Lecture – 37

Neuro-Adaptive Design – II:

A Robustifying Tool for Any Design

Dr. Radhakant Padhi

Asst. Professor

Dept. of Aerospace Engineering

Indian Institute of Science - Bangalore

Motivation

- Perfect system modeling is difficult
- Sources of imperfection
  - Unmodelled dynamics (missing algebraic terms in the model)
  - Inaccurate knowledge of system parameters
  - Change of system parameters/dynamics during operation
- “Black box” adaptive approaches exist. But, making use of existing design is better!

  (faster adaptation, chance of instability before adaptation is minimal)
- The adaptive design should preferably be compatible with “<u>any nominal control</u>” design

Reference

Radhakant Padhi, Nishant Unnikrishnan and S. N. Balakrishnan, "*Model Following Neuro-Adaptive Control Design for Non-square, Non-affine Nonlinear Systems*", IET Control Theory and Applications, Vol. 1 (6), Nov 2007, pp.1650-1661.

Modeling Inaccuracy: A Simple Example

$$\dot{x} = \underbrace{2\sin(x)}_{\text{Known part of actual system (nominal system)}} + \underbrace{0.1\sin(x)}_{\text{Unknown part of actual system}}$$

$$= 2\sin(x) + \underbrace{\Delta c}_{\textit{Weight}}\,\underbrace{\sin(x)}_{\textit{Basis Function}}$$

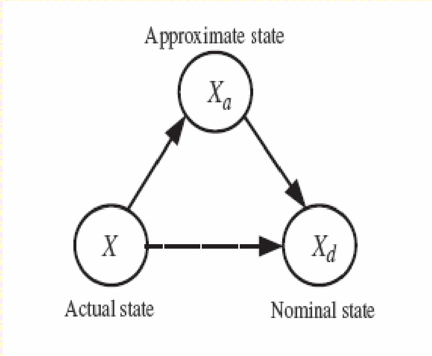
Objective:

To increase the robustness of a “nominal controller” with respect to parameter and/or modeling inaccuracies, which lead to imperfections in the system model.

Problem Description and Strategy

- Desired Dynamics: $\dot{X}_d = f\left(X_d, U_d\right)$
- Actual Plant: $\dot{X} = f(X,U) + d(X)$

  (unknown)



- Goal: $\boxed{X \to X_d, \quad as \quad t \to \infty}$
- Approximate System: $\dot{X}_a = f(X,U) + \hat{d}(X) + K_a\left(X - X_a\right)$

  $\left(K_a > 0\right)$

  NN Approx.

- Strategy: $\boxed{X \to X_a \to X_d, \quad as \quad t \to \infty}$

Steps for assuring $\boxed{X_a \rightarrow X_d}$

- Select a gain matrix $K > 0$ such that

$$\dot{E}_d + K\,E_d = 0, \qquad E_d \triangleq \left(X_a - X_d\right)$$

- This leads to

$$\left\{f\left(X,U\right)+\hat{d}\left(X\right)+K_a\left(X-X_a\right)\right\}-f\left(X_d,U_d\right)+K\left(X_a-X_d\right)=0$$

$$f\left(X,U\right)=\left\{f\left(X_d,U_d\right)-\hat{d}\left(X\right)-K_a\left(X-X_a\right)-K\left(X_a-X_d\right)\right\}$$

$$i.e. \quad f\left(X,U\right)=h\left(X,X_a,X_d,U_d\right)$$

- Solve for the control U

Control Solution:
(No. of controls = No. of states)

- Affine Systems: $f(X)+\left[g(X)\right]U = h\left(X, X_d, X_a, U_d\right)$

$$\boxed{U = \left[g(X)\right]^{-1}\left\{ h\left(X, X_d, X_a, U_d\right) - f(X) \right\}}$$

- Non-affine Systems: $f(X,U) = h\left(X, X_d, X_a, U_d\right)$

Use Numerical Method

(e.g. N-R Technique)

$$\left(U_{guess}\right)_k = \begin{Bmatrix} U_d & : & k=1 \\ U_{k-1} & : & k=2,3,\ldots \end{Bmatrix}$$

Control Solution:
(No. of controls < No. of states)

- Modify X_a dynamics:

$$\dot{X}_a = f(X,U) + \left[\hat{d}(X) - \Psi(X)U_s\right] + \Psi(X)U_s + K_a(X - X_a)$$
$$= f(X,U) + \hat{d}_a(X) + \Psi(X)U_s + K_a(X - X_a)$$

- Solve for the control from:

$$\left\{f(X,U) + \hat{d}_a(X) + \Psi(X)U_s + K_a(X - X_a)\right\} - f(X_d,U_d) + K(X_a - X_d) = 0$$

$$f(X,U) + \Psi(X)U_s = \left\{f(X_d,U_d) - \hat{d}_a(X) - K_a(X - X_a) - K(X_a - X_d)\right\}$$
$$= h(X, X_a, X_d, U_d)$$

Solution for affine systems:
(No. of controls < No. of states)

$$\left\{ f(X) + g(X)U \right\} + \Psi(X)U_s = h(X, X_a, X_d, U_d)$$

$$f(X) + \begin{bmatrix} g(X) & \Psi(X) \end{bmatrix} \begin{bmatrix} U \\ U_s \end{bmatrix} = h(X, X_a, X_d, U_d)$$

$$G(X)\; V = -f(X) + h(X, X_a, X_d, U_d)$$

$$V = \left[G(X) \right]^{-1} \left\{ -f(X) + h(X, X_a, X_d, U_d) \right\}$$

Extract U from V

For simplicity, we will not consider this special case in our further discussion.

Steps for assuring $\boxed{X \rightarrow X_a}$

- Error: $E_a \triangleq (X - X_a), \quad e_{a_i} \triangleq \left(x_i - x_{a_i}\right)$
- Error Dynamics:

$$\dot{x}_i = f_i\left(X, U\right) + d_i\left(X\right)$$

$$\dot{x}_{a_i} = f_i\left(X, U\right) + \hat{d}_i\left(X\right) + k_{a_i} e_{a_i}$$

$$\dot{e}_{a_i} = \dot{x}_i - \dot{x}_{a_i}$$

$$= \left[d_i\left(X\right) - \hat{d}_i\left(X\right)\right] - k_{a_i} e_{a_i}$$

$$= \left\{W_i^T \Phi_i\left(X\right) + \varepsilon_i\right\} - \hat{W}_i^T \Phi_i\left(X\right) - k_{a_i} e_{a_i}$$

$$\dot{e}_{a_i} = \tilde{W}_i^T \Phi_i\left(X\right) + \varepsilon_i - k_{a_i} e_{a_i}$$

Ideal neural network

$$d_i(X) = W_i^T \varphi_i(X) + \varepsilon_i$$

Actual neural network

$$\hat{d}_i(X) = \hat{W}_i^T \varphi_i(X)$$

$$\left(\tilde{W}_i \triangleq W_i - \hat{W}_i\right)$$

Stable Function Learning

Lyapunov Function Candidate

$$L_i = \frac{1}{2}\left(p_i e_{a_i}^2\right) + \frac{1}{2\gamma_i}\left(\tilde{W}_i^T \tilde{W}_i\right) \qquad \left(p_i, \gamma_i > 0\right)$$

Derivative of Lyapunov Function

$$\dot{L}_i = p_i e_{a_i} \dot{e}_{a_i} + \frac{1}{\gamma_i} \tilde{W}_i^T \dot{\tilde{W}}_i$$

$$= p_i e_{a_i} \left(\tilde{W}_i^T \Phi_i(X) + \varepsilon_i - k_{a_i} e_{a_i}\right) - \frac{1}{\gamma_i} \tilde{W}_i^T \dot{\hat{W}}_i \qquad \left(\because \tilde{W}_i \triangleq W_i - \hat{W}_i\right)$$

$$= \tilde{W}_i^T \underbrace{\left(p_i e_{a_i} \Phi_i(X) - \frac{1}{\gamma_i} \dot{\hat{W}}_i\right)}_{\to 0} + p_i e_{a_i} \varepsilon_i - k_{a_i} p_i e_{a_i}^2$$

Stable Function Learning

Weight update rule (Neural network training)

$$\boxed{\dot{\hat{W}}_i = \gamma_i \, p_i e_{a_i} \Phi_i \left(X \right)}$$

Derivative of Lyapunov Function

$$\dot{L}_i = p_i e_{a_i} \varepsilon_i - k_{a_i} \, p_i e_{a_i}^2$$

$$\boxed{\dot{L}_i < 0 \quad \text{if} \quad \left| e_{a_i} \right| > \left(\left| \varepsilon_i \right| / k_{a_i} \right)}$$

The system is “Practically Stable”

Neuro-adaptive Design: Implementation of Controller

Weight update rule:

$$\dot{\hat{W}}_i = \gamma_i p_i e_{a_i} \Phi_i, \qquad \hat{W}_i(0) = 0$$

where, γ_i : Learing rate

$$e_{a_i} = x_i - x_{a_i}$$

Φ_i : Basis function

Estimation of unknown function:

$$\hat{d}(X) = \hat{W}^T \Phi_i$$

Neuro-adaptive Design: Implementation of Controller

- Desired Dynamics: $\dot{X}_d = f\left(X_d, U_d\right)$

- Actual Plant: $\dot{X} = f(X,U) + \overbrace{d(X)}^{\text{(unknown)}}$

In reality, X(t) should be available from sensors and filters!

- Approximate System:

NN Approximation

$$\dot{X}_a = f(X,U) + \hat{d}(X) + K_a\left(X - X_a\right), \qquad K_a > 0\left(pdf\right)$$

- Initial Condition: $X_d\left(0\right) = X_a\left(0\right) = X\left(0\right) : \text{known}$

Example – 1

A Motivating Scalar Problem

Dr. Radhakant Padhi
Asst. Professor
Dept. of Aerospace Engineering
Indian Institute of Science - Bangalore

Example – 1: A scalar problem

- System dynamics (nominal system)

$$\dot{x}_d = \left(x_d + x_d{}^2\right) + \left(1 + x_d{}^2\right) u_d$$

- System dynamics (actual system)

$$\dot{x} = \left(x + x^2\right) + \left(1 + x^2\right) u + d\left(x\right)$$

$$d\left(x\right) = \sin\left(\pi x / 2\right)$$

$$\left(\text{unknown for control design}\right)$$

- Problem objectives:

* Nominal control design: $x_d \to 0$

* Adaptive control design: $x \to x_d$

Note: The objective $x \to x_d$ should be achieved much faster than $x_d \to 0$

Example – 1: A scalar problem

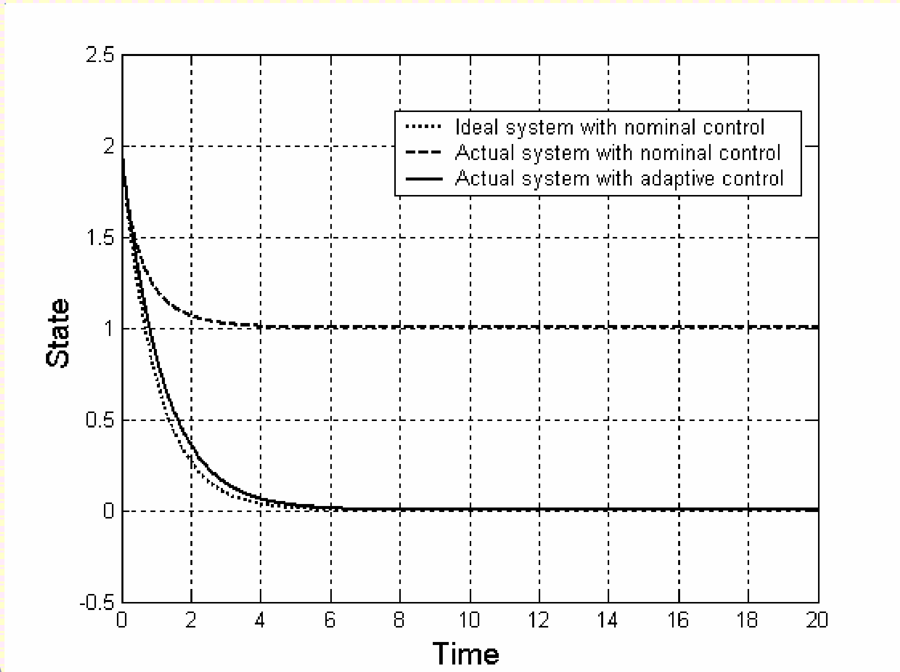
- Nominal control (dynamic inversion) $\left(\dot{x}_d - 0\right) + \left(1/\tau_d\right)\left(x_d - 0\right) = 0 \qquad \left(\tau_d = 1\right)$
- Nominal control $u_d = -\left(1 + x_d^{\ 2}\right)^{-1}\left(x_d + x_d^{\ 2} + x_d\right)$
- Adaptive control $u = \dfrac{1}{1+x^2}\begin{bmatrix}\left(x_d + x_d^{\ 2}\right) + \left(1 + x_d^{\ 2}\right)u_d - k\left(x_a - x_d\right) \\ -\left(x + x^2\right) - \hat{d}\left(x\right) - k_a\left(x - x_a\right)\end{bmatrix}$
- Design parameters

$$k = 2.5 \qquad k_a = 1 \qquad p = 1 \qquad \gamma = 30$$

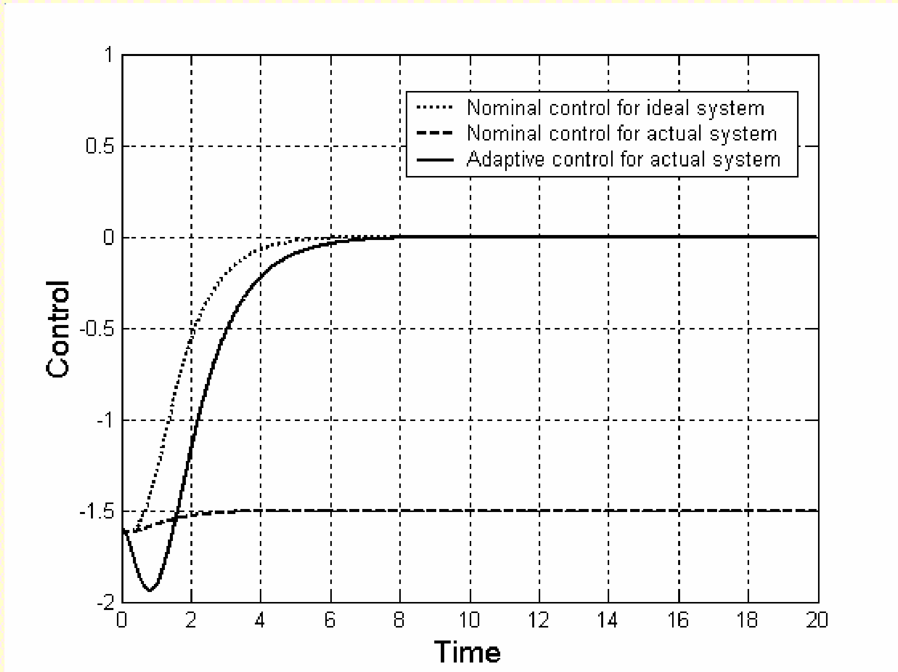
$$\Phi\left(x\right) = \left[\left(x/x_0\right) \quad \left(x/x_0\right)^2 \quad \left(x/x_0\right)^3\right]^T$$

Example – 1:
A scalar problem

State Trajectory

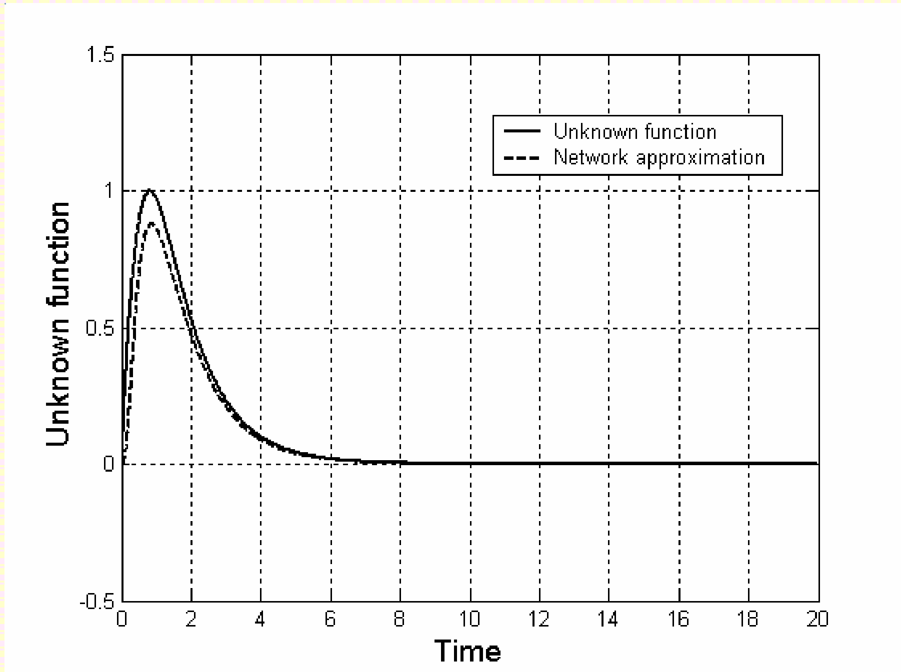


Control Trajectory



Example – 1:
A scalar problem

Approximation of the unknown function

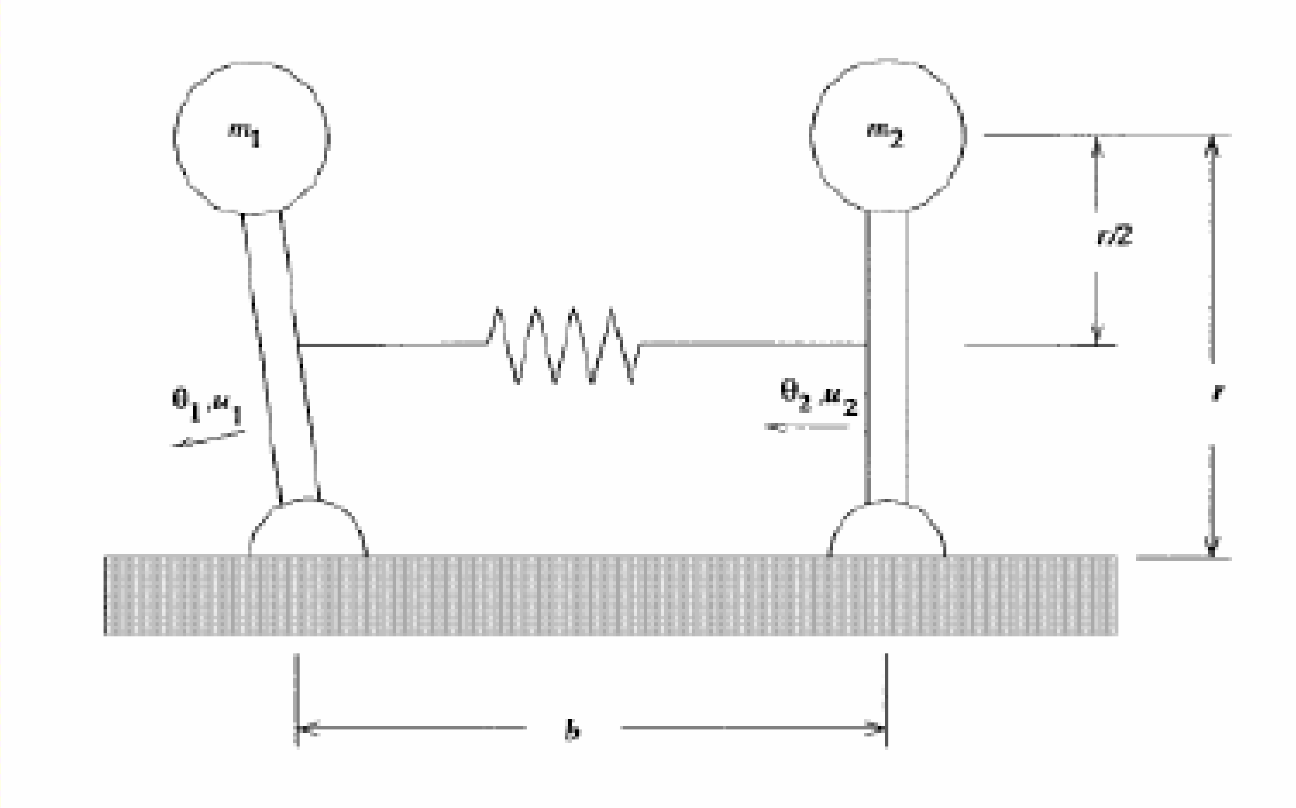


Example – 2

Double Inverted Pendulum: A Benchmark Problem

Dr. Radhakant Padhi
Asst. Professor
Dept. of Aerospace Engineering
Indian Institute of Science - Bangalore

Example – 2: Double inverted pendulum



Example – 2: Double inverted pendulum

- Nominal Plant:

$$\dot{x}_1^1 = x_1^2$$

$$\dot{x}_1^2 = \alpha_1 \sin(x_1^1) + \frac{kr}{2J_1}(l-b) + \left(\frac{u_{1_{\max}}}{J_1}\right)\tanh(u_1) + \left(\frac{kr^2}{4J_1}\right)\sin(x_2^1)$$

$$\dot{x}_2^1 = x_2^2$$

$$\dot{x}_2^2 = \alpha_2 \sin(x_2^1) + \frac{kr}{2J_2}(l-b) + \left(\frac{u_{2_{\max}}}{J_2}\right)\tanh(u_2) + \left(\frac{kr^2}{4J_2}\right)\sin(x_1^1)$$

- Actual Plant:

$$\dot{x}_1^1 = x_1^2$$

$$\dot{x}_1^2 = (\alpha_1 + \Delta\alpha_1)\sin(x_1^1) + \frac{kr}{2J_1}(l-b) + \frac{u_{1_{\max}}\tanh(u_1)}{J_1} + \frac{kr^2}{4J_1}\sin(x_2^1) + K_{m1}e^{a_1 x_1^1}$$

$$\dot{x}_2^1 = x_2^2$$

$$\dot{x}_2^2 = (\alpha_2 + \Delta\alpha_2)\sin(x_2^1) + \frac{kr}{2J_2}(l-b) + \frac{u_{2_{\max}}\tanh(u_2)}{J_2} + \frac{kr^2}{4J_2}\sin(x_1^1) + K_{m2}e^{a_2 x_2^1}$$

Example – 2: Double inverted pendulum

$$\alpha_i \triangleq \left(\frac{m_i g r}{J_i} - \frac{k r^2}{4 J_i} \right)$$

$$\beta_i \triangleq \frac{kr}{2J_i}(l - b)$$

$$\xi_i \triangleq \frac{u_{i_{\max}}}{J_i}$$

$$\sigma_i \triangleq \frac{kr^2}{4J_i}$$

System Parameter	Value	Units
End mass of pendulum 1 (m_1)	2	kg
End mass of pendulum 2 (m_2)	2.5	kg
Moment of inertia (J_1)	0.5	$kg\ m^2$
Moment of inertia (J_2)	0.625	$kg\ m^2$
Spring constant of connecting spring (k)	100	N/m
Pendulum height (r)	0.5	m
Natural length of spring (l)	0.5	m
Gravitational acceleration (g)	9.81	m/s^2
Distance between pendulum hinges (b)	0.4	m
Maximum torque input ($u_{1,\max}$)	20	Nm
Maximum torque input ($u_{2,\max}$)	20	Nm

Example – 2: Double inverted pendulum

- Parameters in unknown function

$$\Delta\alpha_1, \Delta\alpha_2 : 20\% \text{ off} \qquad a_1 = a_2 = 0.01 \quad K_{m_1} = K_{m_2} = 0.1$$

- Control design parameters

$$K = 0.2\, I_4, \quad K_a = I_4$$

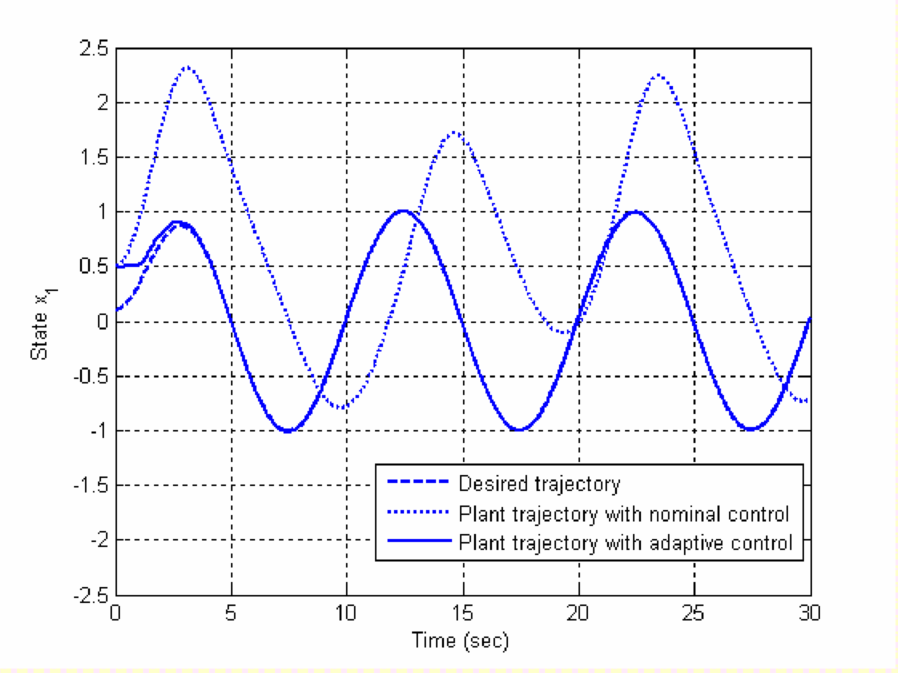
$$\psi(X) = \begin{bmatrix} -10 & 10 & 0 & 0 \\ 10 & -10 & 10 & -10 \end{bmatrix}^T$$

$$p_2 = p_4 = 1 \qquad \Phi_2(X) = \begin{bmatrix} 1 & x_1/1! & \cdots & x_1^{17}/17! & 1 & x_2/1! & \cdots & x_2^{17}/17! \end{bmatrix}^T$$

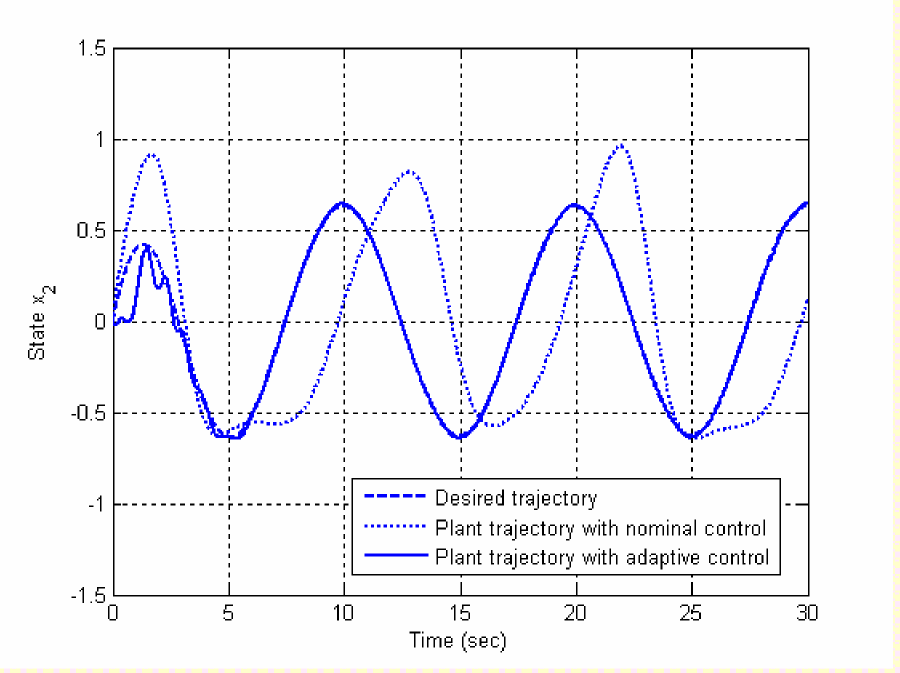
$$\gamma_2 = \gamma_4 = 20 \qquad \Phi_4(X) = \begin{bmatrix} 1 & x_3/1! & \cdots & x_3^{17}/17! & 1 & x_4/1! & \cdots & x_4^{17}/17! \end{bmatrix}^T$$

Example – 2: Double inverted pendulum

Position of Mass – 1

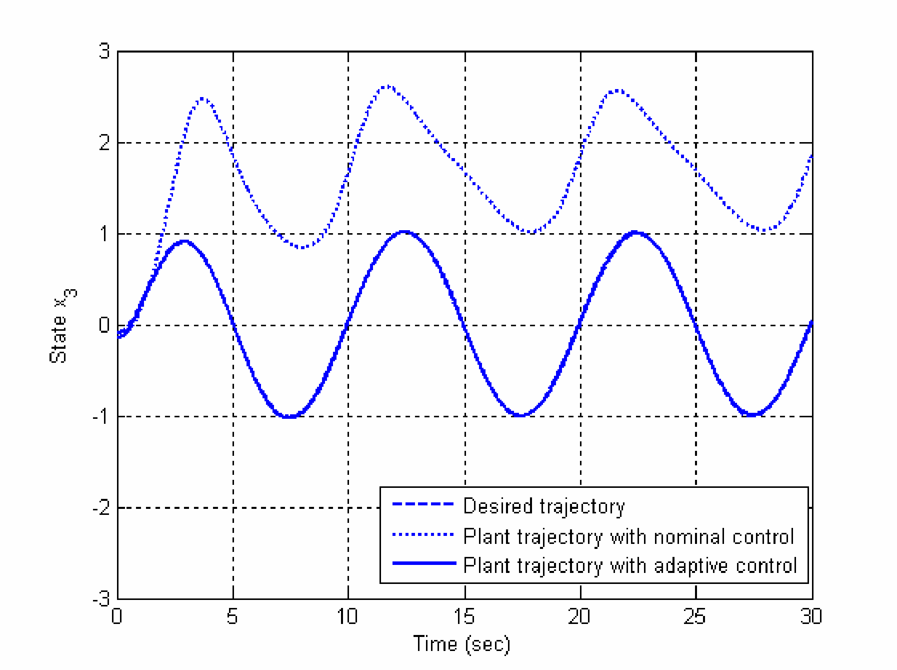


Velocity of Mass – 1

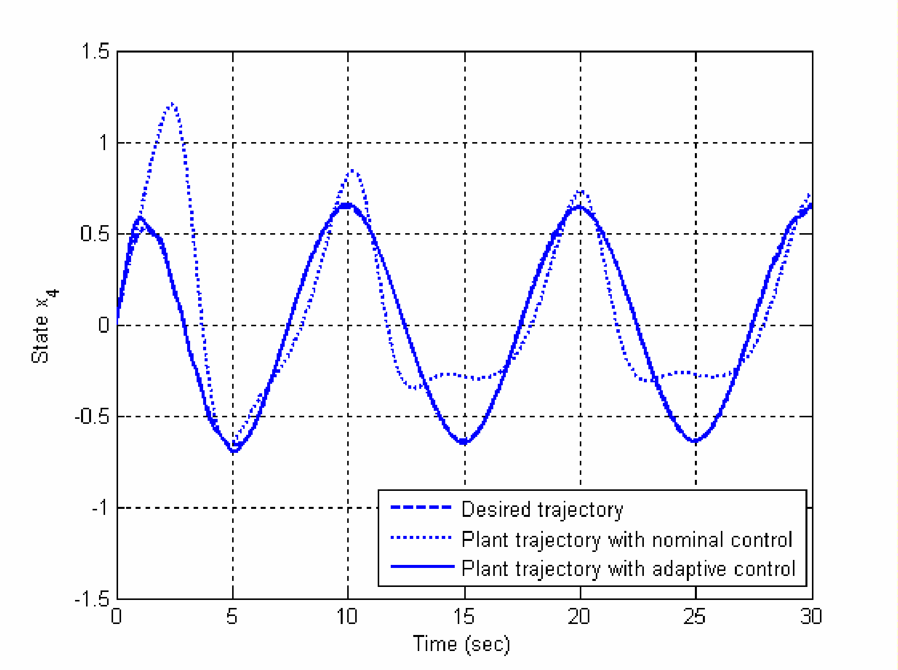


Example – 2: Double inverted pendulum

Position of Mass – 2

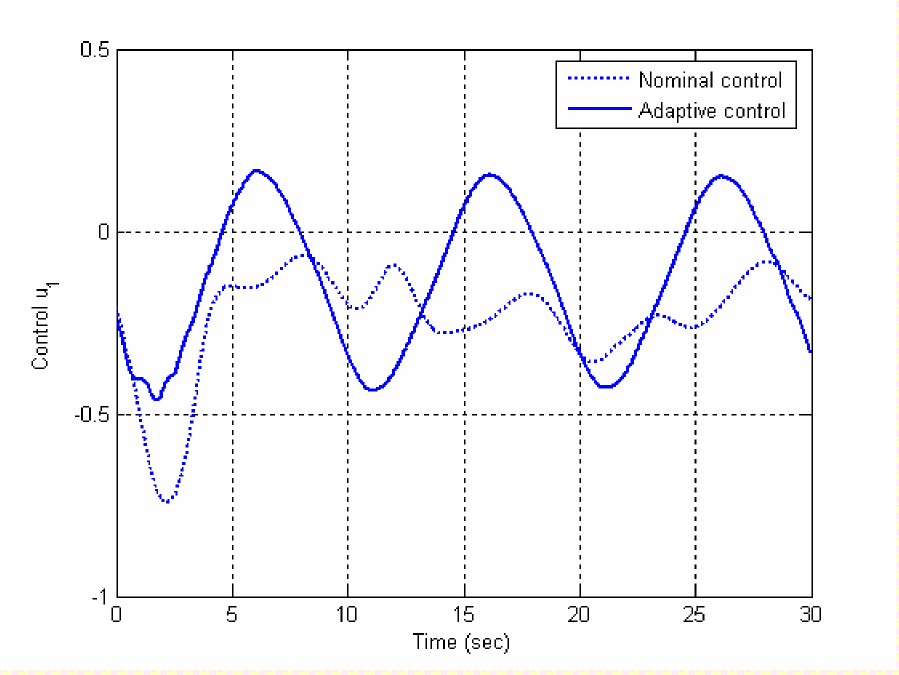


Velocity of Mass – 2

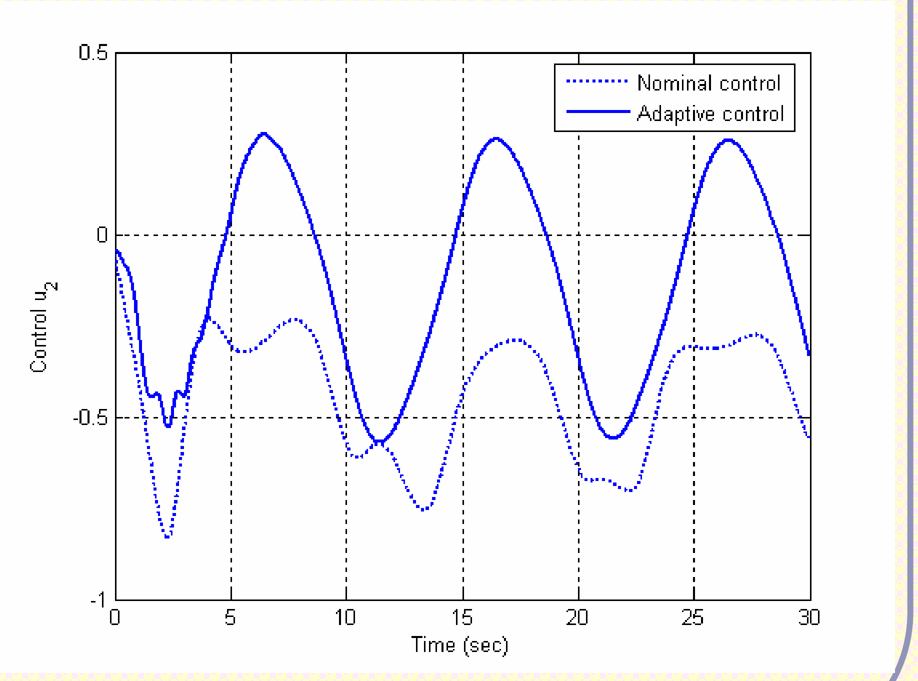


Example – 2: Double inverted pendulum

Torque for Mass – 1

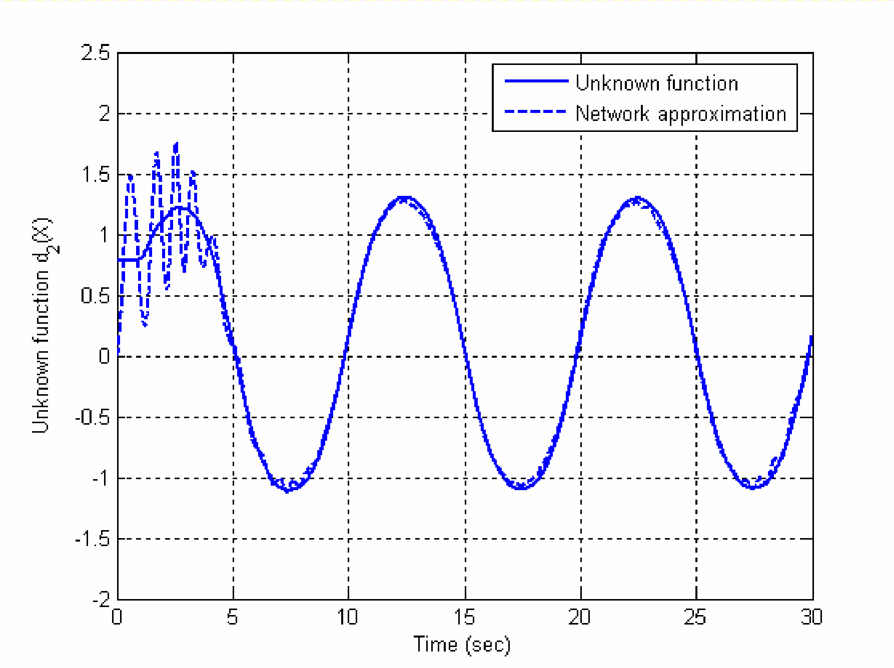


Torque for Mass – 2

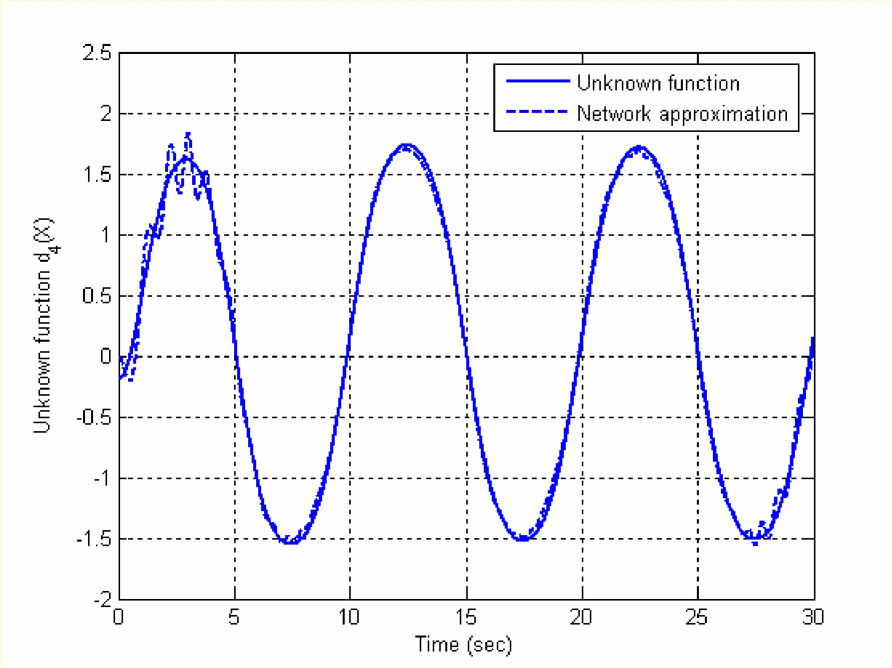


Example – 2: Double inverted pendulum

Capturing $d_2(X)$



Capturing $d_4(X)$



Neuro-Adaptive Design for Output Robustness

Dr. Radhakant Padhi
Asst. Professor
Dept. of Aerospace Engineering
Indian Institute of Science - Bangalore

N-A for Robustness of Output Dynamics

Desired output dynamics:

$$\boxed{\dot{Y}_d = f_{Y_d}(X_d) + G_{Y_d}(X_d)U_d}$$

Actual output dynamics:

$$\boxed{\dot{Y} = f_{Y_d}\left(X\right) + G_{Y_d}\left(X\right)U + d\left(X\right)}$$

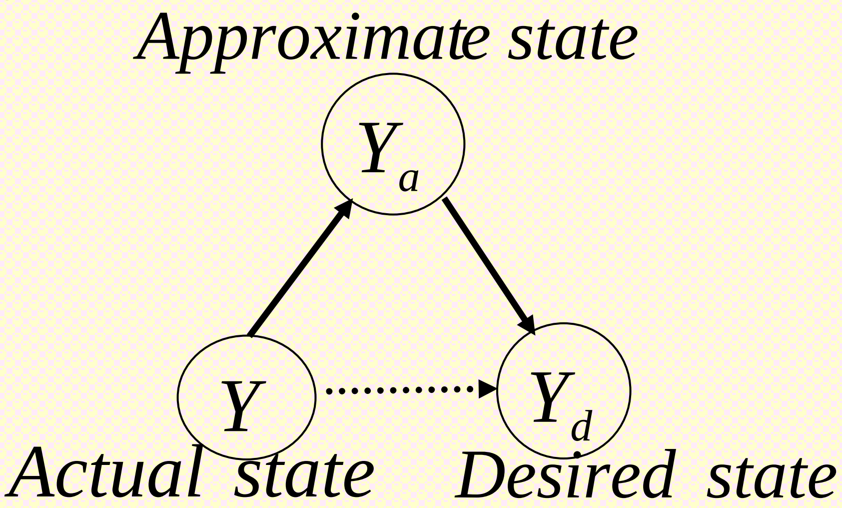
Objective: $\boxed{Y \rightarrow Y_d \text{ as soon as possible}}$

N-A for Robustness of Output Dynamics

- Dynamics of auxiliary output:

$$\boxed{\dot{Y}_a = f_{Y_d}\left(X\right) + G_{Y_d}(X)U + \underbrace{\hat{d}(X)}_{\text{NN Approx.}} + K_a(Y - Y_a)}$$

- **Strategy:**



Steps for assuring $Y_a \to Y_d$

- Enforce the error dynamics

$$\dot{E}_d + KE_d = 0 \qquad\qquad E_d \triangleq (Y_a - Y_d)$$

- After carrying out the necessary algebra

$$f(X,U) = h(X, X_a, X_d, U_d)$$

- In case of control affine system

$$f(X) + [g(X)]U = h(X, X_d, X_a, U_d)$$

- The control is given by

$$U = [g(X)]^{-1}\{h(X, X_d, X_a, U_d) - f(X)\}$$

Steps for assuring $Y \to Y_a$

- The error in the output is defined as

$$E_a \triangleq (Y - Y_a) \qquad\qquad e_{a_i} \triangleq (y_i - y_{ai})$$

- Ideal neural network is given by:

$$d_i(X) = W_i^T \varphi_i(X) + \varepsilon_i$$

where W_i is the weight matrix and $\varphi_i(X)$ is the radial basis function

Function Learning:

Define error $\boxed{e_{a_i} \triangleq (y_i - y_{a_i})}$

Output dynamics

$$\boxed{\begin{aligned} \dot{y}_i &= f_{Y_i}(X) + g_{Y_i}(X)U + d_i(X) \\ \dot{y}_{a_i} &= f_{Y_i}(X) + g_{Y_i}(X)U + \hat{d}_i(X) + k_{a_i} e_{a_i} \end{aligned}}$$

From universal function approximation property

$$d_i(X) = W_i^T \varphi_i(X) + \varepsilon_i$$

$$\hat{d}_i(X) = \hat{W}_i^T \varphi_i(X)$$

Error dynamics

$$\boxed{\begin{aligned} \dot{e}_{a_i} &= d_i(X) - \hat{d}_i(X) - k_{a_i} e_{a_i} \\ &= \tilde{W}_i^T \Phi_i(X) + \varepsilon_i - k_{a_i} e_{a_i} \end{aligned}}$$

Lyapunov Stability Analysis

Lyapunov Function Candidate:

$$\boxed{L_i = \frac{1}{2}\left(e_{a_i} p_i e_{a_i}\right) + \frac{1}{2}\left(\tilde{W}_i^{\,T} \gamma_i \tilde{W}_i\right)}$$

Derivative of Lyapunov Function:

$$\begin{aligned}
\dot{L}_i &= e_{a_i} p_i \dot{e}_{a_i} + \tilde{W}_i^T \gamma_i \dot{\tilde{W}}_i \\
&= e_{a_i} p_i \left[\tilde{W}_i^T \Phi_i(X) + \varepsilon_i - k_{a_i} e_{a_i} \right] - \tilde{W}_i^T \gamma_i \dot{\hat{W}}_i \\
&= \tilde{W}_i^T \left[\underbrace{e_{a_i} p_i \Phi_i(X) - \gamma_i^{-1} \dot{\hat{W}}_i}_{0} \right] + e_{a_i} p_i \varepsilon_i - k_{a_i} e_{a_i}^2 p_i
\end{aligned}$$

Weight Update Rule:

$$\boxed{\dot{\hat{W}}_i = \gamma_i e_{a_i} p_i \Phi_i(X, X_d)}$$

Lyapunov Stability Analysis

This condition leads to $\boxed{\dot{L}_i = e_{a_i} p_i \varepsilon_i - k_{a_i} e_{a_i}^2 p_i}$

$\boxed{\dot{L}_i < 0}$ whenever $\boxed{|e_{a_i}| > |\varepsilon_i| / k_{a_i}}$

Using the Lyapunov stability theory, we conclude that the trajectory of e_{a_i} and $\tilde{W}_i$ are pulled towards the origin.

Hence, the output dynamics is "Practically Stable"!

Neuro-Adaptive Design with Modified Weight Update Rule (with σ modification)

Dr. Radhakant Padhi
Asst. Professor
Dept. of Aerospace Engineering
Indian Institute of Science - Bangalore

Neuro-adaptive Design with σ Modification

Weight update rule:

$$\dot{\hat{W}}_i = \gamma_i e_{a_i} \Phi_i - \gamma_i \sigma_i \hat{W}_i, \qquad \hat{W}_i(0) = 0$$

where, γ_i : Learing rate, $\sigma_i > 0$: Stabilizing factor

$$e_{a_i} = x_i - x_{a_i}$$

Φ_i : Basis function

Estimation of unknown function:

$$\hat{d}(X) = \hat{W}^T \Phi_i$$

Neuro-adaptive Design with σ Modification

Lyapunov function candidate:

$$v_i = \frac{1}{2}e_{a_i}^2 + \frac{1}{2}\tilde{W}_i^T \gamma_i^{-1} \tilde{W}_i \qquad \left(\text{Note: } p_i = 1\right)$$

Then $\quad \dot{v}_i = e_{a_i}\dot{e}_{a_i} + \tilde{W}_i^T \gamma_i^{-1} \dot{\tilde{W}}_i$

$$= \left(e_{a_i}\varepsilon_i - e_{a_i}^2\right) + \sigma_i \tilde{W}_i^T \hat{W}_i \quad \text{(can be derived so, with } k_{a_i} = 1\text{)}$$

Consider the last term in $\dot{v}_i$

$$\tilde{W}_i^T \hat{W}_i = \frac{1}{2} \times 2\left(\tilde{W}_i^T \hat{W}_i\right)$$

$$= \frac{1}{2} \times 2\tilde{W}_i^T \left(W_i - \tilde{W}_i\right) = \frac{1}{2}\left(2\tilde{W}_i^T W_i - 2\tilde{W}_i^T \tilde{W}_i\right)$$

Neuro-adaptive Design with σ Modification

However, $$2\tilde{W}_i^T W_i = \tilde{W}_i^T W_i + \tilde{W}_i^T W_i$$

$$= \tilde{W}_i^T\left(\hat{W}_i + \tilde{W}_i\right) + \left(W_i - \hat{W}_i\right)^T W_i$$

$$= \tilde{W}_i^T \hat{W}_i + \tilde{W}_i^T \tilde{W}_i + W_i^T W_i - \hat{W}_i^T W_i$$

$$= \hat{W}_i^T\left(\tilde{W}_i - W_i\right) + \tilde{W}_i^T \tilde{W}_i + W_i^T W_i$$

$$= -\hat{W}_i^T \hat{W}_i + \tilde{W}_i^T \tilde{W}_i + W_i^T W_i$$

Hence, $$\sigma \tilde{W}_i^T \hat{W}_i = \sigma \frac{1}{2}\left(-\hat{W}_i^T \hat{W}_i + \tilde{W}_i^T \tilde{W}_i + W_i^T W_i - \tilde{W}_i^T \tilde{W}_i - \tilde{W}_i^T \tilde{W}_i\right)$$

$$= \frac{1}{2}\left(-\sigma \hat{W}_i^T \hat{W}_i - \sigma \tilde{W}_i^T \tilde{W}_i + \sigma W_i^T W_i\right)$$

$$\leq \frac{1}{2}\left(-\sigma \left\|\tilde{W}_i\right\|^2 - \sigma \left\|\hat{W}_i\right\|^2 + \sigma \left\|W_i\right\|^2\right)$$

Neuro-adaptive Design with σ Modification

Hence, the equation for $\dot{v}_i$ becomes

$$\dot{v}_i \le e_{a_i}\varepsilon_i - e_{a_i}^2 - \frac{1}{2}\sigma_i\left\|\tilde{W}_i\right\|^2 - \frac{1}{2}\sigma_i\left\|\hat{W}_i\right\|^2 + \frac{1}{2}\sigma_i\left\|W_i\right\|^2$$

$$\le \frac{e_{a_i}^2}{2} + \frac{\varepsilon_i^{\,2}}{2} - e_{a_i}^2 - \frac{1}{2}\sigma_i\left\|\tilde{W}_i\right\|^2 - \frac{1}{2}\sigma_i\left\|\hat{W}_i\right\|^2 + \frac{1}{2}\sigma_i\left\|W_i\right\|^2$$

$$\le -\frac{e_{a_i}^2}{2} + \left(\frac{\varepsilon_i^{\,2}}{2} + \frac{1}{2}\sigma_i\left\|W_i\right\|^2 - \frac{1}{2}\sigma_i\left\|\tilde{W}_i\right\|^2 - \frac{1}{2}\sigma_i\left\|\hat{W}_i\right\|^2\right)$$

Neuro-adaptive Design with σ Modification

Defining
$$\beta_i \triangleq \frac{\varepsilon_i^{\,2}}{2} + \frac{1}{2}\sigma_i\left(\left\|W_i\right\|^2 - \left\|\tilde{W}_i\right\|^2 - \left\|\hat{W}_i\right\|^2\right)$$

We have

$$\dot{v}_i < 0, \quad \text{whenever} \quad \frac{e_{a_i}^2}{2} > \beta_i$$

$$i.e. \qquad \dot{v}_i < 0, \quad \text{whenever} \quad \left|e_{a_i}\right| > \sqrt{2\beta_i}$$

Summary: Neuro-Adaptive Design

- N-A Design: A generic model-following adaptive design for robustifying “any” nominal controller
- It is valid for both non-square and non-affine problems in general
- Extensions:
  - Robustness of output dynamics only
  - “Structured N-A design” for efficient learning

References

- **Padhi, R., Unnikrishnan, N. and Balakrishnan, S. N., "*Model Following Neuro-Adaptive Control Design for Non-square, Non-affine Nonlinear Systems*", IET Control Theory and Applications, Vol. 1 (6), Nov 2007, pp.1650-1661.**
- **Padhi, R., Kothari, M., "*An Optimal Dynamic Inversion Based Neuro-adaptive Approach for Treatment of Chronic Myelogenous Leukemia*", Computer Methods and Programs in Biomedicine, Vol. 87, 2007, pp. 208-224.**
- **Rajasekaran, J., Chunodkar, A. and Padhi, R., "*Structured Model-following Neuro-adaptive Design for Attitude Maneuver of Rigid Bodies*", Control Engineering Practice, Vol. 17, 2009, pp.676-689.**

Thanks for the Attention...!