conveniently expressed in matrix format by defining $x = [x_0 x_1 x_2 \cdots x_{N1}]^T, y = [y_1 y_2 \cdots y_{N3}]^T$, and weight matrices $W^T = [w_{ij}], V^T = [v_{jk}]$. Including $x_0 \equiv 1$ in x allows one to include the threshold vector $[\theta_{v1}\theta_{v2}\cdots\theta_{vN2}]^T$ as the first column of V^T, so that V^T contains both the weights and thresholds of the first- to second-layer connections. Then

$$y = W^T\sigma(V^T x) \tag{2}$$

where, if $z = [z_1 z_2 \cdots]^T$ a vector we define $\sigma(z)$ ... $\sigma(z_2)\cdots]^T$. Including one as a first term in the vect... allows one to incorporate the thresholds θ_{wi} as the fi... of W^T. Any tuning of W and V then includes tun... thresholds as well.

Although, to account for nonzero thresholds, ... augmented by $x_0 = 1$ and σ by the constant firs... one, we loosely say that $x \in R^{N_1}$ and $\sigma: R^{N_2} \to$ ...



Suppose the disturbance is equal to 0, and then you also assume that v is 0. So, this is very nice and simplistic I would say, because you are assuming this w1 is 0, which means you are sort of assuming a lot of nice things, not just this. So, this guy is 0 is okay, this guy being 0 is okay. But then, you are also assuming that this quantity is zero, so, this is of course, I mean I would say pretty restrictive. Then, the tracking error of (r) goes to 0, and all the nice things happen. Because, everything is nice and linear, so you expect all the nice things to happen . So, see if so, let us sort of think about it. I see, I see.

So, if the higher order terms are zero, then this being 0 also make sense, so, this is not such a bad. I mean, once we assumed that there are no higher order terms, that is if f is linear, then this is 0, so this is fine. Actually, this term will be 0, because sigma hat prime is this this guy basically, it is this sort of this guy.

So, so the activation function that you have is there is no activation function per say, I mean, it is linear. So, this this term is of course going to be 0, so this term is of course going to be 0. So, because if there is say if, so basic point being that if f is linear, then there is no higher order terms, I have this. Is not this become 0 is the question.

So, this is the higher order term. If this is the higher order term, do we need n, and this is 0. Then, what is the activation function that we need an activation function would be the sort of question. So, this is also an interesting assumption I would say, because you are taking the derivative of the activation function with respect to an argument.

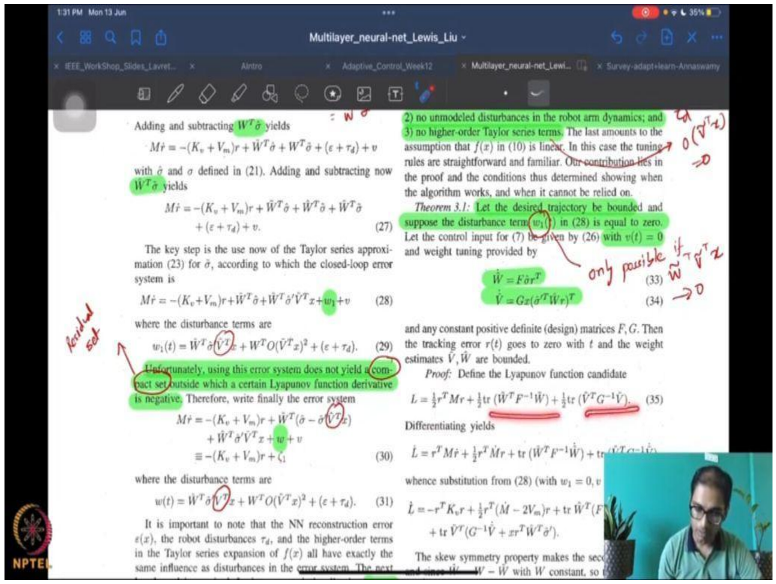
And the activation function if the if the overall function fx is linear, then the question is, do you need an activation function or do you even need the higher layers? So that is the whole point. I believe the activation function becomes the identity function in this case. If the function f is linear, then the activation function is simply the identity function.

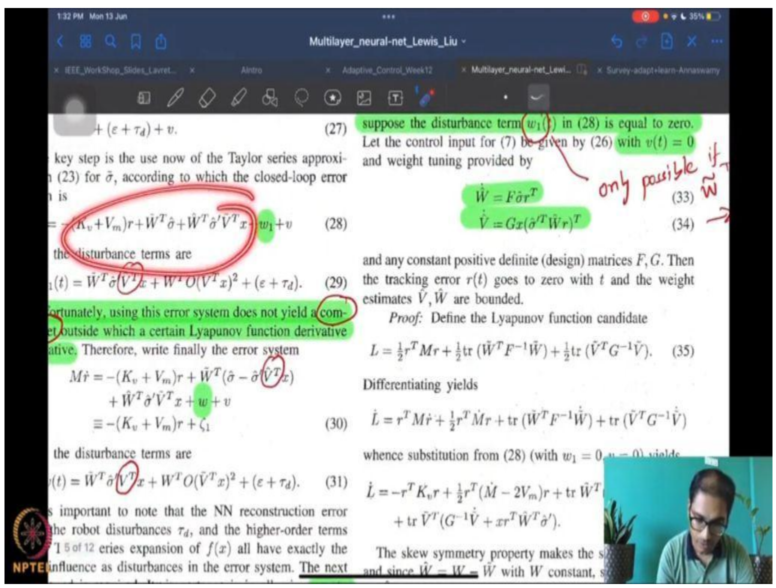
So, sigma of Z is actually equal to Z, so if I take a partial, it just gives me, if you just take the derivative, it just gives me 1. So, this quantity should be just 1, I am wondering if that is how it will be. But, if this quantity is just one, then this is basically V hat transpose x. So basically, you do not have anything like a Taylor series expansion, because this is I mean, in the linear case, this I believe will simply become V tilde transpose x kind of a thing. There is no sigma, sigma is just going to be the identity function is what I believe will happen. And then, but then I wonder what is the, I mean this, why will this still go to 0? Why would this still go to 0 is not very clear.

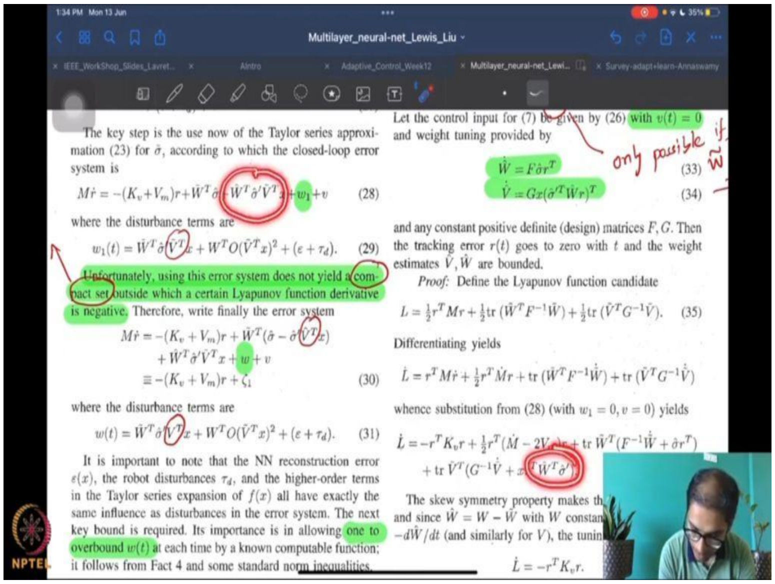
Suppose the disturbance, I mean here the authors are pretty much making an assumption that w1 in 28 is pretty much going to 0. And so, so the question that we are trying to answer to ourselves is of course, this is 0 by the assumptions, no problem. But, what happens to this term is a big question, what happens to this term is a big question.

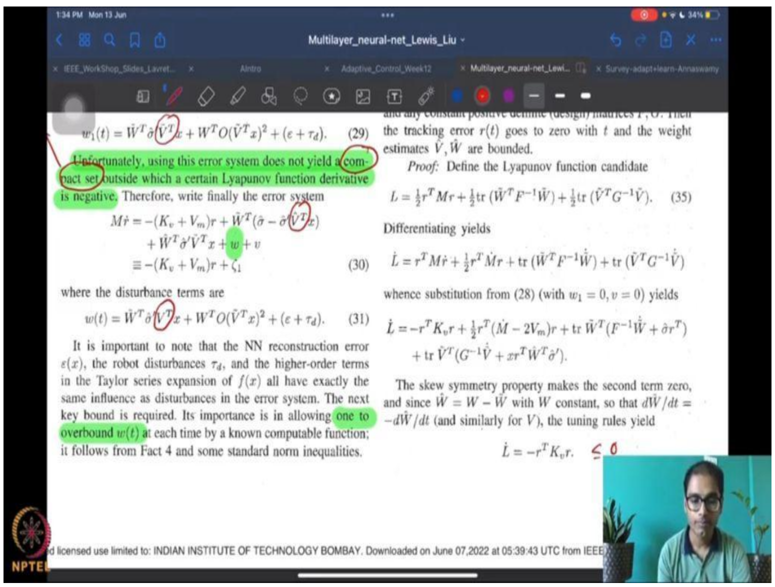
If it is linear, then the activation function is linear, and the derivative is simply one. But, then I still have v tilde transpose x, so, sigma tilde is basically V tilde transpose x. And then I will be left with W tilde transpose times V tilde transpose x. So, this term it is not clear why this would be 0.

(Refer Slide Time: 27:07)









So, this term, so I would say this guy, only possible if W tilde transpose V tilde transpose x is also 0. So, if you assume that this is also some kind of a second order term in the parameter errors, and the parameter errors are relatively small, then yes, this term also goes to 0. I mean, this is a interesting assumption, I would say, this is an interesting assumption is what I would say. I am not very sure of how that will be. But if it does happen, then of course life is super simple. I take my usual Lyapunov function r transpose Mr, which is what I would expect. And the second and third terms are basically these update law type terms.

You I hope you remember these kinds of terms from your model reference adaptive control, because the unknowns are matrices here. So, you you sort of take the trace with some gain matrix F and G, just positive definite symmetric matrices. And then you take the derivative, you get something like this. And if you substitute everything carefully with w1, 0, all you are left with is just these guys. W1 equal to 0, all you are left with is just these guys. You you sort of have and you sort of these terms, you know, if you substitute for r dot, you will have M dot minus 2 Vm, which is going to go to 0. And then you will have the nice minus Kv term, because of this guy.

So, this guy becomes this term, so, that goes to 0 because of the skew symmetry property. Then, you have this nice negative term. Then you have all the sort of the tilde terms, so sorry, you will have this sort of tilde term. And that, of course gets combined here nicely using the trace functions. And this this is what essentially, lets you choose your V hat dot and W hat dot. So, from I am trying to make this quantity 0. So, what I am trying to see is which particular equation the authors have used, here to substitute for r dot. So, here I have this is coming from just the derivative here, and this guy is coming from the equation.

So, here you have W tilde transpose sigma hat r transpose, so r transpose is of course from this. So, W tilde transpose sigma hat that is coming from here, so that is this term, so this term is due to this. Now, if I look at this term, this is W hat term. This is coming due to this guy, so, these two terms. And so, w1 is 0, v is 0, so none of these are there.

So, now if I choose this kind of an update law, everything is rather straightforward, you have a nice tuning function. I mean, you have this nice l dot is minus r transpose Kv r, which is of course a negative semi-definite. But, you can prove that r goes to 0 and you are done, because r goes to 0, e and e dot also go to 0.

So, what did we see today? We started to write the system under the control, where we added a robustness term to the control. And then we sort of did some kind of nice clubbing of the disturbances. Some of it is based on intuition and other one more or less to help your Lyapunov analysis go through. And of course, these are all based on some kind of approximation. So, none of the results that you get will be like precise tracking et-cetera, it will all be boundedness results. But, then we started with this back-propagation type tuning, which helps you give this kind of.

It sort of makes a I would say not a very good assumption that all the disturbances are essentially 0. And, but then what it can also give you is you had zero tracking errors and things like that. So, it is a very very idealized situation, but we do this is sort of a first step. You can use this back propagation tuning to still get some interesting results using this neural network, that is tuned via an adaptive control law. So, we will of course, continue with this in our subsequent final session also. So, I hope you are enjoying our discussion, and I hope to see you again soon.