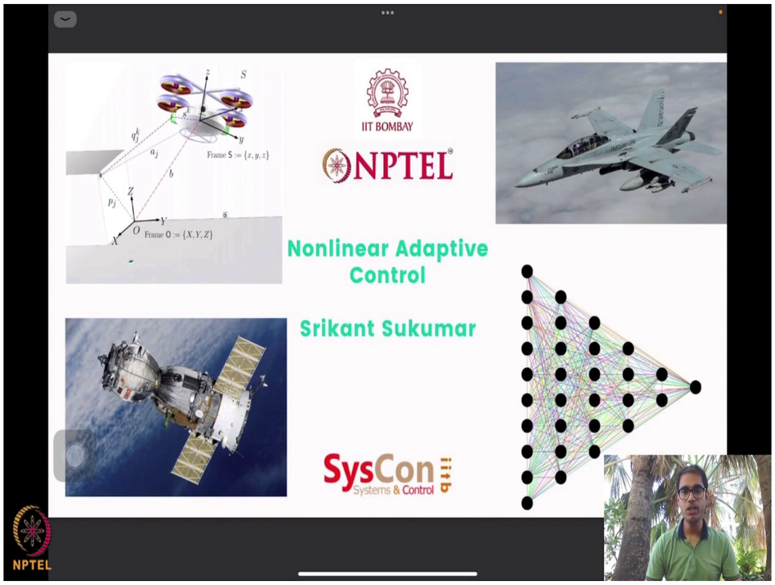
Nonlinear Adaptive Control
Professor Srikant Sukumar
Systems and Control
Indian Institute of Technology, Bombay
Week 12
Lecture No: 69
Real Time Neural Network Based Control of a Robotic Manipulator (Part 3)

(Refer Slide Time: 00:16)



IIT BOMBAY

NPTEL

Nonlinear Adaptive Control

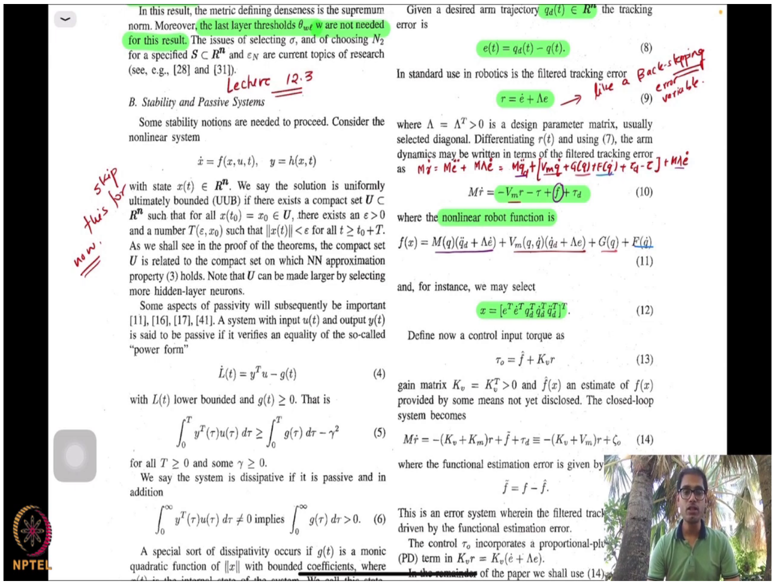
Srikant Sukumar

SysCon Systems & Control

Hello, everyone. Welcome to yet another session of our NPTEL on Nonlinear and Adaptive Control. I am Srikant Sukumar from Systems and Control, IIT Bombay. So, we are in the final week of our course on nonlinear adaptive control, and I really hope that you have learned sufficient material to be able to design algorithms that can drive autonomous systems such as what you see in the background.

Now as always, we are interested in all sorts of applications. Of course, what you see here are mostly aeromechanical systems in the background which is due to my own interest in these, but of course, adaptive control designs are not restricted to these and you can use it on power networks and biological networks and so on and so forth. So, I am always open to hearing more about sort of applications that you folks are envisioning, the use of adaptive control.

(Refer Slide Time: 01:21)



In this result, the metric defining denseness is the supremum norm. Moreover, the last layer thresholds $\theta_{w\ell}$ w are not needed for this result. The issues of selecting σ, and of choosing N_2 for a specified $S \subset R^n$ and ε_N are current topics of research (see, e.g., [28] and [31]).

Lecture 12.3

B. Stability and Passive Systems

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

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

skip this for now.

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

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

A special sort of dissipativity occurs if $g(t)$ is a monic quadratic function of $\|x\|$ with bounded coefficients, where

Given a desired arm trajectory $q_d(t) \in R^n$ the tracking error is

$$e(t) = q_d(t) - q(t). \tag{8}$$

In standard use in robotics is the filtered tracking error

$$r = \dot{e} + \Lambda e \rightarrow \text{like a Back-stepping error variable.} \tag{9}$$

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 $M\dot{r} = M\ddot{e} + M\Lambda\dot{e} = M\ddot{q}_d + [V_m\dot{q} + G(q) + F(\dot{q}) + \tau_d - \tau] + M\Lambda\dot{e}$

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

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

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

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

$$M\dot{r} = -(K_v + K_m)r + \tilde{f} + \tau_d \equiv -(K_v + V_m)r + \zeta_o \tag{14}$$

where the functional estimation error is given by

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

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

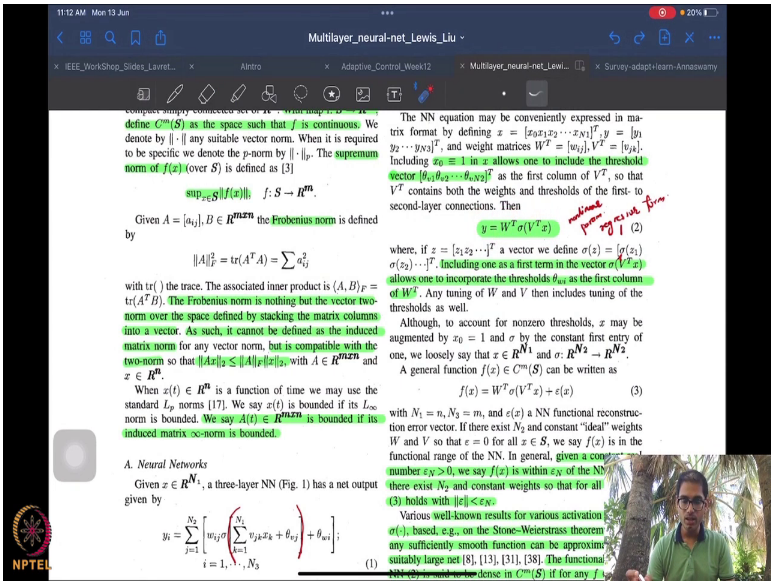
The control τ_o incorporates a proportional-plu (PD) term in $K_v r = K_v(\dot{e} + \Lambda e)$.

~~In the remainder~~ of the paper we shall use (14)

So, what we are looking at in this week, in this final week is the connections with learning, and we are now specifically looking at sort of a deep learning problem. But of course, here, we are sort of different in the typical, different from the typical learning sort of results in the sense that we are not doing the learning offline then implementing it later but we are doing learning and control simultaneously.

We can do this because essentially this is connected to parameter learning which we already know very well. So, that is what we sort of applied. The only difference or which is of course a significant difference is that now we have a very nonlinear regressor parameter form which is not what we, we are used to until now.

(Refer Slide Time: 02:15)


11:12 AM Mon 13 Jun
Multilayer_neural-net_Lewis_Liu
IEEE_WorkShop_Slides_Lavret...
AIntro
Adaptive_Control_Week12
Multilayer_neural-net_Lewi...
Survey-adapt+learn-Annaswamy
20%
NPTEL
nonlinear param.
regressive form

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 $A = [a_{ij}], B \in R^{mxn}$ 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 $\|Ax\|_2 \le \|A\|_F\|x\|_2$, with $A \in R^{mxn}$ and $x \in R^n$.

When $x(t) \in R^n$ is a function of time we may use the standard L_p norms [17]. We say $x(t)$ is bounded if its L_∞ norm is bounded. We say $A(t) \in R^{mxn}$ is bounded if its induced matrix ∞-norm is bounded.

A. Neural Networks

Given $x \in R^{N_1}$, a three-layer NN (Fig. 1) has a net output given by

$$y_i = \sum_{j=1}^{N_2}\left[w_{ij}\sigma\left(\sum_{k=1}^{N_1} v_{jk}x_k + \theta_{vj}\right) + \theta_{wi}\right]; \quad i = 1, \cdots, N_3 \tag{1}$$

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

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

A general function $f(x) \in C^m(S)$ can be written as

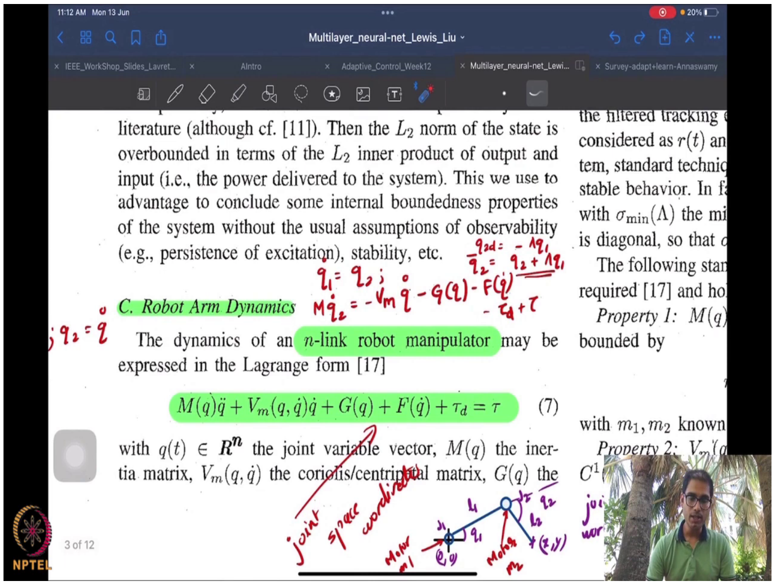
$$f(x) = W^T\sigma(V^T x) + \varepsilon(x) \tag{3}$$

with $N_1 = n, N_3 = m$, and $\varepsilon(x)$ a NN functional reconstruction error vector. If there exist N_2 and constant "ideal" weights W and V so that $\varepsilon = 0$ for all $x \in S$, we say $f(x)$ is in the functional range of the NN. In general, given a con number $\varepsilon_N > 0$, we say $f(x)$ is within ε_N of the NN there exist N_2 and constant weights so that for all (3) holds with $\|\varepsilon\| < \varepsilon_N$.

Various well-known results for various activation $\sigma(\cdot)$, based, e.g., on the Stone–Weierstrass theorem any sufficiently smooth function can be approxima suitably large net [8], [13], [31], [38]. The functional NN (2) is said to be dense in $C^m(S)$ if for any f

We are always assuming linear parameterization and things like that. But you see here that everything is rather nonlinear. And so, we still have to see how we can deal with this kind of nonlinearity.

(Refer Slide Time: 02:38)


11:12 AM Mon 13 Jun
Multilayer_neural-net_Lewis_Liu
IEEE_WorkShop_Slides_Lavret...
AIntro
Adaptive_Control_Week12
Multilayer_neural-net_Lewi...
Survey-adapt+learn-Annaswamy
20%
NPTEL
3 of 12
$q_{2d} = -\Lambda q_1$
$\bar{q}_2 = q_2 + \Lambda q_1$
$\dot{q}_1 = q_2$;
$M\dot{q}_2 = -V_m\dot{q} - G(q) - F(\dot{q}) - \tau_d + \tau$
; $q_2 = \dot{q}$
joint space coordinates
Motor m1
(0,0)
q_1
l_1
l_2
q_2
Motor m2
(x,y)
joi
wor

literature (although cf. [11]). Then the L_2 norm of the state is overbounded in terms of the L_2 inner product of output and input (i.e., the power delivered to the system). This we use to advantage to conclude some internal boundedness properties of the system without the usual assumptions of observability (e.g., persistence of excitation), stability, etc.

C. Robot Arm Dynamics

The dynamics of an n-link robot manipulator may be expressed in the Lagrange form [17]

$$M(q)\ddot{q} + V_m(q,\dot{q})\dot{q} + G(q) + F(\dot{q}) + \tau_d = \tau \tag{7}$$

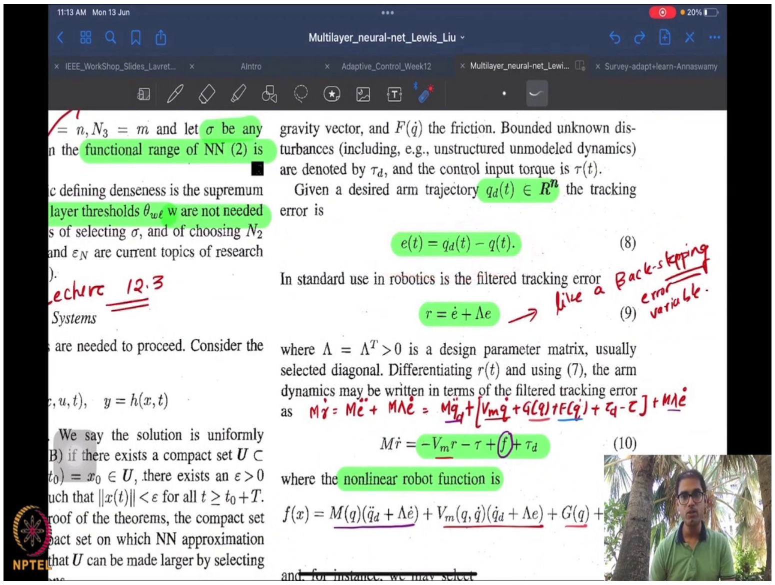
with $q(t) \in R^n$ the joint variable vector, $M(q)$ the inertia matrix, $V_m(q,\dot{q})$ the coriolis/centripetal matrix, $G(q)$ the

the filtered tracking
considered as $r(t)$ an
tem, standard techniq
stable behavior. In f
with $\sigma_{\min}(\Lambda)$ the mi
is diagonal, so that
The following stan
required [17] and ho
Property 1: $M(q)$
bounded by

with m_1, m_2 known
Property 2: $V_m(q$
C^1

In the previous lecture, of course, we started to understand some details about how my robotic arm dynamics looks because this is the application that we are going to be looking at for application, for the use of these neural nets. And we saw the basic Euler-Lagrange dynamics, what each term means, what is the joint on the world space coordinates.

(Refer Slide Time: 02:55)


11:13 AM Mon 13 Jun
20%
Multilayer_neural-net_Lewis_Liu
IEEE_WorkShop_Slides_Lavret...
AIntro
Adaptive_Control_Week12
Multilayer_neural-net_Lewi...
Survey-adapt+learn-Annaswamy
NPTEL

$= n, N_3 = m$ and let σ be any
n the functional range of NN (2) is

c defining denseness is the supremum
layer thresholds $\theta_{w\ell}$ w are not needed
s of selecting σ, and of choosing N_2
and ε_N are current topics of research
).

ecture 12.3

Systems

are needed to proceed. Consider the

$, u, t), \quad y = h(x, t)$

We say the solution is uniformly
B) if there exists a compact set $U \subset$
$t_0) = x_0 \in U$, there exists an $\varepsilon > 0$
uch that $\|x(t)\| < \varepsilon$ for all $t \geq t_0 + T$.
roof of the theorems, the compact set
pact set on which NN approximation
that U can be made larger by selecting

gravity vector, and $F(\dot{q})$ the friction. Bounded unknown disturbances (including, e.g., unstructured unmodeled dynamics) are denoted by τ_d, and the control input torque is $\tau(t)$.

Given a desired arm trajectory $q_d(t) \in \boldsymbol{R}^n$ the tracking error is

$$e(t) = q_d(t) - q(t). \tag{8}$$

In standard use in robotics is the filtered tracking error

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

$\rightarrow$ like a Back-stepping error variable.

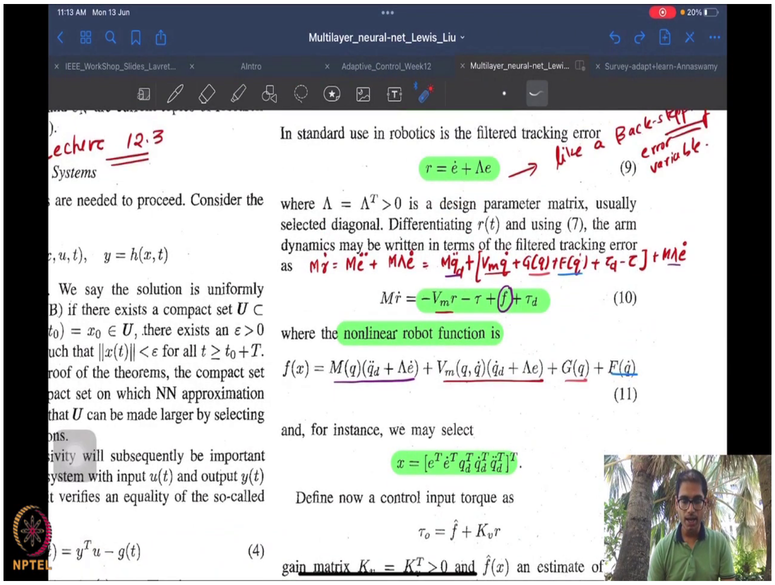
where $\Lambda = \Lambda^T > 0$ is a design parameter matrix, usually selected diagonal. Differentiating $r(t)$ and using (7), the arm dynamics may be written in terms of the filtered tracking error as $M\dot{r} = M\ddot{e} + M\Lambda\dot{e} = M\ddot{q}_d + [V_m\dot{q} + G(q) + F(\dot{q}) + \tau_d - \tau] + M\Lambda\dot{e}$

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

where the nonlinear robot function is

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

and ~~for instance, we may select~~


11:13 AM Mon 13 Jun
20%
Multilayer_neural-net_Lewis_Liu
IEEE_WorkShop_Slides_Lavret...
AIntro
Adaptive_Control_Week12
Multilayer_neural-net_Lewi...
Survey-adapt+learn-Annaswamy
NPTEL

).

ecture 12.3

Systems

are needed to proceed. Consider the

$, u, t), \quad y = h(x, t)$

We say the solution is uniformly
B) if there exists a compact set $U \subset$
$t_0) = x_0 \in U$, there exists an $\varepsilon > 0$
uch that $\|x(t)\| < \varepsilon$ for all $t \geq t_0 + T$.
roof of the theorems, the compact set
pact set on which NN approximation
that U can be made larger by selecting
ons.
ivity will subsequently be important
ystem with input $u(t)$ and output $y(t)$
t verifies an equality of the so-called

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

In standard use in robotics is the filtered tracking error

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

$\rightarrow$ like a Back-stepping error variable.

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 $M\dot{r} = M\ddot{e} + M\Lambda\dot{e} = M\ddot{q}_d + [V_m\dot{q} + G(q) + F(\dot{q}) + \tau_d - \tau] + M\Lambda\dot{e}$

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

where the nonlinear robot function is

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

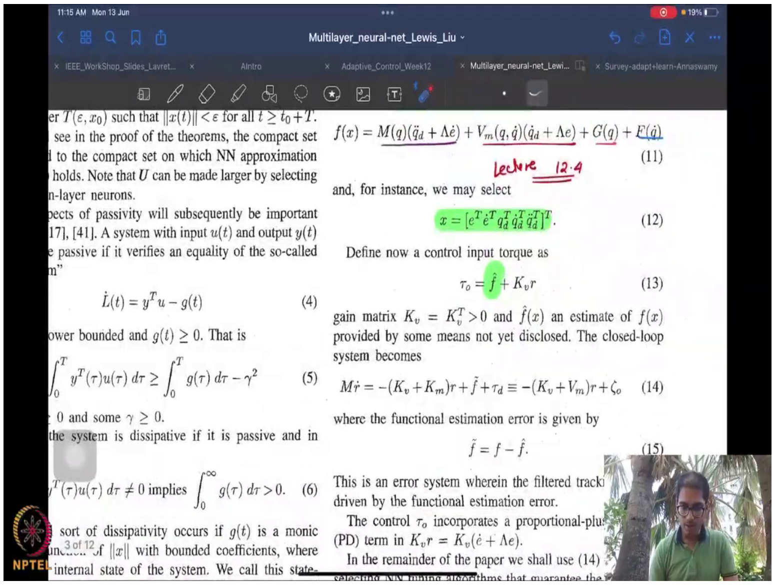
and, for instance, we may select

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

Define now a control input torque as

$$\tau_o = \hat{f} + K_v r$$

gain matrix $K_v = K_v^T > 0$ and $\hat{f}(x)$ an estimate of


11:15 AM Mon 13 Jun
19%
Multilayer_neural-net_Lewis_Liu
IEEE_WorkShop_Slides_Lavret...
AIntro
Adaptive_Control_Week12
Multilayer_neural-net_Lewi...
Survey-adapt+learn-Annaswamy
3 of 12
NPTEL

er $T(\varepsilon, x_0)$ such that $\|x(t)\| < \varepsilon$ for all $t \geq t_0 + T$.
see in the proof of the theorems, the compact set
l to the compact set on which NN approximation
holds. Note that U can be made larger by selecting
n-layer neurons.
pects of passivity will subsequently be important
17], [41]. A system with input $u(t)$ and output $y(t)$
e passive if it verifies an equality of the so-called
m"

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

ower bounded and $g(t) \geq 0$. That is

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

0 and some $\gamma \geq 0$.
the system is dissipative if it is passive and in

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

sort of dissipativity occurs if $g(t)$ is a monic
unc... of $\|x\|$ with bounded coefficients, where
internal state of the system. We call this state-

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

Lecture 12.4

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

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

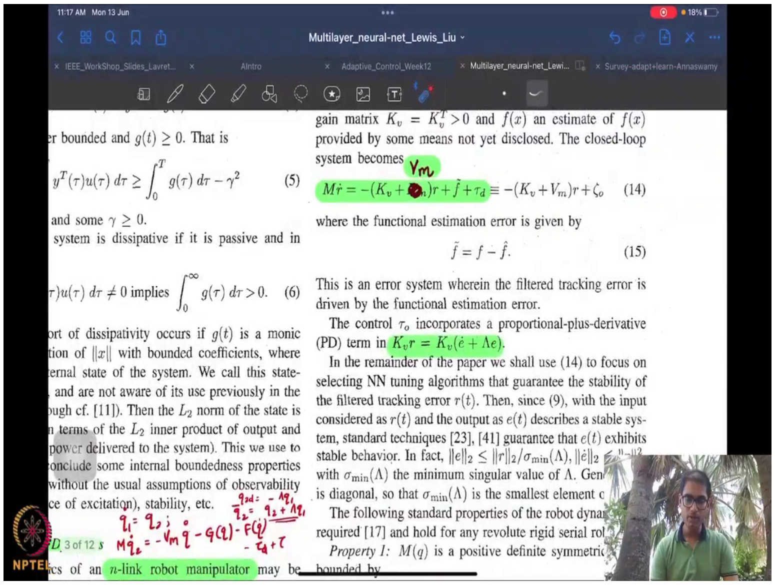
gain matrix $K_v = K_v^T > 0$ and $\hat{f}(x)$ an estimate of $f(x)$ provided by some means not yet disclosed. The closed-loop system becomes

$$M\dot{r} = -(K_v + K_m)r + \tilde{f} + \tau_d \equiv -(K_v + V_m)r + \zeta_o \tag{14}$$

where the functional estimation error is given by

$$\tilde{f} = f - \hat{f}. \tag{15}$$

This is an error system wherein the filtered tracki
driven by the functional estimation error.
The control τ_o incorporates a proportional-plu
(PD) term in $K_v r = K_v(\dot{e} + \Lambda e)$.
In the remainder of the paper we shall use (14)
selecting NN tuning algorithms that guarantee the


11:17 AM Mon 13 Jun
18%
Multilayer_neural-net_Lewis_Liu
IEEE_WorkShop_Slides_Lavret...
AIntro
Adaptive_Control_Week12
Multilayer_neural-net_Lewi...
Survey-adapt+learn-Annaswamy
3 of 12
NPTEL

er bounded and $g(t) \geq 0$. That is

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

and some $\gamma \geq 0$.
system is dissipative if it is passive and in

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

ort of dissipativity occurs if $g(t)$ is a monic
tion of $\|x\|$ with bounded coefficients, where
ernal state of the system. We call this state-
, and are not aware of its use previously in the
ugh cf. [11]). Then the L_2 norm of the state is
n terms of the L_2 inner product of output and
power delivered to the system). This we use to
onclude some internal boundedness properties
without the usual assumptions of observability
ce of excitation), stability, etc.

$\dot{q}_1 = q_2$; $\quad q_{2d} = -\Lambda q_1$

$\bar{q}_2 = q_2 + \Lambda q_1$

$M\dot{q}_2 = -V_m \dot{q} - G(q) - F(\dot{q}) - \tau_d + \tau$

D.
ics of an n-link robot manipulator may be

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

V_m

$$M\dot{r} = -(K_v + K_m)r + \tilde{f} + \tau_d \equiv -(K_v + V_m)r + \zeta_o \tag{14}$$

where the functional estimation error is given by

$$\tilde{f} = f - \hat{f}. \tag{15}$$

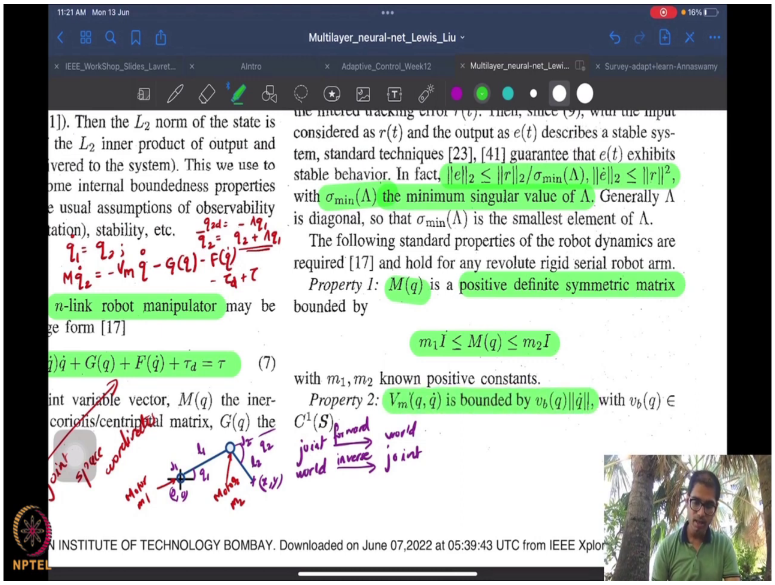
This is an error system wherein the filtered tracking error is driven by the functional estimation error.

The control τ_o incorporates a proportional-plus-derivative (PD) term in $K_v r = K_v(\dot{e} + \Lambda e)$.

In the remainder of the paper we shall use (14) to focus on selecting NN tuning algorithms that guarantee the stability of the filtered tracking error $r(t)$. Then, since (9), with the input considered as $r(t)$ and the output as $e(t)$ describes a stable system, standard techniques [23], [41] guarantee that $e(t)$ exhibits stable behavior. In fact, $\|e\|_2 \leq \|r\|_2/\sigma_{\min}(\Lambda)$, $\|\dot{e}\|_2 \leq$
with $\sigma_{\min}(\Lambda)$ the minimum singular value of Λ. Gen
is diagonal, so that $\sigma_{\min}(\Lambda)$ is the smallest element c

The following standard properties of the robot dynar
required [17] and hold for any revolute rigid serial rol

Property 1: $M(q)$ is a positive definite symmetric
bounded by


11:21 AM Mon 13 Jun
16%
Multilayer_neural-net_Lewis_Liu
IEEE_WorkShop_Slides_Lavret...
AIntro
Adaptive_Control_Week12
Multilayer_neural-net_Lewi...
Survey-adapt+learn-Annaswamy
NPTEL

1]). Then the L_2 norm of the state is the L_2 inner product of output and vered to the system). This we use to ome internal boundedness properties e usual assumptions of observability ation), stability, etc.

$q_{2d} = -\Lambda q_1$

$\bar{q}_2 = q_2 + \Lambda q_1$

$\dot{q}_1 = q_2;$

$M\dot{q}_2 = -V_m \dot{q} - G(q) - F(\dot{q}) - \tau_d + \tau$

n-link robot manipulator may be ge form [17]

$$)\dot{q} + G(q) + F(\dot{q}) + \tau_d = \tau \quad (7)$$

int variable vector, $M(q)$ the iner- coriolis/centripetal matrix, $G(q)$ the

joint space coordinates

the filtered tracking error $r(t)$. Then, since (9), with the input considered as $r(t)$ and the output as $e(t)$ describes a stable system, standard techniques [23], [41] guarantee that $e(t)$ exhibits stable behavior. In fact, $\|e\|_2 \le \|r\|_2/\sigma_{\min}(\Lambda), \|\dot{e}\|_2 \le \|r\|^2$, with $\sigma_{\min}(\Lambda)$ the minimum singular value of Λ. Generally Λ is diagonal, so that $\sigma_{\min}(\Lambda)$ is the smallest element of Λ.

The following standard properties of the robot dynamics are required [17] and hold for any revolute rigid serial robot arm.

Property 1: $M(q)$ is a positive definite symmetric matrix bounded by

$$m_1 I \le M(q) \le m_2 I$$

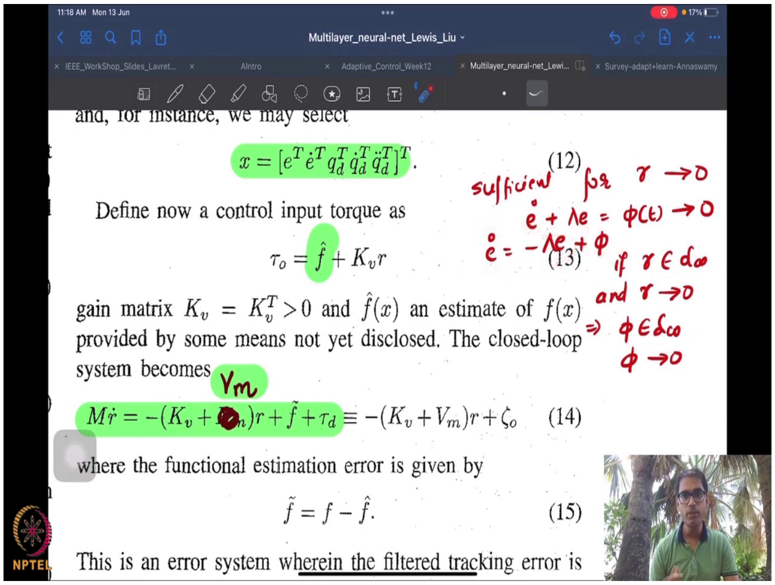
with m_1, m_2 known positive constants.

Property 2: $V_m(q, \dot{q})$ is bounded by $v_b(q)\|\dot{q}\|$, with $v_b(q) \in C^1(S)$

$$\text{joint} \xrightarrow{\text{forward}} \text{world}$$

$$\text{world} \xrightarrow{\text{inverse}} \text{joint}$$

N INSTITUTE OF TECHNOLOGY BOMBAY. Downloaded on June 07,2022 at 05:39:43 UTC from IEEE Xplor


11:18 AM Mon 13 Jun
17%
Multilayer_neural-net_Lewis_Liu
IEEE_WorkShop_Slides_Lavret...
AIntro
Adaptive_Control_Week12
Multilayer_neural-net_Lewi...
Survey-adapt+learn-Annaswamy
NPTEL

and, for instance, we may select

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

sufficient for $r \to 0$

$\dot{e} + \Lambda e = \phi(t) \to 0$

$\dot{e} = -\Lambda e + \phi$

if $r \in \mathcal{L}_\infty$

and $r \to 0$

$\Rightarrow \phi \in \mathcal{L}_\infty$

$\phi \to 0$

Define now a control input torque as

$$\tau_o = \hat{f} + K_v r \quad (13)$$

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

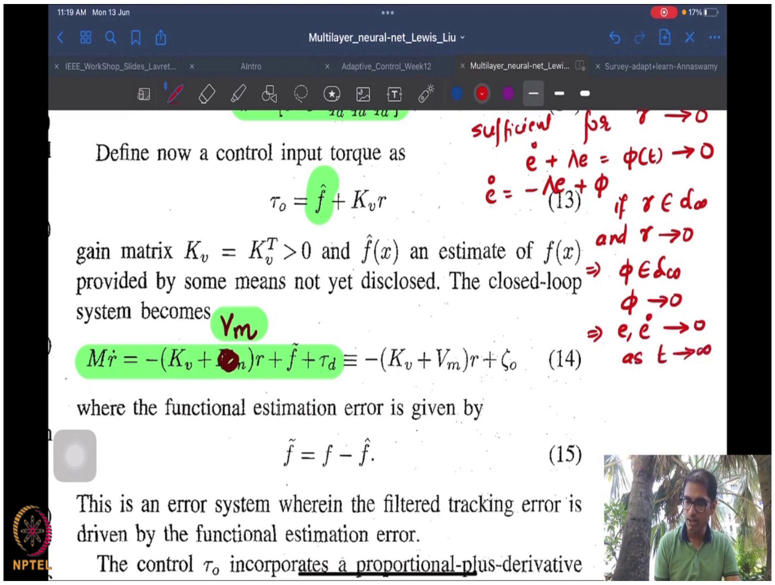
V_m

$$M\dot{r} = -(K_v + V_m)r + \tilde{f} + \tau_d \equiv -(K_v + V_m)r + \zeta_o \quad (14)$$

where the functional estimation error is given by

$$\tilde{f} = f - \hat{f}. \quad (15)$$

This is an error system wherein the filtered tracking error is


11:19 AM Mon 13 Jun
17%
Multilayer_neural-net_Lewis_Liu
IEEE_WorkShop_Slides_Lavret...
AIntro
Adaptive_Control_Week12
Multilayer_neural-net_Lewi...
Survey-adapt+learn-Annaswamy
NPTEL

Define now a control input torque as

$$\tau_o = \hat{f} + K_v r \qquad (13)$$

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

$$M\dot{r} = -(K_v + V_m)r + \tilde{f} + \tau_d \equiv -(K_v + V_m)r + \zeta_o \qquad (14)$$

where the functional estimation error is given by

$$\tilde{f} = f - \hat{f}. \qquad (15)$$

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

The control τ_o incorporates a proportional-plus-derivative

sufficient for $r \to 0$

$\dot{e} + \Lambda e = \phi(t) \to 0$

$\dot{e} = -\Lambda e + \phi$

if $r \in \mathcal{L}_\infty$ and $r \to 0$

$\Rightarrow \phi \in \mathcal{L}_\infty$

$\phi \to 0$

$\Rightarrow e, \dot{e} \to 0$ as $t \to \infty$

V_m

And we also constructed this error variable with respect to a desired arm trajectory. And we also defined this backstepping error type variable, which we have seen before, and we wrote the dynamics in terms of this backstepping error variable. And in these dynamics now, we have this function f which is called a nonlinear robot function. And this is the function, as you can imagine, we will approximate using our three-layer neural network.

So, this is where we were. I am going to mark our lecture here, lecture 12.4. So, now if you look at how we want to design our controller, what we do is, we put in an approximation or a estimation of f, and that is called f hat. So, unlike what you have seen in adaptive control before, we are not approximate, we are not just estimating parameters but we are now estimating functions. And that is why we have the notion, notation f hat.

And then of course we have a K v times r which is like a gain matrix, symmetric positive definite gain matrix times this backstepping error variable r. So, this is the control. So, it is called tau 0 because we, we design the control sort of assuming that there is no disturbance. We cannot do anything to sort of really cancel the disturbance here. So, we just design a control assuming there is none.

And this is you sort of try to cancel this guy with its estimate, just sort of getting motivated from the certainty equivalence type idea. Then we leave this term as it is, because honestly, this term is going to cancel out on its own. So, we do not worry about this term. Now, so as we mentioned f hat is an estimate of f by some means which is not yet disclosed.

So, what happens to the closed loop system? So, there is this notation error here. So, the closed loop system is not K m but V m, becomes this guy, minus K v plus V m times r, and plus an f tilde, and this tau d. And this is where you have sort of this f tilde which is the function estimation error. So, this is just a term zeta naught which is combining these two, nothing special about this.

So, this is essentially like an error system. So, I am going to highlight this and. So, this is sort of the error system. And of course, this is driven by the function estimation error. So, f tilde of course shows up here. So, this is sort of like a, I mean although the tau 0 seems to have only one term which is K v, I mean K v r, I mean just two terms but one of them is to more or less counter the effect of the nonlinear robot function, and the other term is what is the like the stabilizing term that we usually inject

And this term, we always look like just one term, is in fact a proportional derivative type term because of the nature of r. So, I have a e dot term and an e term. So, this is important to remember, that it has both a proportional and a derivative term. So, a pd-type controller, very, again, something which is very standard in aero-mechanical applications even for space stuff, pd type controllers are known and found to be stabilizing.

So, in the remain, of course, in the remainder of the paper we, I mean, the authors use this equation 14. And we, of course, focus on tuning this neural network in a smart way so that you can get r to go to 0 because you want r to go to 0. So, why is it, I mean one very quick aside, why is it sufficient for r to go to 0? I want to quickly point that out wherever r is defined.

So, sufficient for r to go to 0. Why? Because this is an e dot plus lambda e equal to some phi function. This goes to 0, which means e dot is equal to minus lambda e plus phi. And what do I know about phi? Suppose, if phi is also bounded, we can prove that pi is bounded, which means r is bounded. If let us see, I will put it properly. If r is bounded, that is l infinity, and r of course goes to 0, implies phi is bound and phi goes to 0 which means that, and what do I know? I know that this is a stable system because lambda is positive definite symmetry. So, this is a stable system with a bounded forcing.

So, you have already proved this kind of a result when we did this Ortega construction a long time ago, that if this e dot plus lambda e type thing goes to 0, then e has to go to 0. So, this immediately implies that e comma e dot both go to 0 as t goes to infinity. So, it is enough for us to show that r goes to 0. So, that is what we aim to do.

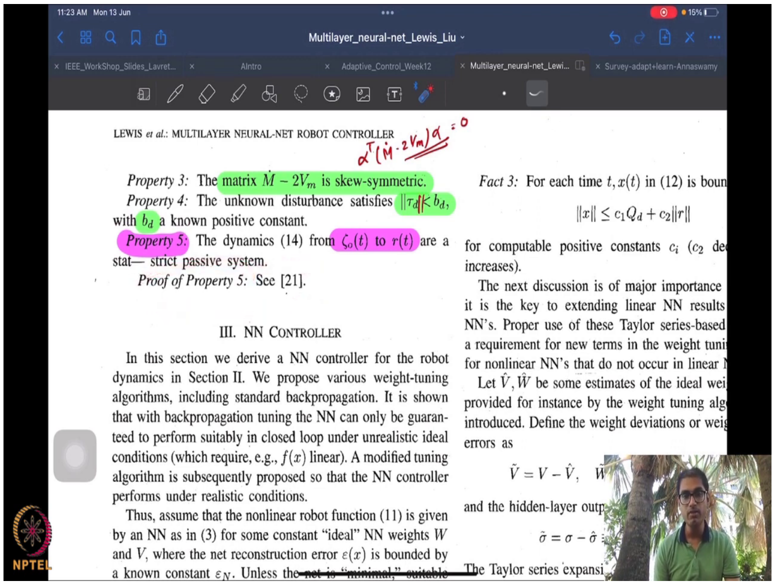
This is essentially like an Ortega type construction idea. Even in the Ortega construction, we did the same thing. So, if you do not remember, again, I would ask you to go back and refer to the Ortega. So, that is what it is mentioned here. So, e exhibit stable behavior, in fact, e is less than r, I mean, even if you do not have, I mean of course we are talking about r going to 0. In this case that may not happen.

But even if r remains bounded, has some nice bound or just by virtue of this equation that r is e dot plus lambda e, then it is, it can be shown, even without talking about things going to 0, because honestly speaking in this article we will not be able to prove that anything goes to 0 per se, but we only prove nice bounded performance because after all, now whenever we talk about using neural networks for functions, it is eventually an approximation of the function. So, there is always some error.

So, obviously, if there is some error, you cannot expect, just like in case of the disturbance, you cannot expect to, for things to converge to 0. So, the best you will do is get some kind of bounded performance. So, just with the relationship between e and r, you can actually conclude that the two norm of e is bounded by the two norm of r and similarly the two norm of e dot is also bounded by the square of, norm r square, norm r.

So, this is important to remember. Then, we also have very standard properties of the robot model. So, M is basically positive definite symmetric, and therefore it satisfies this kind of a unique variable. So, you remember that whenever you have positive definite symmetric matrices, you have this kind of an inequality satisfied. The second property is that V m is bounded. And by some continuous function multiplied by norm of q dot. This is again a property of the centripetal coriolis term.

(Refer Slide Time: 11:25)


11:23 AM Mon 13 Jun
15%
Multilayer_neural-net_Lewis_Liu
IEEE_WorkShop_Slides_Lavret...
AIntro
Adaptive_Control_Week12
Multilayer_neural-net_Lewi...
Survey-adapt+learn-Annaswamy
NPTEL

LEWIS *et al.*: MULTILAYER NEURAL-NET ROBOT CONTROLLER

$\alpha^T(\dot{M}-2V_m)\alpha = 0$

Property 3: The matrix $\dot{M}-2V_m$ is skew-symmetric.

Property 4: The unknown disturbance satisfies $\|\tau_d\| < b_d$, with b_d a known positive constant.

Property 5: The dynamics (14) from $\zeta_o(t)$ to $r(t)$ are a stat— strict passive system.

Proof of Property 5: See [21].

III. NN Controller

In this section we derive a NN controller for the robot dynamics in Section II. We propose various weight-tuning algorithms, including standard backpropagation. It is shown that with backpropagation tuning the NN can only be guaranteed to perform suitably in closed loop under unrealistic ideal conditions (which require, e.g., $f(x)$ linear). A modified tuning algorithm is subsequently proposed so that the NN controller performs under realistic conditions.

Thus, assume that the nonlinear robot function (11) is given by an NN as in (3) for some constant "ideal" NN weights W and V, where the net reconstruction error $\varepsilon(x)$ is bounded by a known constant ε_N. Unless the net is "minimal," suitable

Fact 3: For each time t, $x(t)$ in (12) is boun

$$\|x\| \le c_1 Q_d + c_2\|r\|$$

for computable positive constants c_i (c_2 de
increases).

The next discussion is of major importance
it is the key to extending linear NN results
NN's. Proper use of these Taylor series-based
a requirement for new terms in the weight tuni
for nonlinear NN's that do not occur in linear N

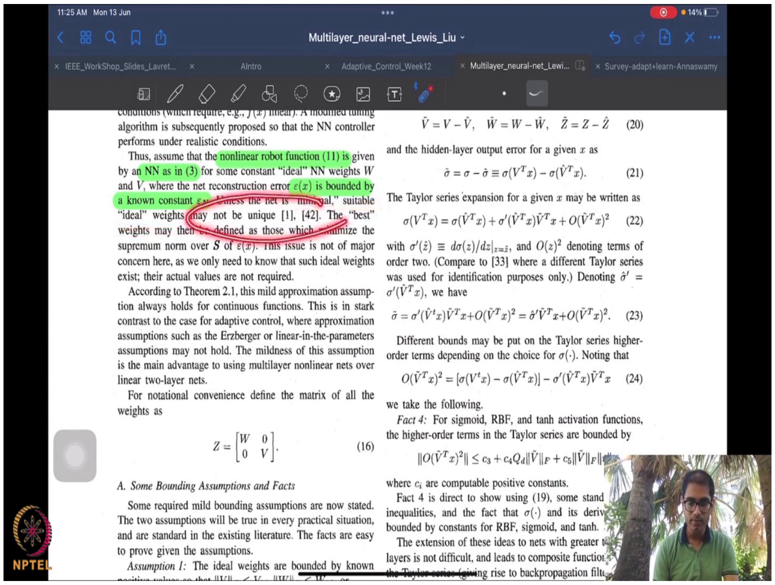
Let $\hat{V}, \hat{W}$ be some estimates of the ideal wei
provided for instance by the weight tuning alg
introduced. Define the weight deviations or weig
errors as

$$\tilde{V} = V - \hat{V}, \quad W$$

and the hidden-layer outp

$$\tilde{\sigma} = \sigma - \hat{\sigma}$$

The Taylor series expansi


11:25 AM Mon 13 Jun
14%
Multilayer_neural-net_Lewis_Liu
IEEE_WorkShop_Slides_Lavret...
AIntro
Adaptive_Control_Week12
Multilayer_neural-net_Lewi...
Survey-adapt+learn-Annaswamy
NPTEL

conditions (which require, e.g., $f(x)$ linear). A modified tuning algorithm is subsequently proposed so that the NN controller performs under realistic conditions.

Thus, assume that the nonlinear robot function (11) is given by an NN as in (3) for some constant "ideal" NN weights W and V, where the net reconstruction error $\varepsilon(x)$ is bounded by a known constant ε_N. Unless the net is "minimal," suitable "ideal" weights may not be unique [1], [42]. The "best" weights may then be defined as those which minimize the supremum norm over S of $\varepsilon(x)$. This issue is not of major concern here, as we only need to know that such ideal weights exist; their actual values are not required.

According to Theorem 2.1, this mild approximation assumption always holds for continuous functions. This is in stark contrast to the case for adaptive control, where approximation assumptions such as the Erzberger or linear-in-the-parameters assumptions may not hold. The mildness of this assumption is the main advantage to using multilayer nonlinear nets over linear two-layer nets.

For notational convenience define the matrix of all the weights as

$$Z = \begin{bmatrix} W & 0 \\ 0 & V \end{bmatrix}. \tag{16}$$

A. Some Bounding Assumptions and Facts

Some required mild bounding assumptions are now stated. The two assumptions will be true in every practical situation, and are standard in the existing literature. The facts are easy to prove given the assumptions.

Assumption 1: The ideal weights are bounded by known

$$\tilde{V} = V - \hat{V}, \quad \tilde{W} = W - \hat{W}, \quad \tilde{Z} = Z - \hat{Z} \tag{20}$$

and the hidden-layer output error for a given x as

$$\tilde{\sigma} = \sigma - \hat{\sigma} \equiv \sigma(V^T x) - \sigma(\hat{V}^T x). \tag{21}$$

The Taylor series expansion for a given x may be written as

$$\sigma(V^T x) = \sigma(\hat{V}^T x) + \sigma'(\hat{V}^T x)\tilde{V}^T x + O(\tilde{V}^T x)^2 \tag{22}$$

with $\sigma'(\hat{z}) \equiv d\sigma(z)/dz|_{z=\hat{z}}$, and $O(z)^2$ denoting terms of order two. (Compare to [33] where a different Taylor series was used for identification purposes only.) Denoting $\hat{\sigma}' = \sigma'(\hat{V}^T x)$, we have

$$\tilde{\sigma} = \sigma'(\hat{V}^t x)\tilde{V}^T x + O(\tilde{V}^T x)^2 = \hat{\sigma}'\tilde{V}^T x + O(\tilde{V}^T x)^2. \tag{23}$$

Different bounds may be put on the Taylor series higher-order terms depending on the choice for $\sigma(\cdot)$. Noting that

$$O(\tilde{V}^T x)^2 = [\sigma(V^t x) - \sigma(\hat{V}^T x)] - \sigma'(\hat{V}^T x)\tilde{V}^T x \tag{24}$$

we take the following.

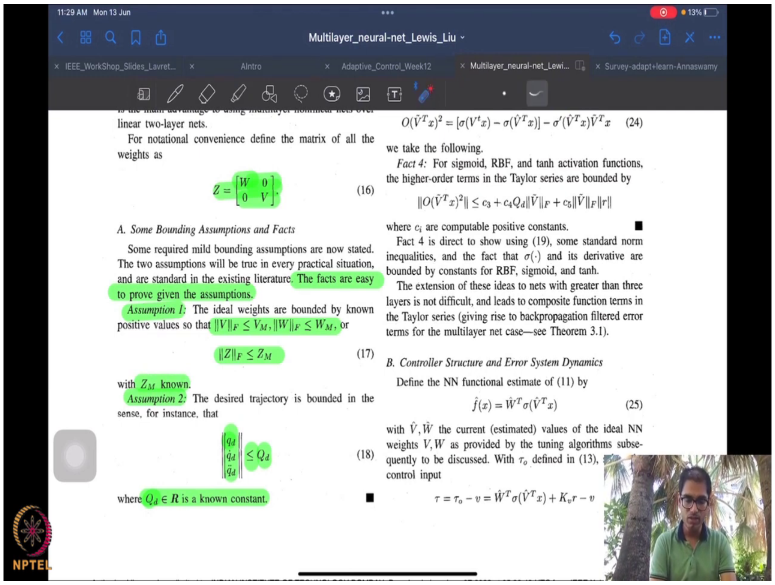
Fact 4: For sigmoid, RBF, and tanh activation functions, the higher-order terms in the Taylor series are bounded by

$$\|O(\tilde{V}^T x)^2\| \le c_3 + c_4 Q_d\|\tilde{V}\|_F + c_5\|\tilde{V}\|_F\|$$

where c_i are computable positive constants.

Fact 4 is direct to show using (19), some stand
inequalities, and the fact that $\sigma(\cdot)$ and its deriv
bounded by constants for RBF, sigmoid, and tanh.

The extension of these ideas to nets with greater t
layers is not difficult, and leads to composite functio
~~the Taylor series (giv~~ing rise to backpropagation filt


11:29 AM Mon 13 Jun
13%
Multilayer_neural-net_Lewis_Liu
IEEE_WorkShop_Slides_Lavret...
AIntro
Adaptive_Control_Week12
Multilayer_neural-net_Lewi...
Survey-adapt+learn-Annaswamy
NPTEL

linear two-layer nets.

For notational convenience define the matrix of all the weights as

$$Z = \begin{bmatrix} W & 0 \\ 0 & V \end{bmatrix}. \tag{16}$$

A. Some Bounding Assumptions and Facts

Some required mild bounding assumptions are now stated. The two assumptions will be true in every practical situation, and are standard in the existing literature. The facts are easy to prove given the assumptions.

Assumption 1: The ideal weights are bounded by known positive values so that $\|V\|_F \le V_M, \|W\|_F \le W_M$, or

$$\|Z\|_F \le Z_M \tag{17}$$

with Z_M known.

Assumption 2: The desired trajectory is bounded in the sense, for instance, that

$$\left\| \begin{matrix} q_d \\ \dot{q}_d \\ \ddot{q}_d \end{matrix} \right\| \le Q_d \tag{18}$$

where $Q_d \in R$ is a known constant. ■

$$O(\tilde{V}^T x)^2 = [\sigma(V^t x) - \sigma(\hat{V}^T x)] - \sigma'(\hat{V}^T x)\tilde{V}^T x \tag{24}$$

we take the following.

Fact 4: For sigmoid, RBF, and tanh activation functions, the higher-order terms in the Taylor series are bounded by

$$\|O(\tilde{V}^T x)^2\| \le c_3 + c_4 Q_d \|\tilde{V}\|_F + c_5 \|\tilde{V}\|_F \|r\|$$

where c_i are computable positive constants. ■

Fact 4 is direct to show using (19), some standard norm inequalities, and the fact that $\sigma(\cdot)$ and its derivative are bounded by constants for RBF, sigmoid, and tanh.

The extension of these ideas to nets with greater than three layers is not difficult, and leads to composite function terms in the Taylor series (giving rise to backpropagation filtered error terms for the multilayer net case—see Theorem 3.1).

B. Controller Structure and Error System Dynamics

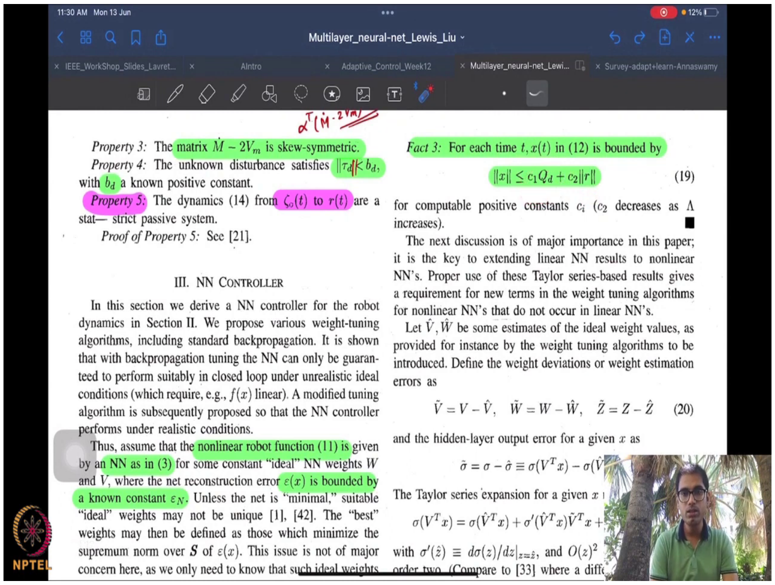
Define the NN functional estimate of (11) by

$$\hat{f}(x) = \hat{W}^T \sigma(\hat{V}^T x) \tag{25}$$

with $\hat{V}, \hat{W}$ the current (estimated) values of the ideal NN weights V, W as provided by the tuning algorithms subsequently to be discussed. With τ_o defined in (13),

control input

$$\tau = \tau_o - v = \hat{W}^T \sigma(\hat{V}^T x) + K_v r - v$$


11:30 AM Mon 13 Jun
12%
Multilayer_neural-net_Lewis_Liu
IEEE_WorkShop_Slides_Lavret...
AIntro
Adaptive_Control_Week12
Multilayer_neural-net_Lewi...
Survey-adapt+learn-Annaswamy
NPTEL

$\alpha^T(\dot{M} - 2V_m)$

Property 3: The matrix $\dot{M} - 2V_m$ is skew-symmetric.

Property 4: The unknown disturbance satisfies $\|\tau_d\| \le b_d$, with b_d a known positive constant.

Property 5: The dynamics (14) from $\zeta_o(t)$ to $r(t)$ are a stat— strict passive system.

Proof of Property 5: See [21].

III. NN Controller

In this section we derive a NN controller for the robot dynamics in Section II. We propose various weight-tuning algorithms, including standard backpropagation. It is shown that with backpropagation tuning the NN can only be guaranteed to perform suitably in closed loop under unrealistic ideal conditions (which require, e.g., $f(x)$ linear). A modified tuning algorithm is subsequently proposed so that the NN controller performs under realistic conditions.

Thus, assume that the nonlinear robot function (11) is given by an NN as in (3) for some constant "ideal" NN weights W and V, where the net reconstruction error $\varepsilon(x)$ is bounded by a known constant ε_N. Unless the net is "minimal," suitable "ideal" weights may not be unique [1], [42]. The "best" weights may then be defined as those which minimize the supremum norm over S of $\varepsilon(x)$. This issue is not of major concern here, as we only need to know that such ideal weights

Fact 3: For each time $t, x(t)$ in (12) is bounded by

$$\|x\| \le c_1 Q_d + c_2 \|r\| \tag{19}$$

for computable positive constants c_i (c_2 decreases as Λ increases). ■

The next discussion is of major importance in this paper; it is the key to extending linear NN results to nonlinear NN's. Proper use of these Taylor series-based results gives a requirement for new terms in the weight tuning algorithms for nonlinear NN's that do not occur in linear NN's.

Let $\hat{V}, \hat{W}$ be some estimates of the ideal weight values, as provided for instance by the weight tuning algorithms to be introduced. Define the weight deviations or weight estimation errors as

$$\tilde{V} = V - \hat{V}, \quad \tilde{W} = W - \hat{W}, \quad \tilde{Z} = Z - \hat{Z} \tag{20}$$

and the hidden-layer output error for a given x as

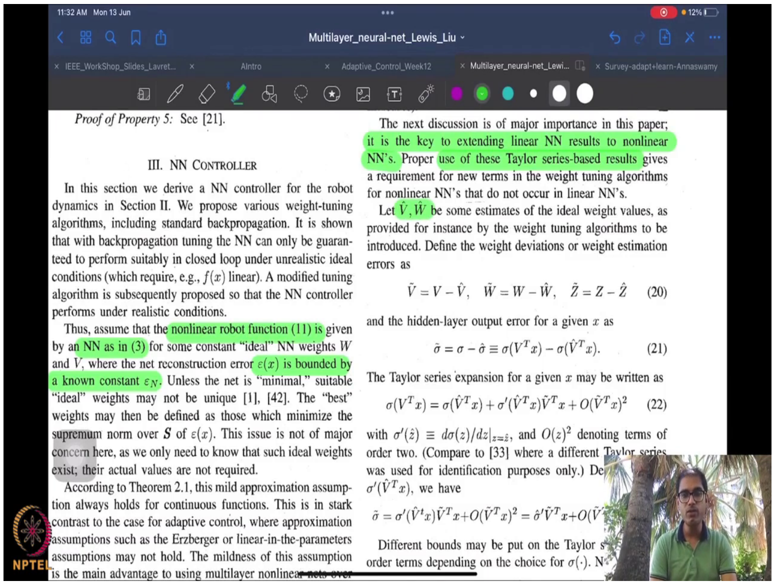
$$\tilde{\sigma} = \sigma - \hat{\sigma} \equiv \sigma(V^T x) - \sigma(\hat{V}$$

The Taylor series expansion for a given x

$$\sigma(V^T x) = \sigma(\hat{V}^T x) + \sigma'(\hat{V}^T x)\tilde{V}^T x +$$

with $\sigma'(\hat{z}) \equiv d\sigma(z)/dz|_{z=\hat{z}}$, and $O(z)^2$

order two. (Compare to [33] where a diff


11:32 AM Mon 13 Jun
Multilayer_neural-net_Lewis_Liu
IEEE_WorkShop_Slides_Lavret...
AIntro
Adaptive_Control_Week12
Multilayer_neural-net_Lewi...
Survey-adapt+learn-Annaswamy
12%
NPTEL

Proof of Property 5: See [21].

III. NN Controller

In this section we derive a NN controller for the robot dynamics in Section II. We propose various weight-tuning algorithms, including standard backpropagation. It is shown that with backpropagation tuning the NN can only be guaranteed to perform suitably in closed loop under unrealistic ideal conditions (which require, e.g., $f(x)$ linear). A modified tuning algorithm is subsequently proposed so that the NN controller performs under realistic conditions.

Thus, assume that the nonlinear robot function (11) is given by an NN as in (3) for some constant "ideal" NN weights W and V, where the net reconstruction error $\varepsilon(x)$ is bounded by a known constant ε_N. Unless the net is "minimal," suitable "ideal" weights may not be unique [1], [42]. The "best" weights may then be defined as those which minimize the supremum norm over S of $\varepsilon(x)$. This issue is not of major concern here, as we only need to know that such ideal weights exist; their actual values are not required.

According to Theorem 2.1, this mild approximation assumption always holds for continuous functions. This is in stark contrast to the case for adaptive control, where approximation assumptions such as the Erzberger or linear-in-the-parameters assumptions may not hold. The mildness of this assumption is the main advantage to using multilayer nonlinear nets over

The next discussion is of major importance in this paper; it is the key to extending linear NN results to nonlinear NN's. Proper use of these Taylor series-based results gives a requirement for new terms in the weight tuning algorithms for nonlinear NN's that do not occur in linear NN's.

Let $\hat{V}, \hat{W}$ be some estimates of the ideal weight values, as provided for instance by the weight tuning algorithms to be introduced. Define the weight deviations or weight estimation errors as

$$\tilde{V} = V - \hat{V}, \quad \tilde{W} = W - \hat{W}, \quad \tilde{Z} = Z - \hat{Z} \tag{20}$$

and the hidden-layer output error for a given x as

$$\tilde{\sigma} = \sigma - \hat{\sigma} \equiv \sigma(V^T x) - \sigma(\hat{V}^T x). \tag{21}$$

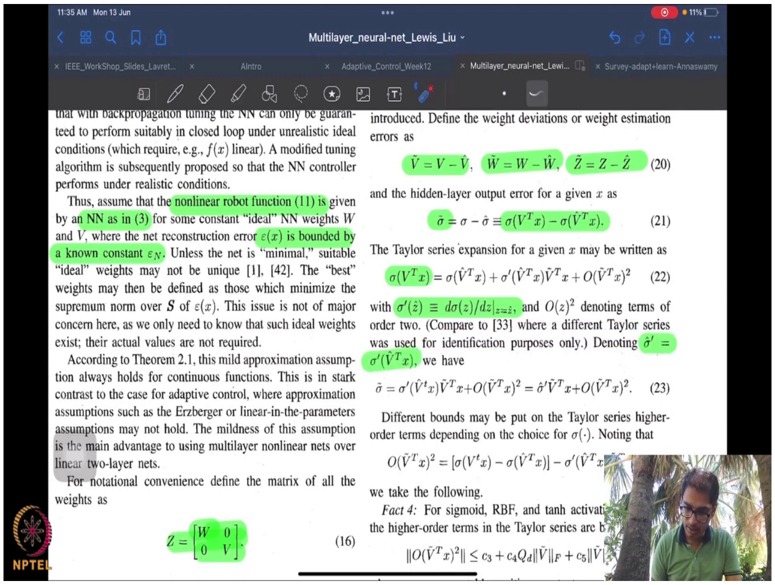
The Taylor series expansion for a given x may be written as

$$\sigma(V^T x) = \sigma(\hat{V}^T x) + \sigma'(\hat{V}^T x)\tilde{V}^T x + O(\tilde{V}^T x)^2 \tag{22}$$

with $\sigma'(\hat{z}) \equiv d\sigma(z)/dz|_{z=\hat{z}}$, and $O(z)^2$ denoting terms of order two. (Compare to [33] where a different Taylor series was used for identification purposes only.) De $\sigma'(\hat{V}^T x)$, we have

$$\tilde{\sigma} = \sigma'(\hat{V}^t x)\tilde{V}^T x + O(\tilde{V}^T x)^2 = \hat{\sigma}'\tilde{V}^T x + O(\tilde{V}$$

Different bounds may be put on the Taylor s order terms depending on the choice for $\sigma(\cdot)$. N


11:35 AM Mon 13 Jun
Multilayer_neural-net_Lewis_Liu
IEEE_WorkShop_Slides_Lavret...
AIntro
Adaptive_Control_Week12
Multilayer_neural-net_Lewi...
Survey-adapt+learn-Annaswamy
11%
NPTEL

that with backpropagation tuning the NN can only be guaranteed to perform suitably in closed loop under unrealistic ideal conditions (which require, e.g., $f(x)$ linear). A modified tuning algorithm is subsequently proposed so that the NN controller performs under realistic conditions.

Thus, assume that the nonlinear robot function (11) is given by an NN as in (3) for some constant "ideal" NN weights W and V, where the net reconstruction error $\varepsilon(x)$ is bounded by a known constant ε_N. Unless the net is "minimal," suitable "ideal" weights may not be unique [1], [42]. The "best" weights may then be defined as those which minimize the supremum norm over S of $\varepsilon(x)$. This issue is not of major concern here, as we only need to know that such ideal weights exist; their actual values are not required.

According to Theorem 2.1, this mild approximation assumption always holds for continuous functions. This is in stark contrast to the case for adaptive control, where approximation assumptions such as the Erzberger or linear-in-the-parameters assumptions may not hold. The mildness of this assumption is the main advantage to using multilayer nonlinear nets over linear two-layer nets.

For notational convenience define the matrix of all the weights as

$$Z = \begin{bmatrix} W & 0 \\ 0 & V \end{bmatrix}. \tag{16}$$

introduced. Define the weight deviations or weight estimation errors as

$$\tilde{V} = V - \hat{V}, \quad \tilde{W} = W - \hat{W}, \quad \tilde{Z} = Z - \hat{Z} \tag{20}$$

and the hidden-layer output error for a given x as

$$\tilde{\sigma} = \sigma - \hat{\sigma} \equiv \sigma(V^T x) - \sigma(\hat{V}^T x). \tag{21}$$

The Taylor series expansion for a given x may be written as

$$\sigma(V^T x) = \sigma(\hat{V}^T x) + \sigma'(\hat{V}^T x)\tilde{V}^T x + O(\tilde{V}^T x)^2 \tag{22}$$

with $\sigma'(\hat{z}) \equiv d\sigma(z)/dz|_{z=\hat{z}}$, and $O(z)^2$ denoting terms of order two. (Compare to [33] where a different Taylor series was used for identification purposes only.) Denoting $\hat{\sigma}' = \sigma'(\hat{V}^T x)$, we have

$$\tilde{\sigma} = \sigma'(\hat{V}^t x)\tilde{V}^T x + O(\tilde{V}^T x)^2 = \hat{\sigma}'\tilde{V}^T x + O(\tilde{V}^T x)^2. \tag{23}$$

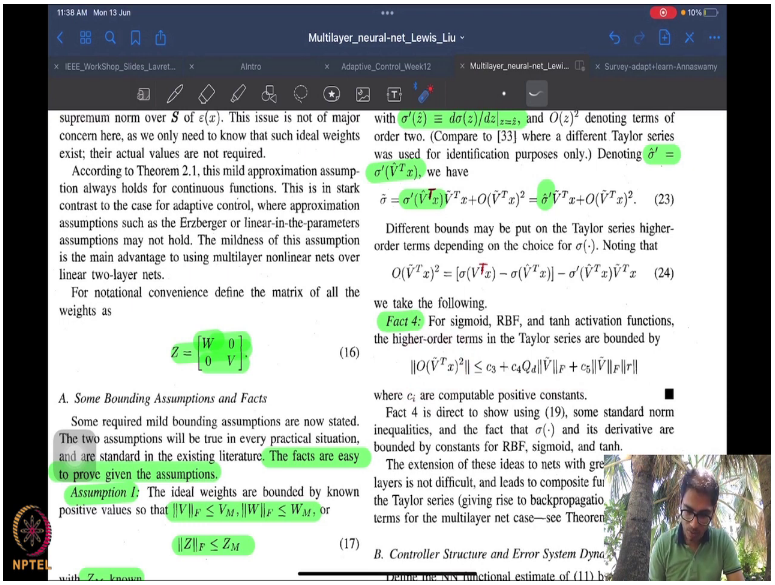
Different bounds may be put on the Taylor series higher-order terms depending on the choice for $\sigma(\cdot)$. Noting that

$$O(\tilde{V}^T x)^2 = [\sigma(V^t x) - \sigma(\hat{V}^T x)] - \sigma'(\hat{V}^T x$$

we take the following.

Fact 4: For sigmoid, RBF, and tanh activati the higher-order terms in the Taylor series are b

$$\|O(\tilde{V}^T x)^2\| \le c_3 + c_4 Q_d \|\tilde{V}\|_F + c_5 \|\tilde{V}\|$$


11:38 AM Mon 13 Jun
10%
Multilayer_neural-net_Lewis_Liu
IEEE_WorkShop_Slides_Lavret...
AIntro
Adaptive_Control_Week12
Multilayer_neural-net_Lewi...
Survey-adapt+learn-Annaswamy
NPTEL

supremum norm over S of $\varepsilon(x)$. This issue is not of major concern here, as we only need to know that such ideal weights exist; their actual values are not required.

According to Theorem 2.1, this mild approximation assumption always holds for continuous functions. This is in stark contrast to the case for adaptive control, where approximation assumptions such as the Erzberger or linear-in-the-parameters assumptions may not hold. The mildness of this assumption is the main advantage to using multilayer nonlinear nets over linear two-layer nets.

For notational convenience define the matrix of all the weights as

$$Z = \begin{bmatrix} W & 0 \\ 0 & V \end{bmatrix}. \tag{16}$$

A. Some Bounding Assumptions and Facts

Some required mild bounding assumptions are now stated. The two assumptions will be true in every practical situation, and are standard in the existing literature. The facts are easy to prove given the assumptions.

Assumption 1: The ideal weights are bounded by known positive values so that $\|V\|_F \le V_M, \|W\|_F \le W_M$, or

$$\|Z\|_F \le Z_M \tag{17}$$

with Z_M known

with $\sigma'(\hat{z}) \equiv d\sigma(z)/dz|_{z=\hat{z}}$, and $O(z)^2$ denoting terms of order two. (Compare to [33] where a different Taylor series was used for identification purposes only.) Denoting $\hat{\sigma}' = \sigma'(\hat{V}^T x)$, we have

$$\tilde{\sigma} = \sigma'(\hat{V}^T x)\tilde{V}^T x + O(\tilde{V}^T x)^2 = \hat{\sigma}'\tilde{V}^T x + O(\tilde{V}^T x)^2. \tag{23}$$

Different bounds may be put on the Taylor series higher-order terms depending on the choice for $\sigma(\cdot)$. Noting that

$$O(\tilde{V}^T x)^2 = [\sigma(V^T x) - \sigma(\hat{V}^T x)] - \sigma'(\hat{V}^T x)\tilde{V}^T x \tag{24}$$

we take the following.

Fact 4: For sigmoid, RBF, and tanh activation functions, the higher-order terms in the Taylor series are bounded by

$$\|O(\tilde{V}^T x)^2\| \le c_3 + c_4 Q_d \|\tilde{V}\|_F + c_5 \|\tilde{V}\|_F \|r\|$$

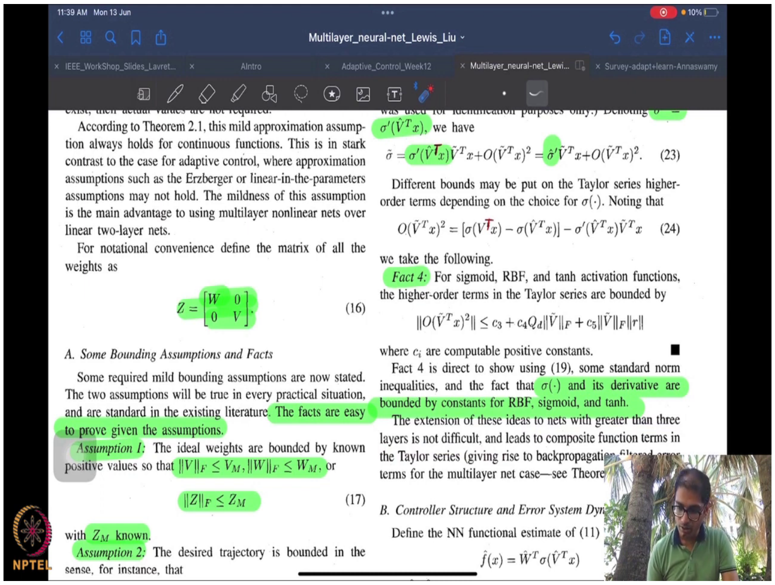
where c_i are computable positive constants. ■

Fact 4 is direct to show using (19), some standard norm inequalities, and the fact that $\sigma(\cdot)$ and its derivative are bounded by constants for RBF, sigmoid, and tanh.

The extension of these ideas to nets with gre layers is not difficult, and leads to composite fu the Taylor series (giving rise to backpropagatio terms for the multilayer net case—see Theoren

B. Controller Structure and Error System Dyna

Define the NN functional estimate of (11) b


11:39 AM Mon 13 Jun
10%
Multilayer_neural-net_Lewis_Liu
IEEE_WorkShop_Slides_Lavret...
AIntro
Adaptive_Control_Week12
Multilayer_neural-net_Lewi...
Survey-adapt+learn-Annaswamy
NPTEL

exist; their actual values are not required.

According to Theorem 2.1, this mild approximation assumption always holds for continuous functions. This is in stark contrast to the case for adaptive control, where approximation assumptions such as the Erzberger or linear-in-the-parameters assumptions may not hold. The mildness of this assumption is the main advantage to using multilayer nonlinear nets over linear two-layer nets.

For notational convenience define the matrix of all the weights as

$$Z = \begin{bmatrix} W & 0 \\ 0 & V \end{bmatrix}. \tag{16}$$

A. Some Bounding Assumptions and Facts

Some required mild bounding assumptions are now stated. The two assumptions will be true in every practical situation, and are standard in the existing literature. The facts are easy to prove given the assumptions.

Assumption 1: The ideal weights are bounded by known positive values so that $\|V\|_F \le V_M, \|W\|_F \le W_M$, or

$$\|Z\|_F \le Z_M \tag{17}$$

with Z_M known.

Assumption 2: The desired trajectory is bounded in the sense, for instance, that

was used for identification purposes only.) Denoting $\hat{\sigma}' =$ $\sigma'(\hat{V}^T x)$, we have

$$\tilde{\sigma} = \sigma'(\hat{V}^T x)\tilde{V}^T x + O(\tilde{V}^T x)^2 = \hat{\sigma}'\tilde{V}^T x + O(\tilde{V}^T x)^2. \tag{23}$$

Different bounds may be put on the Taylor series higher-order terms depending on the choice for $\sigma(\cdot)$. Noting that

$$O(\tilde{V}^T x)^2 = [\sigma(V^T x) - \sigma(\hat{V}^T x)] - \sigma'(\hat{V}^T x)\tilde{V}^T x \tag{24}$$

we take the following.

Fact 4: For sigmoid, RBF, and tanh activation functions, the higher-order terms in the Taylor series are bounded by

$$\|O(\tilde{V}^T x)^2\| \le c_3 + c_4 Q_d \|\tilde{V}\|_F + c_5 \|\tilde{V}\|_F \|r\|$$

where c_i are computable positive constants. ■

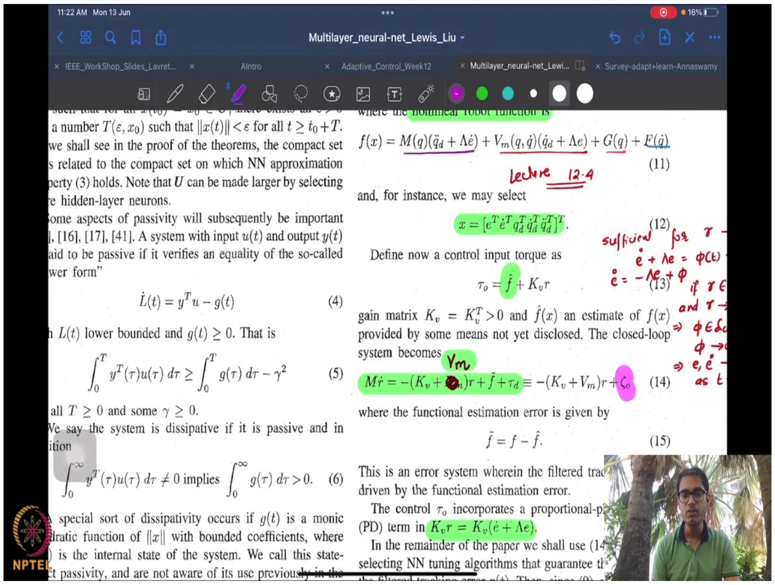
Fact 4 is direct to show using (19), some standard norm inequalities, and the fact that $\sigma(\cdot)$ and its derivative are bounded by constants for RBF, sigmoid, and tanh.

The extension of these ideas to nets with greater than three layers is not difficult, and leads to composite function terms in the Taylor series (giving rise to backpropagati terms for the multilayer net case—see Theore

B. Controller Structure and Error System Dyn

Define the NN functional estimate of (11)

$$\hat{f}(x) = \hat{W}^T \sigma(\hat{V}^T x)$$


11:22 AM Mon 13 Jun
Multilayer_neural-net_Lewis_Liu
IEEE_WorkShop_Slides_Lavret...
AIntro
Adaptive_Control_Week12
Multilayer_neural-net_Lewi...
Survey-adapt+learn-Annaswamy
16%
NPTEL

a number $T(\varepsilon, x_0)$ such that $\|x(t)\| < \varepsilon$ for all $t \geq t_0 + T$. we shall see in the proof of the theorems, the compact set s related to the compact set on which NN approximation perty (3) holds. Note that *U* can be made larger by selecting e hidden-layer neurons.

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

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

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

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

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

Ve say the system is dissipative if it is passive and in ition

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

special sort of dissipativity occurs if $g(t)$ is a monic dratic function of $\|x\|$ with bounded coefficients, where) is the internal state of the system. We call this statet passivity, and are not aware of its use previously in the

where the nonlinear robot function is

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

Lecture 12.4

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

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

sufficient for r

$\dot{e} + \Lambda e = \phi(t)$

$\dot{e} = -\Lambda e + \phi$

if $r \in$

and $r \rightarrow$

$\Rightarrow \phi \in \mathcal{L}$

$\phi \rightarrow$

$\Rightarrow e, \dot{e}$

as t

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

$$M\dot{r} = -(K_v + \overset{V_m}{V_m})r + \tilde{f} + \tau_d \equiv -(K_v + V_m)r + \zeta_o \tag{14}$$

where the functional estimation error is given by

$$\tilde{f} = f - \hat{f}. \tag{15}$$

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

The control τ_o incorporates a proportional-p (PD) term in $K_v r = K_v(\dot{e} + \Lambda e)$.

In the remainder of the paper we shall use (14 selecting NN tuning algorithms that guarantee th

And the third property which is a very important property in the Lyapunov analysis is that the matrix M dot minus twice V m is skew-symmetric. What does it mean? It means that any quadratic form which is alpha transpose M dot minus 2 V m alpha is 0. So, the, so the quadratic form corresponding to any skew-symmetric matrix is always 0. So, that is what we have in property number 3. Of course, we assume that the disturbances are bounded. So, norm is missing here but you assume that the disturbances are bounded.

This property five, we do not use yet. I mean we do not talk about it. So, we, so we are, which is where there is a psi naught and r. That is why you had the psi naught here. Sorry, the zeta naught here. So, if you remember, this is zeta naught, that was sort of inserted here. The purpose was to talk about passivity. But you remember that we said that we will not discuss the passivity aspects of this article in these series. So, as of now we skip it. If we need to, we will talk about it later.

This is a property. So, this is not an assumption. So, these are properties. So, it is not like we are assuming anything. So, of course this property can also be proved. There is a nice reference that is given for this. So, now we know that we need only good behavior of the r variable and we have the dynamics of r given by this equation 14 here, which is also something nice.

Now, if f tilde was 0 and tau d was 0, then this is well known to be a stable, asymptotically stable system. So, if both of these are 0, then you are in very good shape, no problem. Now, the issues happen when this is non-zero. And of course, there is disturbance. So, of course that also, if it is non-zero, then this is at least bounded. So, you want some bounded performance.

So, what you need at the least is that you have a some nice bound on this f tilde. So, this is the least you want. So, you want a good function approximator. So, that is why you start talking about the neural network control. That is where we start discussing the neural network control. So, that is what we say, that this nonlinear robot function, if you remember, 11, so this guy, this whole thing, assume that it is given by a neural network as in 3.

So, it is essentially like a, the function f x is replaced by this kind of a weights thing, I mean you had it here, you had it here. Suppose this function f x is approximated in this manner, where of course epsilon is small enough. And then you have these weights and offsets and so on and this activation function. So, it is like a, approximated by a three-layer neural network.

Now, why can we do this, because we, by this nice Theorem 2.1, we have that you can approximate almost any continuous function in this way. And f x is of course continuous. So, of course, we have this fact that this error is bounded by some constant epsilon n. So, you always need bounds on the errors side. If you do not have bounds on the errors, then you can not get bounded performance even with your Lyapunov analysis.

So, of course there is results which say that these ideal weights are not necessarily unique. So, I mean, that is, one cannot expect, like, when in standard adaptive control problem, the value of the parameter is sort of fixed and known. For example, if you say the mass of your drone is unknown and you use an adaptive controller, so the mass is a fixed quantity, you know what the mass is.

Well, you do not know what the mass is, for example, but you know that it is a fixed quantity. But in this case, that is not necessarily the case because these are not masses and properties of the system but these are more like weights to some kind of sigmoidal function, which is used to approximate actual nonlinear function. So, therefore the choices are not necessarily unique. So, we should keep this in mind.

So, of course this according to Theorem 2.1, this mild approximation assumption always holds for continuous functions. So, anyway. So, this is of course, this is of course, the important part here. So, then of course, just for, we define a new notation, which is this z equal to W 0 0 V. This is just to make our lives easy in terms of notation.

Now, before doing any actual neural network, type of design, what we want to do is to look at a few different bounding assumptions and facts. And some of them are assumptions, of course, and some of them are facts. That is to say that some of them are guaranteed to hold true and others have to be based on some assumptions.

So, the first is an assumption. The facts are easy to prove given the assumptions. Therefore, we start with the assumptions. The first assumption is that the weights are bounded. This is a very fair assumption. I mean if the weights are not bounded, then possibly your function x is not ground, function f x is not bounded. So, having bounded weights is a very reasonable assumption.

The next is that the desired trajectory is also bounded. So, desired trajectory is defined by q d. But here, we talk not just about q d, but about all the derivatives or at least two derivatives of q d also. Now, this is again no different from the assumption which we made in our previous adaptive control problems. We just said it in words. We said that you have a bounded trajectory with bounded derivatives.

So, here we are making it a little bit more specific, we have taken q d, and its two derivatives. So, it is again, a very reasonable assumption. You do not want your system to be following trajectories which are too sharp or too jerky and so on. So, therefore you do assume that you have bounded reference with bounded derivatives. So, this is again something very reasonable.

The important thing here is of course that this is a known constant. So, you know this value. And similarly, here, also you know these terms. So, the important thing is that you know these bounds. In case of trajectory, of course this is not difficult, but in case of weights, knowing a bound is sort of the kind of thing we assumed when we talked about projection in adaptive control. So, in this case, we are posteriorly assuming that there is a known bound on the variables, that we are trying to learn.

Then we come to the first fact. It says that for each time t, x is bounded in this way. So, x is bounded means it is not like a uniform bound or anything, but we know that x contains of, consists of what? x is, x consists of q, well actually, I should probably point it out here. So, if you look at this, I am sorry, we define x here.

So, the x is defined this way. And so, x contains what, it contains q d and its two derivatives. So, these we already know are bounded by the cap q d, by our assumption and then the error contains e and e dot. So, these contain what? These contain again q and q d dots, q and q d and q d dots and q dots. And this is of course bounded by some r along with some q d. So, this is not difficult to see because you can actually get this sort of a bound on x, like a linear bound with respect to, a fine bound with respect to r.

So, now we sort of want to talk about using some kind of Taylor series approximations so that we can in fact do, deal with these nonlinear regressor parameter form. Remember, the linearity in the regressor parameter forms is what helped us design most of our adaptive controllers. Without that, we would be in quite a soup.

So, this is what. I mean, it is e for extending linear NN to nonlinear NNs. And this requires use of Taylor series based results. So, this is what is important. I mean, eventually when we have a nonlinear regressive parameter from, the nonlinear terms in the neural network, we still want to look at some linear versions and we do it by taking Taylor series approximations of these nonlinear terms.

Because without the linearity, it would be impossible to deal with the structure that comes about and different kind of nonlinearities that come about. And it would be impossible to get any kind of stabilization results. So, we of course, as always, assume that V hat and W hat are estimates for the ideal weights. And we of course define the tilde versions for V, W and z.

And of course, we, we also have the sort of error in the activation function values. This is defined using this notation as sigma applied on V transpose x and sigma applied on V hat transpose x. So, this is again, a notation. So, what we want to do is to sort of expand this into, expand this with some kind of linear terms. And that is the whole idea. We want to use the Taylor series.

So, Taylor series gives us some linear terms. This is standard when we linearize nonlinear systems also. So, what is the Taylor series expansion of say this V transpose x. We know that V can be written as V tilde plus V cap. So, we take the V cap as the center, sort of, and, so we have sigma V cap transpose x.

And then, you take the, sort of, you sort of take the derivative with respect to your V cap transpose, this V cap transpose x term. So, you take the partial of the derivative with respect to x and that is what is this sigma prime notation. And then, you have multiply use the error term which is this V tilde transpose x. And then you have all the second and sorry, the higher order terms here.

I am sorry, I think we, right, have the higher order terms right here. So, of course this is defined. So, this is defined as a partial with respect to some, sigma prime is defined as derivative with respect to some z, evaluated at z hat and so on. And o z square of course define higher order terms, and this is all well known.

So, of course there are some, I mean the authors point to some other reference where a different Taylor series was used, but that is okay. So, now of course, we, the authors want to simplify notation. So, they call this term as simply sigma hat prime. Why sigma hat prime? Because this sigma prime evaluated with this hat term. So, that is the rationale for using this kind of a notation.

And so, what do we have? We have sigma tilde as, sigma minus sigma cap. And so, sigma, we use this kind of an expansion here. So, what do you have? So, sigma cap also contains, sigma cap is essentially this. So, whenever I subtract, if I subtract these two, I just get this term. And that is sigma prime V hat transpose. So, let me fix this, V hat transpose x. And this V tilde transpose x. So, you get these two terms right here.

And then you have a second order term. So, this is of course simplified. We just call it sigma hat prime, and this is v tilde transpose x. So, this, this, this term is just to, I mean I will just mark it. So, this term is just this whole guy. And it is just a shorthand, it is nothing other than that. So, now, there is also the requirement to put some kind of bounds on the higher order terms. And so, we, I mean the authors do do that. So, this kind of a bound is sort of used. And again, this is, I believe transpose, And, so this is like this kind of a bound. It is, so let me see. I am trying to see if this is any different from what we have, different bounds are put on there.

So, this term is bounded with again, the sigma tilde because this looks like sigma tilde. So, essentially this, I mean honestly speaking, this bound is essentially derived from this equation, because if I take this on one side, I simply have this subtracting this. And this is sigma tilde, this is essentially sigma tilde. And then you have this guy here. So, this is what you have. So, this, if this guy is to be evaluated, then you take this to the other side. So, you have sigma tilde, which is this, minus this whole thing. It is this whole thing. So, now in order to get some kind of a reasonable bound for this term, you use this expression 24.

So, you have a Fact 4, which is that for sigmoid, RBF and tan hyperbolics, all the three activation functions that we looked at, you have this to be bounded in this way. And these are of course, obtained using some kind of bounds on these guys. So, this is somehow using that. So, because you see that these bounds also contain the Frobenius norm of V tilde itself. So, these bonds are obtained using this expression right here. So, that is the whole idea.

So, of course they also use the, not just the, they also use the fact that sigma and its derivative are bounded by constants for the RBF, sigmoidal and tan hyperbolics. So, of course, I mean the author has mentioned that, for nets greater than three, this kind of bounding and these kind of ideas are not too difficult to extend either, and then it can be done.

So, what did we look at in this session is that we had already seen the robot dynamics. So, we saw a little bit more of sort of how the control structure looks in today's session. We also saw how the nonlinear robot function looks, and we understood why r going to 0 is sufficient for our purposes.

And finally, we, once we understood that this f tilde that is the function approximation error is what we want to drive as close to 0 as possible, we understood that this is where the neural network comes in. So, this neural network is used to approximate this function using this theorem that we looked at earlier.

We know that, we sort of know that this function approximation using neutral networks will help us to get close to true value of f. Therefore, f tilde can be made to have like an epsilon n bound. And that is what we looked at, at how this neural network approximation will work, how we use the Taylor series to sort of linearize the terms in the neural network and bound the nonlinear terms.

And this is what we will use in the subsequent analysis to define these parameter estimation and parameter learning and these parameters, of course, the weights of the neural network, and we will hopefully be able to show that you have nice, stable performance the way you require. So, we will continue with our discussion in the subsequent session, and I hope to see you again soon. Thank you.