Lecture – 36

Neuro-Adaptive Design - I:

A Robustifying Tool for Dynamic Inversion 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 and/or change of system parameters during operation
  - Inaccuracy in computations (like matrix inversion)
- Objective: To increase the robustness of “dynamic inversion” with respect to parameter and/or modeling inaccuracies

Reference

B. S. Kim and A. J. Calise, **Nonlinear Flight Control Using Neural Networks,** *Journal of Guidance, Control and Dynamics, Vol. 20, No. 1, 1997, pp. 26-33*.

Neuro-adaptive design

System $$\ddot{X} = f\left(X, \dot{X}, \delta\right)$$

$$X, \dot{X} \in \mathbb{R}^n, \delta \in \mathbb{R}^m$$

Assumptions:

(1) $X, \dot{X}$ are available for control computation

(2) $m = n$ (square system, i.e. $X, \dot{X}, \delta \in \mathbb{R}^m$)

(3) f is invertible

Goal: $X \to X_c$ $\left(X_c\text{: commanded signal}\right)$

and $\dot{X} \to \dot{X}_c$

Neuro-adaptive design

Ideal Case:

Define $\tilde{X} \triangleq X_c - X$

Design δ such that

$$\ddot{\tilde{X}} + K_d \dot{\tilde{X}} + K_p \tilde{X} = 0 \text{ , where } K_p, K_d > 0 \quad (\text{pdf matrices})$$

If $\quad K_d = diag\left(k_{d_i}\right), \; K_p = diag\left(k_{p_i}\right), \quad k_{p_i}, k_{d_i} > 0$

Then

$$\ddot{\tilde{x}}_i + k_{d_i} \dot{\tilde{x}}_i + k_{p_i} \tilde{x}_i = 0$$

$$\left(\ddot{x}_{c_i} - \ddot{x}_i\right) + k_{d_i} \dot{\tilde{x}}_i + k_{p_i} \tilde{x}_i = 0$$

Neuro-adaptive design

$$\ddot{x}_i = \ddot{x}_{c_i} + k_{d_i}\dot{\tilde{x}}_i + k_{p_i}\tilde{x}_i$$

$$\triangleq u_i \quad (i^{th} \text{ compoent of "}\underline{\text{pseudo control}}\text{"})$$

Repeating this exercise for $i = 1, 2, ..., n$

we get $\ddot{X} = U$ (U : Pseudo control variable)

$$\therefore \quad f\left(X, \dot{X}, \delta\right) = U$$

$$\boxed{\delta = f^{-1}\left(X, \dot{X}, U\right)} \left(\begin{array}{l}\text{with the assumtion} \\ \text{that the inverse exits}\end{array}\right)$$

Neuro-adaptive design

Note: Real - time computation of this inverse may be difficult. In the case, a Neural Network can learn this relationship offline (in an approximate sense)

i.e, $\hat{\delta} = f^{-1}\left(X, \dot{X}, U \right) = NN_1\left(X, \dot{X}, U \right)$

Problem: The model and the neural network training are NOT perfect....these introduce errors.

Neuro-adaptive design

Solution : Design another neural network to cancel this error!

Design of Adaptive NN

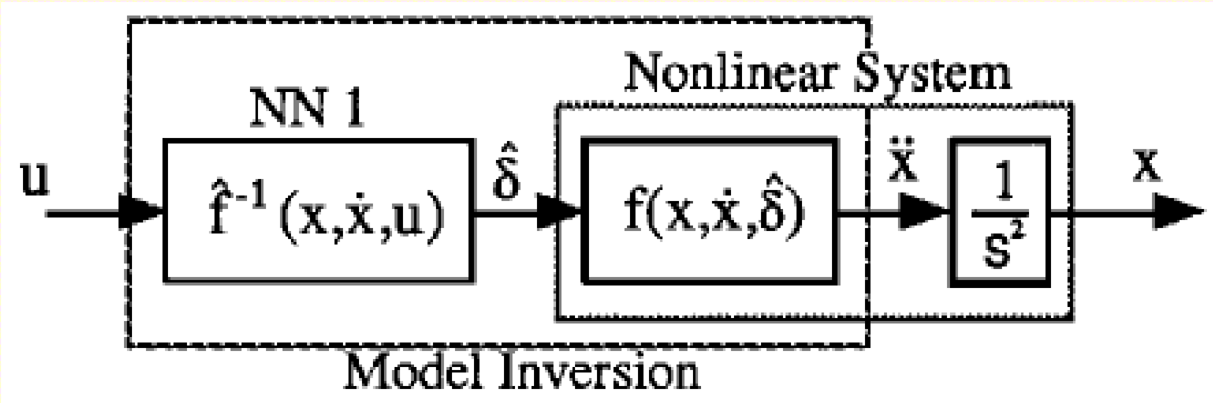
After synthesizing $\hat{\delta}$, the system dynamics is

$$\ddot{X} = f\left(X, \dot{X}, \hat{\delta}\right)$$

$$= \underbrace{f\left(X, \dot{X}, \delta\right)}_{U} + \underbrace{\left[f\left(X, \dot{X}, \hat{\delta}\right) - \underbrace{f\left(X, \dot{X}, \delta\right)}_{U} \right]}_{\Delta'\left(X, \dot{X}, U\right)}$$

Neuro-adaptive design

i.e $$\ddot{X} = U + \Delta'\left(X, \dot{X}, U\right)$$



Neuro-adaptive design

Note: If $\Delta' = 0$, then the error dynamics behaves like the ideal case, and hence, the objective will be met.

Trick : Modify the Pseudo control U as $U = U_I - \hat{U}_{ad}$

where $U_I = \ddot{X}_c + K_d \dot{\tilde{X}} + K_p \tilde{X}$

is the Pseudo control for the "ideal case".

and $\hat{U}_{ad}$: Adaptive control to cancel the unwarted effect.

Note: Pseudo-control modification is the reason why technique is applicable to dynamic inversion only.

Neuro-adaptive design

With this, the closed loop error dynamics becomes

$$\ddot{\tilde{X}} + K_d \dot{\tilde{X}} + K_p \tilde{X} = \ddot{X}_c - \ddot{X} + K_d \dot{\tilde{X}} + K_p \tilde{X}$$

$$= \left(\ddot{X}_c + K_d \dot{\tilde{X}} + K_p \tilde{X} \right) - \left[U + \Delta'\left(X, \dot{X}, U \right) \right]$$

$$= \cancel{U_I} - \left(\cancel{U_I} - \hat{U}_{ad} \right) - \Delta'\left(X, \dot{X}, U \right)$$

i.e, $\ddot{\tilde{X}} + K_d \dot{\tilde{X}} + K_p \tilde{X} = \left[\hat{U}_{ad} - \Delta'\left(X, \dot{X}, U \right) \right]$

Question: Can we design $\hat{U}_{ad}$ (adaptively) such that

$$\hat{U}_{ad} \approx \Delta'\left(X, \dot{X}, U \right)$$

Neuro-adaptive design

$$\left[\text{If this happens, then } \tilde{X}, \dot{\tilde{X}} \rightarrow 0 \text{ (approximately)}\right]$$

$\underline{i^{th} \text{ channel}}$:

$$\ddot{\tilde{x}}_i + k_{d_i} \dot{\tilde{x}}_i + k_{p_i} \tilde{x}_i = \hat{U}_{ad_i} - \Delta_i'\left(X, \dot{X}, U\right)$$

Define $e_i \triangleq \begin{bmatrix} \tilde{x}_i \\ \dot{\tilde{x}}_i \end{bmatrix}$

Then $\dot{e}_i = \underbrace{\begin{pmatrix} 0 & 1 \\ -k_{p_i} & -k_{d_i} \end{pmatrix}}_{A_i} e_i + \underbrace{\begin{bmatrix} 0 \\ 1 \end{bmatrix}}_{b} \left[\hat{U}_{ad_i} - \Delta_i'\left(X, \dot{X}, U\right)\right]$

Neuro-adaptive design

Goal for $\hat{U}_{ad_i}$

The "ideal purpose" of $\hat{U}_{ad_i}$ is to capture the function Δ_i' "pefectly" so that $e_i \to 0$ asymptotically However, this is difficult to achieve.

Hence, aim for "practical stability "only

i.e, e_i remains bounded, where the bound can be made arbitrarity small.

Neuro-adaptive design

Let $\hat{\Delta}_i'\left(X, \dot{X}, U\right)$ be a NN realization of $\Delta_i'\left(X, \dot{X}, U\right)$

when a finite number (N) of basis functions

$\beta_{ij}'\left(X, \dot{X}, U\right), \; j = 1, 2, \ldots, N$ are used, with the

corresponding weights of the network being

$\hat{W}_{ij}'(t), \; j = 1, 2, \ldots, N$

i.e, $\hat{\Delta}_i'\left(X, \dot{X}, U\right) = \sum \hat{W}_{ij}'(t)\beta_{ij}'\left(X, \dot{X}, U\right)$

Neuro-adaptive design

$$\hat{\Delta}_i'\left(X,\dot{X},U\right)=\left[\hat{W}_{i_1},\hat{W}_{i_2},\ldots,\hat{W}_{i_N}\right]\begin{bmatrix}\beta_{i_1}'\\ \beta_{i_2}'\\ \ldots\\ \beta_{i_N}'\end{bmatrix}$$

i.e. $$\boxed{\hat{U}_{ad_i}=\hat{\Delta}_i'\left(X,\dot{X},U\right)=\hat{W}_i^T(t).\,\beta_i'\left(X,\dot{X},U\right)}$$

Ideal case:

When $\hat{W}_i$ is optimized over some compact domain in $\left\{X,\dot{X},U\right\}$,

let the result be $\hat{W}_i^*$. In that case $\hat{\Delta}_i'^*\left(X,\dot{X},U\right)=\hat{W}_i^{*T}(t).\,\beta_i'\left(X,\dot{X},U\right)$

and $\left|\hat{\Delta}_i'-\hat{\Delta}_i'^*\right|<\varepsilon_i$ where ε_i is the ideal error of function.

Neuro-adaptive design

Note:

ε_i depends on the followings:

(i) N : Size of the i^{th} network

(ii) Choice of basis functions $\beta_{ij}\,(j = 1, \ldots, N)$

(iii) Accuracy of the first neural network

(which determines Δ_{i}')

Neuro-adaptive design

Estimation Error:

$$\hat{U}_{ad_i} - \hat{\Delta}_i'^T = \hat{W}_i^T(t)\ \beta_i'\left(X, \dot{X}, U\right) - \hat{W}_i^{*T}\ \beta_i'\left(X, \dot{X}, U\right)$$

$$= \left[\hat{W}_i(t) - \hat{W}_i^*\right]^T \beta_i'\left(X, \dot{X}, U\right)$$

$$= \tilde{W}_i^T \beta_i'\left(X, \dot{X}, U\right)$$

where, $\tilde{W}(t) = \hat{W}(t) - \hat{W}^*$: Error in weight at time t

Note: $\hat{W}^*$ is constant. Hence

$$\dot{\tilde{W}}(t) = \dot{\hat{W}}(t) - \underbrace{\dot{\hat{W}}^*}_{=0} = \dot{\hat{W}}(t)$$

Neuro-adaptive design

<u>Notation</u> :

$$\beta_i'\left(X,\dot{X},U\right) = \beta_i'\left(t,e,\tilde{W}\right)$$

$$\hat{\Delta}_i'^*\left(X,\dot{X},U\right) = \hat{\Delta}_i'^*\left(t,e,\tilde{W}\right)$$

$$\underbrace{\hat{\Delta}_i'\left(X,\dot{X},U\right)}_{\text{Used in implementation}} = \underbrace{\hat{\Delta}_i'\left(t,e,\tilde{W}\right)}_{\text{used in proof}}$$

<u>Note</u>: $e \triangleq \begin{bmatrix} \tilde{X} \\ \hline \dot{\tilde{X}} \end{bmatrix}, \quad e_i \triangleq \begin{bmatrix} \tilde{x}_i \\ \hline \dot{\tilde{x}}_i \end{bmatrix}$

Neuro-adaptive design

Error dynamics in i^{th} channel:

$$\dot{e}_i = A_i e_i + b\left(\hat{U}_{ad_i} - \Delta_i'\right)$$

$$= A_i e_i + b\left(\hat{U}_{ad_i} - \hat{\Delta}_i'^* + \hat{\Delta}_i'^* - \Delta_i'\right)$$

$$= A_i e_i + b\tilde{W}_i^T \beta_i'\left(t, e, \tilde{W}\right) + b\left(\hat{\Delta}_i'^* - \Delta_i'\right)$$

Note: $A_i = \begin{pmatrix} 0 & 1 \\ -k_{p_i} & -k_{d_i} \end{pmatrix}$ is a Hurwitz matrix.

Characteristic equation: $s^2 + k_{d_i} s + k_{p_i} = 0, \quad$ with $k_{d_i}, k_{p_i} > 0$

Neuro-adaptive design

Stability Analysis: Define a Lyapunov Function candidate:

$$V = \sum_{i=1}^{n} V_i\left(e_i, \tilde{W}_i\right)$$

where, $$V_i\left(e_i, \tilde{W}_i\right) = \begin{cases} \frac{1}{2} e_i^T P_i e_i + \frac{1}{2\gamma_i} \tilde{W}_i^T \tilde{W}_i, & \left\| e_i \right\|_{P_i} > E_i \\ E_i + \frac{1}{2\gamma_i} \tilde{W}_i^T \tilde{W}_i, & \left\| e_i \right\|_{P_i} \le E_i \end{cases} \qquad \left(E_i : \text{ defines "dead zone"}\right)$$

By definition, $\left\| e_i \right\|_{P_i} \triangleq \sqrt{e_i^T P_i e_i}$ $\quad$ P_i is a 2×2 pdf matrix satisfying

$$P_i A_i + A_i^T P_i = -Q_i \ , \quad Q_i > 0 \text{ (pdf)}$$

Note : By the selection, V_i is continuous at the boundary of dead zone.

Neuro-adaptive design

Note: (1) Existence of such a pdf matrix P_i is guaranteed by the fact A_i is Hurwitz.

(2) If $Q_i = I$, then the solution for is given by

$$P_i = \begin{bmatrix} \dfrac{k_{d_i}}{2k_{p_i}} + \dfrac{k_{p_i}}{2k_{d_i}}\left(1 + \dfrac{1}{k_{p_i}}\right) & \dfrac{1}{2k_{p_i}} \\ \dfrac{1}{2k_{p_i}} & \dfrac{1}{2k_{d_i}}\left(1 + \dfrac{1}{k_{p_i}}\right) \end{bmatrix}$$

Neuro-adaptive design

$$\dot{V}_i = \frac{1}{2}\left(\dot{e}_i^T P_i e_i + e_i^T P_i \dot{e}_i\right) + \frac{1}{\gamma_i}\tilde{W}_i^T \dot{\tilde{W}}_i$$

$$= \begin{bmatrix} \left[A_i e_i + b\tilde{W}_i^T \beta_i'\left(t, e, \tilde{W}\right) + b\left(\hat{\Delta}_i'^* - \Delta_i'\right)\right]^T P_i e_i \\ + e_i^T P_i \left[A_i e_i + b\tilde{W}_i^T \beta_i'\left(t, e, \tilde{W}\right) + b\left(\hat{\Delta}_i'^* - \Delta_i'\right)\right] \end{bmatrix} + \frac{1}{\gamma_i}\tilde{W}_i^T \dot{\hat{W}}_i$$

$$= \frac{1}{2} e_i^T \left(A_i^T P_i + P_i A_i\right) e_i + e_i^T P_i b \left[\tilde{W}_i^T \beta_i' + \left(\hat{\Delta}_i'^* - \Delta_i'\right)\right] + \frac{1}{\gamma_i}\tilde{W}_i^T \dot{\hat{W}}_i$$

$$\leq -\frac{1}{2} e_i^T Q_i e_i + \left\| e_i^T P_i b \right\|_2 \varepsilon_i + \tilde{W}_i^T \left[e_i^T P_i b \beta_i' + \frac{\dot{\hat{W}}_i}{\gamma_i} \right]$$

$$\left[\text{Weight update rule: } \boxed{\dot{\hat{W}}_i = -\gamma_i e_i^T P_i b \beta_i'}\right]$$

Neuro-adaptive design

$$\leq -\frac{1}{2} e_i^T Q_i e_i + \varepsilon_i \left\| e_i^T \sqrt{P_i} \sqrt{P_i} b \right\|_2 \quad \left(\sqrt{P_i} \text{ is deined since } P_i \text{ is pdf} \right)$$

$$\leq -\frac{1}{2} \left\| e_i \right\|_2^2 \lambda_{\min}\left(Q_i\right) + \varepsilon_i \left\| e_i^T \sqrt{P_i} \right\|_2 \left\| \sqrt{P_i} \right\|_2 \left\| b \right\|_2$$

However, $e_i^T P_i e_i \leq \lambda_{\max}\left(P_i\right) \left\| e_i \right\|_2^2$

i.e, $\left\| e_i \right\|_2^2 \geq \dfrac{e_i^T P_i e_i}{\lambda_{\max}\left(P_i\right)}$

$$-\left\| e_i \right\|_2^2 \leq -\frac{e_i^T P_i e_i}{\lambda_{\max}\left(P_i\right)} = -\frac{\left\| e_i \right\|_{P_i}^2}{\lambda_{\max}\left(P_i\right)}$$

Neuro-adaptive design

$$\dot{V}_i \leq -\frac{1}{2}\frac{\|e_i\|_{P_i}^2}{\lambda_{\max}(P_i)}\lambda_{\min}(Q_i)+\varepsilon_i\sqrt{\left(e_i^T\sqrt{P_i}\right)\left(\sqrt{P_i}e_i\right)}\sqrt{\lambda_{\max}(P_i)}$$

$$= \left[-\frac{1}{2}\frac{\|e_i\|_{P_i}^2\,\lambda_{\min}(Q_i)}{\lambda_{\max}(P_i)}+\varepsilon_i\underbrace{\sqrt{e_i^T P_i e_i}}_{\|e_i\|_{P_i}}\sqrt{\lambda_{\max}(P_i)}\right]$$

$$\dot{V}_i \leq \|e_i\|_{P_i}^2\left[-\frac{1}{2}\frac{\|e_i\|_{P_i}\,\lambda_{\min}(Q_i)}{\lambda_{\max}(P_i)}+\varepsilon_i\sqrt{\lambda_{\max}(P_i)}\right]$$

$$\therefore \dot{V}_i \leq 0, \quad \text{when} \quad -\frac{1}{2}\frac{\|e_i\|_{P_i}\,\lambda_{\min}(Q_i)}{\lambda_{\max}(P_i)}+\varepsilon_i\sqrt{\lambda_{\max}(P_i)}<0$$

Neuro-adaptive design

$$\text{i.e,} \quad \varepsilon_i \sqrt{\lambda_{\max}(P_i)} < \frac{1}{2} \frac{\|e_i\|_{P_i} \lambda_{\min}(Q_i)}{\lambda_{\max}(P_i)}$$

$$\text{i.e,} \quad \boxed{\|e_i\|_{P_i} > \frac{2\varepsilon_i \left[\lambda_{\max}(P_i)\right]^{\frac{3}{2}}}{\lambda_{\min}(Q_i)}}$$

Note: (1) This defines E_i, However it contains ε_i, which is unkown. However, it may require iterative simulation study.

Neuro-adaptive design

(2) If $Q_i = I$

$$E_i = 2\varepsilon_i \left[\lambda_{\max}\left(P_i\right)\right]^{\frac{3}{2}}$$

where, P_i is given before.

(3) Selecting $Q_i = I$ also leads to the least bounds.

<u>Inside the dead zone:</u> $\dot{V}_i = \tilde{W}_i^T \dot{\hat{W}}_i$

Select $\boxed{\dot{\hat{W}}_i = 0}$

Then $\dot{V}_i = 0$

Note: By the selection, V_i is continuous at the boundary of this deadzone.

Neuro-adaptive Design: Implementation of Controller

(1) Design Parameter selection:

$$\hat{W}_i(0) = 0$$

$$Q_i = I$$

$$P_i = \text{[formula given]}$$

$$E_i = 2\varepsilon_i \left[\lambda_{\max}\left(P_i\right)\right]^{3/2}$$

(ε_i is unknown $\Rightarrow$ Try with iteration)

Neuro-adaptive Design: Implementation of Controller

(2) Weight update Rule:

$$\dot{\hat{W}}_i = -\gamma_i e_i^T P_i b \beta_i' \ , \ \text{if} \ \|e_i\|_{P_i} > E_i$$

$$= \ 0 \qquad\qquad \text{otherwise}$$

where, $e_i = \begin{bmatrix} \tilde{x}_i \\ \dot{\tilde{x}}_i \end{bmatrix}, \ b = \begin{bmatrix} 0 \\ 1 \end{bmatrix},$

$\beta_i\left(X, \dot{X}, U\right)$: Basis function selection

$\left(U\text{: pseudo control}\right)$

γ_i : Learing rate

Neuro-adaptive Design: Implementation of Controller

(3) Control Computation:

Adaptive Control: $\hat{U}_{ad_i} = \hat{W}_i^T \beta_i'$

Pseudo Control: $U = U_I - \hat{U}_{ad}$

$$= \ddot{X}_c + K_d \dot{\tilde{X}} + K_p \tilde{X} - \hat{U}_{ad}$$

Actual Control: $\hat{\delta} = \hat{f}^{-1}\left(X, \dot{X}, U\right)$

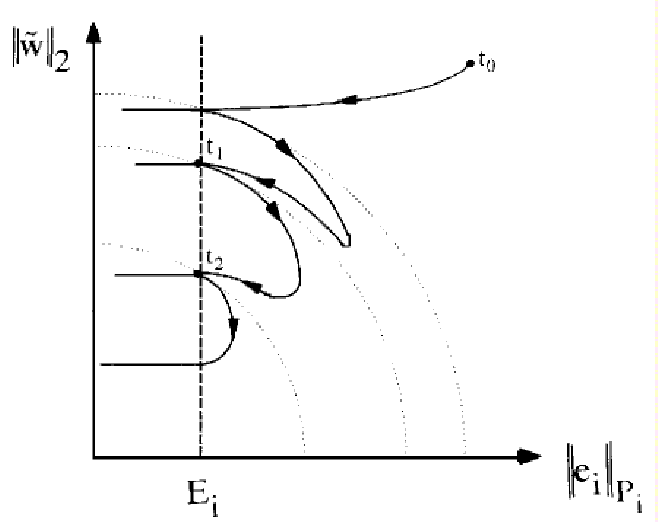
$$= \mathrm{NN}_1\left(X, \dot{X}, U\right)$$

Neuro-adaptive design

Nice Results:

(i) Adaption happens in finite time

(ii) As $t \to \infty$, the error $e_i(t)$ lies inside the deadzone and $\tilde{W}_i(\mathrm{t})$ approaches a constant value.



Promising Results: (Flight control Problem)

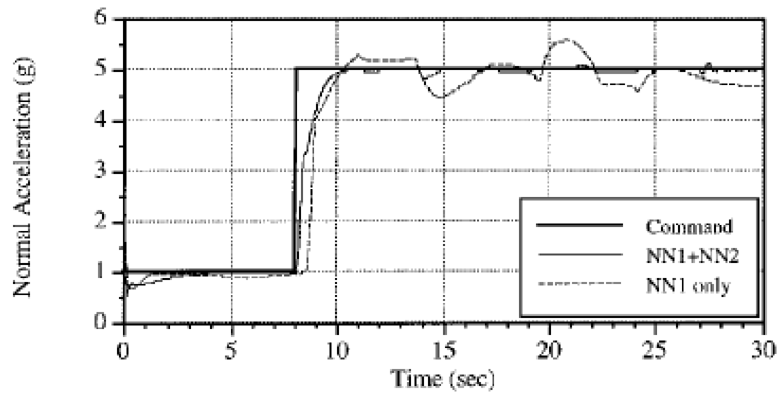


Fig. 10 **Performance results of the NN2-based controller for the flight inside the off-line training region, $M_0 = 0.6$, $h_0 = 10{,}000$ ft.**

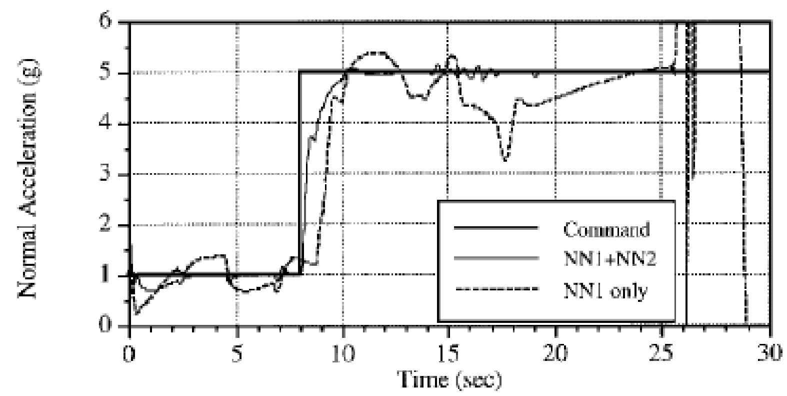


Fig. 11 **Performance results of the NN2-based controller for the flight outside the off-line training region, $M_0 = 0.8$, $h_0 = 10{,}000$ ft.**

References

- B. S. Kim and A. J. Calise, **Nonlinear Flight Control Using Neural Networks,** *Journal of Guidance, Control and Dynamics, Vol. 20, No. 1, 1997, pp. 26-33*.
- J. Leitner, A. J. Calise and J. V. R. Prasad, **Analysis of Adaptive Neural Networks for Helicopter Flight Controls**, *Journal of Guidance, Control, and Dynamics, Vol. 20, No. 5, 1997, pp. 972-979.*
- M. Sharma and A. J. Calise, **Neural-Network Augmentation of Existing Linear Controllers**, *J. of Guidance, Control and Dynamics, Vol. 28, No. 1, 2005, pp. 12-19.*

Thanks for the Attention….!