JOURNAL OF GUIDANCE, CONTROL, AND DYNAMICS
Vol. 23, No. 3, May–June 2000

Neural Networks and Adaptive Nonlinear Control of Agile Antiair Missiles

Michael B. McFarland*
U.S. Air Force Research Laboratory, Eglin Air Force Base, Florida 32542
and
Anthony J. Calise†
Georgia Institute of Technology, Atlanta, Georgia 30332

Research has shown that neural networks can be used to improve on approximate dynamic inversion for control of uncertain nonlinear systems. In one architecture, the neural network adaptively cancels inversion errors through on-line learning. Such learning is accomplished by a simple weight update rule derived from Lyapunov theory, thus assuring the stability of the closed-loop system. This methodology is reviewed and extended to incorporate an important class of neural networks with one sigmoidal hidden layer. An agile antiair-missile autopilot is subsequently designed using this control scheme. A control law based on approximate inversion of the nonlinear dynamics is presented. This control system is augmented by the addition of a neural network with on-line learning. Numerical results from a nonlinear agile antiair-missile simulation demonstrate the effectiveness of the resulting autopilot.

Introduction

ADVANCES in fighter aircraft technology continue to create challenges for designers of antiair weapon systems. The introduction of low-observable aircraft has increased the need for small, lightweight missiles. This, in turn, has led to control problems associated with airbreathing propulsion, asymmetric airframes, and reduced aerodynamic control surface area. Similarly, the advent of supermaneuverable aircraft has motivated efforts to expand the flight envelope of a missile to include high-angle-of-attack conditions. Accordingly, some proposed next-generation missiles employ propulsive control mechanisms such as thrust vectoring and reaction jets. These agile missiles achieve increased range by executing propulsive heading changes during the boost phase of flight. Such weapons can be deployed during high-angle-of-attack maneuvers and may engage targets in the rear hemisphere relative to the launch aircraft.

The dynamics of an agile missile flying in bank-to-turn mode at a high angle of attack are inherently nonlinear and may vary rapidly with time. Further, these dynamics are uncertain because aerodynamic data for vehicles operating under such conditions is difficult to obtain and may be a poor approximation to actual flight conditions. These and other concerns have prompted researchers to look beyond classical methods, which have historically dominated missile autopilot design. Robust, nonlinear, and neural-network-based control algorithms are particularly well suited for use in agile missile flight control.

Most nonlinear control techniques are based on linearizing the equations of motion by the application of nonlinear feedback. Known as feedback linearization, or dynamic inversion, these methods rely heavily on precise a priori knowledge of the plant dynamics. An early application of this theory to the missile autopilot design problem is found in Ref. 1, whereas Ref. 2 presents a more sophisticated approach involving variable structure control. More recently, neural networks have emerged as a means of explicitly accounting for uncertainties in plant dynamics. Their on-line learning and functional approximation capabilities make neural networks an excellent candidate for this application. Reference 3 is one example of a missile autopilot using neural networks.

This paper concerns the neural-network-based approach to direct adaptive control of nonlinear systems that was featured in Ref. 4. In this approach, first proposed in Ref. 5 and further developed in Ref. 6, a simple dynamic inversion controller approximately linearizes the vehicle dynamics. This controller is augmented by a neural network that acts to improve the linearization by adaptively canceling inversion errors in real time. The neural-network implementation features a stable on-line learning algorithm derived from Lyapunov theory. Because Refs. 5 and 6 dealt specifically with fighter aircraft and helicopter applications, respectively, the work described in this paper extends these efforts while adapting them specifically to the agile missile control problem. The result is an autopilot that combines the best features of dynamic inversion, neural networks, and adaptive control to increase the effectiveness and versatility of tomorrow's missile systems.

The preliminary study presented in Ref. 7 revealed that neural networks are capable of attaining sufficiently high learning rates to make adaptation feasible even during the most demanding aerial engagements. In fact, previous work documented in Ref. 8 shows that the methodology considered here compares favorably to traditional gain-scheduled linear methods for this application. This paper extends the results of earlier research, which admitted only linear-in-parameters (LIP) neural networks with no hidden layers, to accommodate the important class of single-hidden-layer (SHL) networks. These architectures are of particular interest because of their proven functional approximation capabilities. This theoretical extension is based on the work described in Ref. 9, which was originally specialized to the control of robotic manipulators. It is expanded herein to address a more general class of nonlinear systems, including systems that depend nonlinearly on the input.

The paper begins with a description of the proposed controller architecture, including some of the principal aspects of its Lyapunov-based proof of stability. For the complete proof of stability, the reader may consult Ref. 10. The methodology in question is then applied to an agile antiair-missile autopilot design problem. First, the baseline control scheme consisting of an approximate inversion of the missile's six-degree-of-freedom nonlinear dynamics is presented. An SHL neural network is then designed to offset residual nonlinearities, enhancing the performance of this nonlinear controller. Numerical simulation results for an agile antiair missile demonstrate the feasibility of this autopilot design technique. Finally, conclusions and future research directions are discussed.

Control Design Methodology

Consider a block-triangular nonlinear system with the following structure:

$$\dot{\boldsymbol{x}}_1 = \boldsymbol{f}_1(\boldsymbol{x}_1) + \boldsymbol{g}_1(\boldsymbol{x}_1)\boldsymbol{x}_2 \qquad \dot{\boldsymbol{x}}_2 = \boldsymbol{f}_2(\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{u}) + \mathbf{d}(t) \tag{1}$$

Received 24 April 1999; revision received 26 October 1999; accepted for publication 2 November 1999. This material is declared a work of the U.S. Government and is not subject to copyright protection in the United States.

*Aerospace Engineer, Munitions Directorate, 101 West Eglin Boulevard. Senior Member AIAA.

†Professor, School of Aerospace Engineering, 270 Ferst Drive. Fellow AIAA.

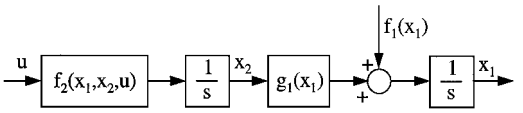

Fig. 1 Open-loop nonlinear system.

where $\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{u}, \boldsymbol{d} \in \Re^n$ and $\boldsymbol{g}_1(\boldsymbol{x}_1)$ remains nonsingular for all $\boldsymbol{x}_1$. We assume that the external disturbance $\boldsymbol{d}$ is bounded by $|\boldsymbol{d}| \leq \boldsymbol{d}^*$. A block diagram of this system is included in Fig. 1. Given a reference trajectory defined by $\boldsymbol{x}_{1c}(t)$ and its derivatives $\dot{\boldsymbol{x}}_{1c}(t)$ and $\ddot{\boldsymbol{x}}_{1c}(t)$, the objective is to achieve stable command following via full-state feedback.

Approximate Nonlinear Control

Suppose that the dynamics of the $\boldsymbol{x}_1$ subsystem are well known, but the nonlinearity $\boldsymbol{f}_2$ is only known approximately. With the state error defined by $\tilde{\boldsymbol{x}}_1 = \boldsymbol{x}_{1c}(t) - \boldsymbol{x}_1$, suppose furthermore that an asymptotically stabilizing controller $\boldsymbol{\alpha}_1(\tilde{\boldsymbol{x}}_1, t)$ for the $\tilde{\boldsymbol{x}}_1$ error dynamics with $\boldsymbol{x}_2$ as input is also known with an associated Lyapunov function L_1. One simple choice for designing $\boldsymbol{\alpha}_1$ is to use dynamic inversion. Note, however, that $\boldsymbol{\alpha}_1$ need not be an inverting controller. If, for example, the plant dynamics include so-called beneficial nonlinearities, these terms need not be canceled by the nonlinear feedback. This is described as lean nonlinear control in Ref. 11 with regard to integrator backstepping. Next, form an augmented control law as follows:

$$\bar{\boldsymbol{\alpha}}_1(\tilde{\boldsymbol{x}}_1, t) = \boldsymbol{\alpha}_1 + \boldsymbol{g}_1^T \left(\frac{\partial L_1}{\partial \tilde{\boldsymbol{x}}_1} \right)^T \tag{2}$$

This control ensures that the $\tilde{\boldsymbol{x}}_1$ error dynamics are input-to-state stable (ISS) with respect to $\tilde{\boldsymbol{x}}_2$, an error variable defined as follows:

$$\tilde{\boldsymbol{x}}_2 = \bar{\boldsymbol{\alpha}}_1 - \boldsymbol{x}_2 \tag{3}$$

The ISS property allows us to address the problem of stabilizing the $\tilde{\boldsymbol{x}}_2$ dynamics independently of $\tilde{\boldsymbol{x}}_1$. Note that when $\boldsymbol{d}(t) \equiv \boldsymbol{0}$, these dynamics take the form

$$\dot{\tilde{\boldsymbol{x}}}_2 = \dot{\bar{\boldsymbol{\alpha}}}_1 - \boldsymbol{f}_2(\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{u}) \tag{4}$$

By redefining the input, $\boldsymbol{x}_2$ the dynamics may be rewritten (in the absence of external disturbances) as follows:

$$\dot{\boldsymbol{x}}_2 = \boldsymbol{v}$$

$$\boldsymbol{v} = \boldsymbol{f}_2(\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{u}) \tag{5}$$

where $\boldsymbol{v}(t) \in R^n$ is a pseudocontrol input. If the mapping $\boldsymbol{f}_2(\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{u})$ is invertible and full-state feedback is available, the definition of $\boldsymbol{v}$ provides a linearizing transformation of the control. The corresponding inverse transformation is written as

$$\boldsymbol{u} = \boldsymbol{f}_2^{-1}(\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{v}) \tag{6}$$

and must be computed in real time when the control is implemented. If the mapping $\boldsymbol{f}_2$ is known and its inverse is computed accurately, then the system is exactly linearized.

In the case of perfect inversion, linear control theory may be used to construct a stabilizing pseudocontrol input for the $\tilde{\boldsymbol{x}}_2$ dynamics. For this purpose, we propose a pseudocontrol of the following form:

$$\boldsymbol{v} = K_2 \tilde{\boldsymbol{x}}_2 + \dot{\bar{\boldsymbol{\alpha}}}_1 \tag{7}$$

The last term in Eq. (7) has the effect of the command derivative term in model-following control and is necessary to achieve tracking of arbitrary smooth trajectories. For slowly varying commands, this term may be neglected.

Because $\boldsymbol{f}_2$ is uncertain and external disturbances are present, exact linearization is impossible. Equation (6) is approximated as $\hat{\boldsymbol{u}} = \hat{\boldsymbol{f}}_2^{-1}(\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{v})$, resulting in

$$\dot{\boldsymbol{x}}_2 = \boldsymbol{v} + \Delta'(\boldsymbol{x}_1, \boldsymbol{x}_2, \hat{\boldsymbol{u}}) + \boldsymbol{d} \tag{8}$$

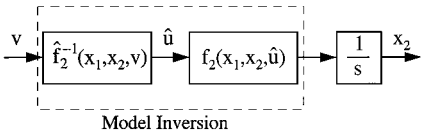

Fig. 2 Approximate dynamic inversion.

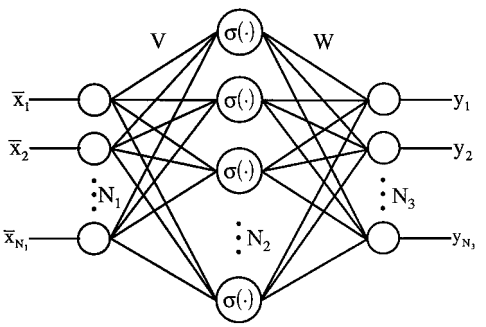

Fig. 3 Neural network with one hidden layer.

where

$$\Delta'(\boldsymbol{x}_1, \boldsymbol{x}_2, \hat{\boldsymbol{u}}) = \boldsymbol{f}_2(\boldsymbol{x}_1, \boldsymbol{x}_2, \hat{\boldsymbol{u}}) - \hat{\boldsymbol{f}}_2(\boldsymbol{x}_1, \boldsymbol{x}_2, \hat{\boldsymbol{u}}) = \Delta(\tilde{\boldsymbol{x}}_1, \tilde{\boldsymbol{x}}_2, \bar{\boldsymbol{\alpha}}_1, \boldsymbol{v}) \tag{9}$$

In Eq. (9), $\Delta': R^n \times R^n \times R^n \rightarrow R^n$ is a nonlinear mapping representing the inversion error. The second set of dependencies in Eq. (9) denotes an equivalent representation of this inversion error, which will become useful in the subsequent adaptive control development. Figure 2 illustrates the approximate dynamic inversion.

Neural-Network-Based Adaptation

In the case of nonzero inversion error Δ, the control law in Eq. (7) must be augmented by an adaptive term, to be denoted by $\boldsymbol{v}_{\text{ad}}$. This adaptive control is designed to counteract dynamic inversion errors by taking advantage of the documented functional approximation capabilities of SHL neural networks such as the one shown in Fig. 3. Following the notation used in Ref. 9, the output of this network, with input $\bar{\boldsymbol{x}} \in \Re^{N_1}$, takes the form

$$y_i = \sum_{j=1}^{N_2} \left[w_{ij} \sigma \left(\sum_{k=1}^{N_1} v_{jk} \bar{x}_k + \theta_{v_j} \right) + \theta_{w_i} \right], \qquad i = 1, 2, \ldots, N_3 \tag{10}$$

where σ is the hidden-layer activation function, v_{jk} are the interconnection weights between the input and hidden layers, and w_{ij} are the interconnection weights between the hidden and output layers. The bias terms θ_{wi} and θ_{vj} represent thresholds. This architecture has N_1 inputs, N_2 hidden-layer neurons, and N_3 outputs. The form of the hidden-layer activation function is a design parameter, but we will consider the case of sigmoidal activation functions:

$$\sigma(z) = 1/(1 + e^{-az}) \tag{11}$$

We may express Eq. (10) in matrix form as

$$\boldsymbol{y} = W \sigma(V \bar{\boldsymbol{x}}) \tag{12}$$

where the thresholds are incorporated into the weight matrices as follows:

$$W = \left[\begin{array}{c|ccc} \theta_{w_1} & w_{11} & \cdots & w_{1N_2} \\ \vdots & \vdots & \ddots & \vdots \\ \theta_{w_{N3}} & w_{N_3 1} & \cdots & w_{N_3 N_2} \end{array} \right]$$

$$V = \left[\begin{array}{c|ccc} \theta_{V_1} & w_{11} & \cdots & w_{1N_1} \\ \vdots & \vdots & \ddots & \vdots \\ \theta_{V_{N3}} & w_{N_3 1} & \cdots & w_{N_3 N_1} \end{array} \right] \tag{13}$$

Here, we have defined

$$\boldsymbol{y}^T = \begin{bmatrix} y_1 & y_2 & \cdots & y_{N_3} \end{bmatrix} \tag{14}$$

$$\bar{\boldsymbol{x}}^T = \begin{bmatrix} 1 & \bar{x}_1 & \bar{x}_2 & \cdots & \bar{x}_{N_1} \end{bmatrix} \tag{15}$$

$$\boldsymbol{\sigma}^T(z) = \begin{bmatrix} 1 & \sigma(z_1) & \sigma(z_2) & \cdots \end{bmatrix} \tag{16}$$

A neural network of the type shown in Fig. 3 is capable of approximating any smooth function to any desired accuracy, provided the number of hidden-layer neurons N_2 is sufficiently large. This implies that for continuous Δ and any $\varepsilon_N > 0$ there exists a finite N_2 and weight matrices W and V such that

$$\Delta = W\sigma(V\bar{\boldsymbol{x}}) + \varepsilon(\bar{\boldsymbol{x}}) \tag{17}$$

with $\|\varepsilon\| \leq \varepsilon_N$. Therefore, Eq. (7) may be modified to account for inversion errors by introducing as an additional term a neural-network approximation of Δ. In this paper, an adaptive scheme will be employed to tune the weight matrices on-line so that the network requires no off-line learning phase. The pseudocontrol $\boldsymbol{\nu}$ in Eq. (7) may be represented in terms of its various components as follows:

$$\boldsymbol{\nu} = \boldsymbol{\nu}_0 - \boldsymbol{\nu}_{\text{ad}} + \bar{\boldsymbol{\nu}}, \qquad \boldsymbol{\nu}_0 = K_2\tilde{\boldsymbol{x}}_2 + \dot{\tilde{\boldsymbol{\alpha}}}_1, \qquad \boldsymbol{\nu}_{\text{ad}} = \hat{W}\sigma(\hat{V}\bar{\boldsymbol{x}}) \tag{18}$$

where $\bar{\boldsymbol{\nu}}$ is an additional pseudocontrol term, to be defined. In the expression for $\boldsymbol{\nu}_{\text{ad}}$, the terms $\hat{W}$ and $\hat{V}$ are adaptive estimates of weight matrices W and V that have, in some sense, ideal constant values.

Lyapunov stability theory will be applied to determine stable learning rules for the parameter estimates $\hat{W}$ and $\hat{V}$ as well as $\bar{\boldsymbol{\nu}}$. We assume that the ideal weight matrices are bounded in the sense that $\|W\|_F \leq \bar{W}$ and $\|V\|_F \leq \bar{V}$, where $\|\cdot\|_F$ denotes the Frobenius norm, $\|A\|_F^2 = \text{tr}\{A^TA\}$. When not specified otherwise, $\|\cdot\|$ indicates the Euclidean norm. Introducing more compact notation, we may also write

$$Z = \begin{bmatrix} W & 0 \\ 0 & V \end{bmatrix} \qquad \text{with} \qquad \|Z\|_F \leq \bar{Z} \tag{19}$$

The reference command $\boldsymbol{x}_{1c}$ and its first two time derivatives are also assumed to be bounded.

Observing the functional dependencies in Eqs. (9) and anticipating other dependencies that will arise later, we choose the network input

$$\bar{\boldsymbol{x}}^T = \begin{bmatrix} 1 & \tilde{\boldsymbol{x}}_1^T & \tilde{\boldsymbol{x}}_2^T & \boldsymbol{x}_{1c}^T & \dot{\boldsymbol{x}}_{1c}^T & \ddot{\boldsymbol{x}}_{1c}^T & \boldsymbol{\nu}_{\text{ad}}^T & \|\hat{Z}\|_F \end{bmatrix} \tag{20}$$

which is bounded as follows:

$$\|\bar{\boldsymbol{x}}\| \leq c_1 + c_2\|\tilde{\boldsymbol{x}}\| + c_3\|\tilde{Z}\|_F \qquad c_i > 0 \tag{21}$$

where we have introduced the partitioned vector $\tilde{\boldsymbol{x}}^T = \begin{bmatrix} \tilde{\boldsymbol{x}}_1^T & \tilde{\boldsymbol{x}}_2^T \end{bmatrix}$ of state errors.

A key result that facilitates the extension of theory previously developed for simple LIP neural networks to more complicated SHL neural networks has been described in Ref. 9. It involves the use of a Taylor series expansion of the hidden-layer output. Define the error variables $\tilde{W} = \hat{W} - W$, $\tilde{V} = \hat{V} - V$, and $\tilde{Z} = \hat{Z} - Z$. Also, define the hidden-layer output error as follows:

$$\tilde{\boldsymbol{\sigma}} = \hat{\boldsymbol{\sigma}} - \boldsymbol{\sigma} = \sigma(\hat{V}\bar{\boldsymbol{x}}) - \sigma(V\bar{\boldsymbol{x}}) \tag{22}$$

The Taylor series expansion of σ about $\hat{\sigma}$ may then be written as

$$\sigma(V\bar{\boldsymbol{x}}) = \sigma(\hat{V}\bar{\boldsymbol{x}}) - \sigma'(\hat{V}\bar{\boldsymbol{x}})\tilde{V}\bar{\boldsymbol{x}} + \mathcal{O}(\tilde{V}\bar{\boldsymbol{x}})^2 \tag{23}$$

where $\sigma'(\hat{z}) = \text{d}\sigma(z)/\text{d}z|_{z=\hat{z}}$ and $\mathcal{O}(\cdot)^2$ denotes terms with order greater than one. These higher-order terms satisfy

$$\mathcal{O}(\tilde{V}\bar{\boldsymbol{x}})^2 = [\sigma(V\bar{\boldsymbol{x}}) - \sigma(\hat{V}\bar{\boldsymbol{x}})] + \sigma'(\hat{V}\bar{\boldsymbol{x}})\tilde{V}\bar{\boldsymbol{x}}$$

$$\Rightarrow \|\mathcal{O}(\tilde{V}^T\bar{\boldsymbol{x}})^2\| \leq c_4 + c_5\|\tilde{V}\|_F + c_6\|\tilde{V}\|_F\|\bar{\boldsymbol{x}}\| + c_7\|\tilde{V}\|_F\|\tilde{W}\|_F$$

$$c_i > 0 \tag{24}$$

where the inequality follows from the definition of the sigmoid activation function and certain properties of vector norms.

Here, it is necessary to rewrite the $\tilde{\boldsymbol{x}}_2$ dynamics as

$$\dot{\tilde{\boldsymbol{x}}}_2 = -K_2\tilde{\boldsymbol{x}}_2 + \hat{W}\hat{\boldsymbol{\sigma}} - W\boldsymbol{\sigma} - (\boldsymbol{d} + \boldsymbol{\varepsilon}) - \bar{\boldsymbol{\nu}}$$

$$= -K_2\tilde{\boldsymbol{x}}_2 + \tilde{W}\hat{\boldsymbol{\sigma}} - \tilde{W}\tilde{\boldsymbol{\sigma}} + \hat{W}\tilde{\boldsymbol{\sigma}} - (\boldsymbol{d} + \boldsymbol{\varepsilon}) - \bar{\boldsymbol{\nu}} \tag{25}$$

Recalling Eq. (22), we may also write

$$\dot{\tilde{\boldsymbol{x}}}_2 = -K_2\tilde{\boldsymbol{x}}_2 + \tilde{W}(\hat{\boldsymbol{\sigma}} - \hat{\boldsymbol{\sigma}}'\hat{V}\bar{\boldsymbol{x}}) + \hat{W}\hat{\boldsymbol{\sigma}}'\tilde{V}\bar{\boldsymbol{x}} + (\boldsymbol{w} - \bar{\boldsymbol{\nu}}) \tag{26}$$

where $\hat{\boldsymbol{\sigma}}' = \boldsymbol{\sigma}'(\hat{V}\bar{\boldsymbol{x}})$ and any extraneous terms have been collected into the disturbance signal $\boldsymbol{w}$ as follows:

$$\boldsymbol{w} = \tilde{W}^T\hat{\boldsymbol{\sigma}}'V^T\bar{\boldsymbol{x}} + W^T\mathcal{O}(\tilde{V}\bar{\boldsymbol{x}})^2 - (\boldsymbol{d} + \boldsymbol{\varepsilon})$$

$$\Rightarrow \|\boldsymbol{w}\| \leq C_0 + C_1\|\tilde{Z}\|_F + C_2\|\tilde{Z}\|_F\|\tilde{\boldsymbol{x}}\| + C_3\|\tilde{Z}\|_F^2$$

$$C_i > 0 \tag{27}$$

The norm bound on $\boldsymbol{w}$ follows from Eqs. (21) and (24) along with the boundedness of $\boldsymbol{d}$, $\boldsymbol{e}$, and $\hat{\boldsymbol{\sigma}}'$ by certain norm inequalities.

For the stability analysis, consider a typical choice of Lyapunov function candidate,

$$L_2 = \tfrac{1}{2}\tilde{\boldsymbol{x}}_2^T\tilde{\boldsymbol{x}}_2 + \tfrac{1}{2}\text{tr}\left(\tilde{W}\Gamma_W^{-1}\tilde{W}^T\right) + \tfrac{1}{2}\text{tr}\left(\tilde{V}\Gamma_W^{-1}\tilde{V}^T\right) \tag{28}$$

The Lyapunov analysis is similar to that of Ref. 9, with modifications to account for the more general structure of the open-loop nonlinear plant as described in Eq. (1). Based on this development, learning rules for the weight matrices are chosen as follows:

$$\dot{\hat{V}}^T = \dot{\tilde{V}}^T = -\Gamma_V\left[\bar{\boldsymbol{x}}\left(\tilde{\boldsymbol{x}}_2^T\hat{W}\hat{\boldsymbol{\sigma}}'\right) + \lambda\|\tilde{\boldsymbol{x}}_2\|\hat{V}^T\right]$$

$$\dot{\hat{W}}^T = \dot{\tilde{W}}^T = -\Gamma_W\left[(\hat{\boldsymbol{\sigma}} - \hat{\boldsymbol{\sigma}}'\hat{V}\bar{\boldsymbol{x}})\tilde{\boldsymbol{x}}_2^T + \lambda\|\tilde{\boldsymbol{x}}_2\|\hat{W}^T\right] \tag{29}$$

Note that the λ terms in Eq. (29) correspond to the $\boldsymbol{e}$-modification found in adaptive control literature. These terms provide additional damping, which helps to contain the growth of the parameter estimates. In addition, the pseudocontrol term $\bar{\boldsymbol{\nu}}$ in Eq. (18) is defined as

$$\bar{\boldsymbol{\nu}} = K_Z(\|\hat{Z}\|_F + \bar{Z})(\|\tilde{\boldsymbol{x}}_1\| + \|\tilde{\boldsymbol{x}}_2\|)\hat{\boldsymbol{e}}_2 \tag{30}$$

where $\hat{\boldsymbol{e}}_2 = \tilde{\boldsymbol{x}}_2/\|\tilde{\boldsymbol{x}}_2\|$ is a unit vector in the direction of $\tilde{\boldsymbol{x}}_2$. The gain K_Z is chosen such that

$$K_Z > C_2 \tag{31}$$

to effectively dominate nonlinear phenomena associated with parameter errors.

We then conclude that the trajectories of the closed-loop error system are uniformly ultimately bounded whenever, in addition to $\lambda > C_3$, one of the following two inequalities holds:

$$\|\tilde{\boldsymbol{x}}\| > \frac{(\lambda - C_3)C_4^2 + 4 + C_0}{\underline{\sigma}(K_2)} = b_r$$

$$\|\tilde{Z}\|_F > C_4/2 + \sqrt{C_4^2 + 4 + C_0/(\lambda - C_3)} = b_Z \tag{32}$$

where $C_4 = (\lambda\bar{Z} + C_1)/(\lambda - C_3)$ and $\underline{\sigma}(K_2)$ is the minimum singular value of the gain matrix K_2.

This result is a consequence of an extension of Lyapunov stability theory that requires that the Lyapunov derivative $\dot{L}_2$ remain negative outside some compact set. In fact, the quantities on the right-hand sides of the inequalities in Eq. (32) may be considered as practical upper bounds on the errors. This completes the development of the neural-network-based adaptive control system. Figure 4 shows the $\tilde{x}_2$ subsystem under this adaptive control architecture.

Note from Fig. 4 and Eq. (20) that the network output $\boldsymbol{\nu}_{\text{ad}}$ is also a network input. Thus, a critical assumption in the stability proof involves the existence of a fixed-point solution for the quantity ν_{ad}. This condition is guaranteed to be satisfied because sigmoidal activation functions are bounded in magnitude. Furthermore, when a stable fixed point exists, a simple iterative scheme may be employed

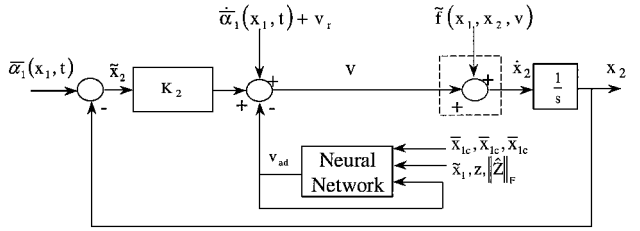

Fig. 4 Adaptive control architecture.

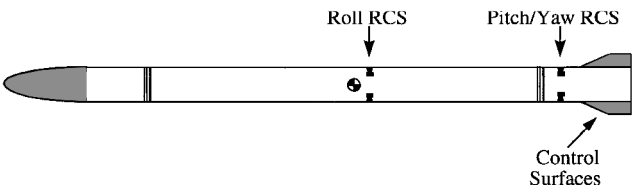

Fig. 5 Agile missile configuration.

to compute the network output. Note also that the plant dynamics in Eq. (1) do not include unmodeled dynamics. Extensions of the present methodology to allow input unmodeled dynamics, such as those associated with actuators, are presented in Refs. 12 and 13.

Missile Autopilot Application

In this section, the neural-network-basedadaptive nonlinear control methodology described earlier is used to design an autopilot for a next-generation antiair missile as shown in Fig. 5. This vehicle is described in detail in Ref. 14. In the current scenario, the objective is to design a bank-to-turn autopilot that tracks external guidance commands in angle of attack and bank angle while holding sideslip near zero.

Approximate Dynamic Inversion

The motion of a symmetric missile about its velocity vector may be described by the following equations:

$$\dot{V} = (a_x \cos\alpha + a_z \sin\alpha)\cos\beta + a_y \sin\beta$$

$$\dot{\alpha} = q - (r\sin\alpha + p\cos\alpha)\tan\beta + (-a_x \sin\alpha + a_z \cos\alpha)/V\cos\beta$$

$$\dot{\beta} = p\sin\alpha - r\cos\alpha - [(a_x \cos\alpha + a_z \sin\alpha)\sin\beta - a_y \cos\beta]/V \tag{33}$$

where V, α, and β are airspeed, angle of attack, and sideslip angle. Here, a_x, a_y, and a_z are the body-axis components of acceleration (including gravitational effects) whereas p, q, and r are the body-axis angular rates. The moment equations have the form

$$\dot{p} = L/I_{xx}, \qquad \dot{q} = M/I_{yy} + (1 - I_{xx}/I_{yy})pr$$

$$\dot{r} = N/I_{yy} - (1 - I_{xx}/I_{yy})pq \tag{34}$$

where L, M, and N are aerodynamic moments about the body axes. Finally, I_{xx} and I_{yy} are rolling and pitching moments of inertia and are assumed to have nearly constant values.

To simplify roll control in bank-to-turn flight, we now introduce an aerodynamic bank angle μ about the velocity vector. The bank angle dynamics are described as follows:

$$\dot{\mu} = [(p\cos\alpha + r\sin\alpha)/\cos\beta] + [(a_x \sin\alpha - a_z \cos\alpha)\tan\beta]/V \tag{35}$$

The following development assumes the presence of a guidance law that commands angle of attack, sideslip angle, and bank angle. There are several possible alternative choices for the roll command, including body roll angle ϕ, body roll rate p, and stability-axis roll rate ($p_s = p\cos\alpha + r\sin\alpha$).

By the introduction of more compact notation,

$$\boldsymbol{x} = \begin{Bmatrix} \alpha \\ \beta \\ \mu \end{Bmatrix} \qquad \boldsymbol{\omega} = \begin{Bmatrix} p \\ q \\ r \end{Bmatrix} \tag{36}$$

the last two of Eqs. (33) may be rewritten along with Eq. (35) as

$$\dot{\boldsymbol{x}} = \boldsymbol{a}_f + T(\boldsymbol{x})\boldsymbol{\omega} \tag{37}$$

where $T(\boldsymbol{x})$ and $\boldsymbol{a}_f$ are given by

$$T(\boldsymbol{x}) = \begin{bmatrix} -\cos\alpha\tan\beta & 1 & -\sin\alpha\tan\beta \\ \sin\alpha & 0 & -\cos\alpha \\ \cos\alpha\sec\beta & 0 & \sin\alpha\sec\beta \end{bmatrix}$$

$$\boldsymbol{a}_f = \begin{Bmatrix} (-a_x \sin\alpha + a_z \cos\alpha)/V\cos\beta \\ [-(a_x \cos\alpha + a_z \sin\alpha)\sin\beta + a_y \cos\beta]/V \\ (a_x \sin\alpha - a_z \cos\alpha)\tan\beta/V \end{Bmatrix} \tag{38}$$

The autopilot design methodology presented will now be applied to the system described by Eqs. (37) and (34). First, a first-order filter is applied to the guidance commands (α_c, β_c, and μ_c). Its outputs are the filtered commands and their first time derivatives:

$$\bar{\boldsymbol{x}}_c = \begin{Bmatrix} \bar{\alpha}_c \\ \bar{\beta}_c \\ \bar{\mu}_c \end{Bmatrix}, \qquad \dot{\bar{\boldsymbol{x}}}_c = \begin{Bmatrix} \dot{\bar{\alpha}}_c \\ \dot{\bar{\beta}}_c \\ \dot{\bar{\mu}}_c \end{Bmatrix} \tag{39}$$

Next, body-axis angular rate commands that stabilize the aerodynamic angle error dynamics are computed as follows:

$$\boldsymbol{\omega}_c = T^{-1}(\boldsymbol{x})(K_x\tilde{\boldsymbol{x}} + \dot{\boldsymbol{x}}_c - \boldsymbol{a}_f) + T^T(\boldsymbol{x})\tilde{\boldsymbol{x}} \tag{40}$$

where $\tilde{\boldsymbol{x}} = \bar{\boldsymbol{x}}_c - \boldsymbol{x}$ and we have used dynamic inversion. The positive definite gain matrix K_x is typically chosen to be diagonal. An integral term may also be included in Eq. (40), but has been omitted here for simplicity. No adaptive control terms are necessary at this stage of the design because the nonlinear dynamics are sufficiently well known. Equation (40) makes use of available acceleration and velocity information, as well as estimates of angle of attack and sideslip angle that are constructed from inertial data. The accelerometer measurements are filtered and biased appropriately to account for gravitational effects. Note that the matrix $T(\boldsymbol{x})$ may be inverted because it is nonsingular for all α and β except $\beta = \pi/2$, which should not be encountered in bank-to-turn flight.

At this point in the control design procedure, we turn our attention to the body-axis angular rates. A pseudocontrol input ν is defined by

$$\dot{\boldsymbol{\omega}} = \boldsymbol{\nu}, \qquad \boldsymbol{\nu} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{\omega}, \hat{\boldsymbol{\delta}}) \tag{41}$$

where the function $\boldsymbol{f}(\cdot)$ refers to the right-hand-side of Eq. (34). The missile's body-axis angular rate dynamics must be approximately inverted to determine the required control input. For an approximate inversion, the body-axis moments are represented linearly:

$$L \approx \hat{L} = L_\beta\beta + L_p p + L_r r + L_{\delta p}\hat{\delta}_p + L_{\delta r}\hat{\delta}_r$$

$$M \approx \hat{M} = M_\alpha\alpha + M_q q + M_{\delta q}\hat{\delta}_q$$

$$N \approx \hat{N} = N_\beta\beta + N_p p + N_r r + N_{\delta p}\hat{\delta}_p + N_{\delta r}\hat{\delta}_r \tag{42}$$

The body angular rate dynamics of Eq. (34) are then rewritten as

$$\dot{\boldsymbol{\omega}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{\omega}, \hat{\boldsymbol{\delta}}) = \hat{F}(\boldsymbol{x}, \boldsymbol{\omega}) + \hat{B}\begin{Bmatrix} \hat{\delta}_p \\ \hat{\delta}_q \\ \hat{\delta}_r \end{Bmatrix} + \Delta(\boldsymbol{x}, \boldsymbol{\omega}, \hat{\boldsymbol{\delta}}) \tag{43}$$

where $\boldsymbol{f}$ is approximated linearly by introducing

$$\hat{\boldsymbol{F}} = \begin{Bmatrix} \hat{F}_p \\ \hat{F}_q \\ \hat{F}_r \end{Bmatrix}, \qquad \hat{B} = \begin{bmatrix} L_{\delta p}/I_{xx} & 0 & L_{\delta r}/I_{xx} \\ 0 & M_{\delta q}/I_{yy} & 0 \\ N_{\delta p}/I_{xx} & 0 & N_{\delta r}/I_{yy} \end{bmatrix} \tag{44}$$

with

$$\hat{F}_p = (L_\beta \beta + L_p p + L_r r)/I_{xx} \tag{45}$$

$$\hat{F}_q = (M_\alpha \alpha + M_q q)/I_{yy} + (1 - I_{xx}/I_{yy})pr \tag{46}$$

$$\hat{F}_r = (N_\beta \beta + N_p p + N_r r)/I_{yy} + (1 - I_{xx}/I_{yy})pq \tag{47}$$

Performing the approximate dynamic inversion, the control input $\hat{\boldsymbol{\delta}}$ is written as follows:

$$\hat{\boldsymbol{\delta}} = \hat{B}^{-1}(\boldsymbol{\nu} - \hat{\boldsymbol{F}}) \tag{48}$$

The next step in the control design is the computation of the pseudocontrol input $\boldsymbol{\nu}$. Here, the commands generated by Eq. (40) are filtered to obtain

$$\bar{\omega}_c = \begin{Bmatrix} \bar{p}_c \\ \bar{q}_c \\ \bar{r}_c \end{Bmatrix}, \qquad \dot{\bar{\omega}}_c = \begin{Bmatrix} \dot{\bar{p}}_c \\ \dot{\bar{q}}_c \\ \dot{\bar{r}}_c \end{Bmatrix} \tag{49}$$

which are used in Eq. (18) to obtain the following:

$$\boldsymbol{\nu} = K_\omega \tilde{\omega} + \dot{\bar{\omega}}_c - \boldsymbol{\nu}_{\text{ad}} + K_Z\left(\|Z\|_F + \bar{Z}\right)\left(\|\tilde{x}\| + \|\tilde{\omega}\|\right)\tilde{\omega}/\|\tilde{\omega}\| \tag{50}$$

where $\boldsymbol{\nu}_{\text{ad}}$ is the adaptive control component included to cancel nonlinear inversion errors. As in the aerodynamic angle loops, K_ω is a positive definite gain matrix, and the error variable $\tilde{\omega} = \bar{\omega}_c - \omega$ has been introduced. The matrix K_Z must be chosen sufficiently large as described earlier in Eq. (31).

In practice, the stability and control derivatives in Eq. (42), along with trim values of the control inputs, could be scheduled as a function of flight condition. This, however, presumes that an accurate full-envelope model of the vehicle is available. In the current implementation, these derivatives are computed based on the assumption of constant values of the nondimensional moment coefficients, shifting the burden of gain scheduling to the neural network. The dynamic inversion control law may be simplified further by introducing additional approximations as simulation results warrant. The neural network would then be required to additionally compensate for any effects that are neglected by the dynamic inversion. Finally, the control inputs computed by Eq. (48) are distributed among the missile's aerodynamic control surfaces and Reaction Control System thrusters using a suitable control allocation algorithm.

Neural-Network Architecture

By the substituting of Eq. (40) into Eq. (37) and Eq. (48) into Eq. (43), the closed-loop error dynamics can be written as follows:

$$\dot{\tilde{x}} = -K_x \tilde{x}, \qquad \dot{\tilde{\omega}} = -K_\omega \tilde{\omega} + \boldsymbol{\nu}_{\text{ad}} - \boldsymbol{\Delta}(\boldsymbol{x}, \boldsymbol{\omega}, \boldsymbol{\nu}) - \bar{\nu} \tag{51}$$

A sufficiently large neural network of the type shown in Fig. 3 is capable of approximately reconstructing the nonlinear inversion error $\boldsymbol{\Delta}$ and may, therefore, be used to compute the adaptive contribution to the pseudocontrol. The adaptive pseudocontrol input in each channel is, thus, computed by Eq. (18) presented earlier. A stable learning rule is again given by Eq. (29).

Various neural-network topologies and choices of inputs were considered in this study. The results presented correspond to a neural network with only five sigmoidal hidden-layer neurons. Each neuron has an internal activation potential of $a = 0.01$. Network inputs were chosen to include $\tilde{\boldsymbol{x}}$, $\tilde{\omega}$, $\boldsymbol{x}_c$, $\dot{\boldsymbol{x}}_c$, $\boldsymbol{\nu}_{\text{ad}}$, $\|Z\|_F$, and Mach number. Although it does not appear in Eq. (20), Mach number is included here because its variation affects the nonlinear inversion error. This does not alter the proof of stability because for our purposes Mach number may be considered a bounded, slowly varying parameter. The network inputs $\ddot{\boldsymbol{x}}_c$ have been omitted to reduce the size of the network and the complexity of the overall design. Also, note that because $\beta_c(t) \equiv \dot{\beta}_c(t) \equiv \ddot{\beta}_c(t) \equiv 0$, these particular inputs may be omitted without loss of generality. The learning rates Γ_W and Γ_V were chosen as diagonal matrices with nonzero elements equal to 400. Other parameter values used in the neural-network implementation were $\lambda = 0.01$ in Eq. (29) and $K_Z = 0.5$ and $\bar{Z} = 50$ in Eq. (30).

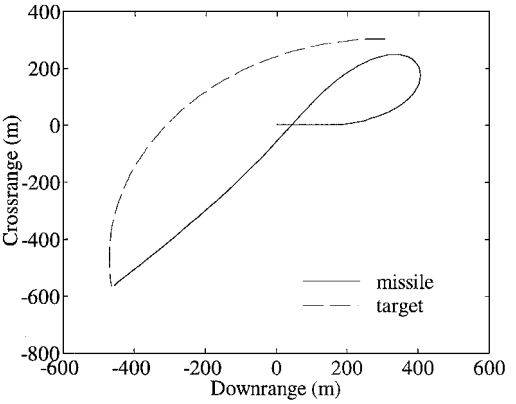

Fig. 6 Intercept trajectories.

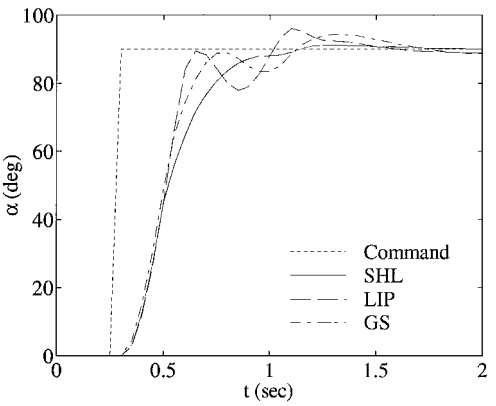

Fig. 7 Angle-of-attack step responses.

Simulation Results

Numerical results have been obtained using a neural-adaptive nonlinear autopilot in a nonlinear, six-degree-of-freedom (6-DOF) simulation of the agile antiair missile featured in Ref. 14. A typical engagement geometry for this missile is the head-on merge scenario shown in Fig. 6. In this example, the missile and target intercept trajectories lie primarily in the local horizontal plane.

Figure 7 compares responses of several autopilot designs to a 90-deg step command in angle of attack. These include a conventional linear gain schedule, the SHL neural network autopilot described earlier, and an autopilot design that employs a LIP neural network as investigated in Ref. 8.

The simpler LIP network required a learning rate as high as 10,000 to achieve desired performance, causing undesirable oscillation in the transient response. In this case, the SHL neural network provides a more effective, though nonlinear, parameterization the inversion error. This parameterization facilitates improved approximation without an excessive learning rate. The SHL neural-network controller surpasses the gain schedule in performance, even with reduced knowledge of the system dynamics.

Moreover, the neural networks in these two controllers have approximately the same number of parameters. Here, the SHL network has 15 inputs, 5 hidden layers, and 3 outputs, which (with thresholds) add up to 98 adjustable weights. The LIP network in Ref. 8 had 10 inputs and performed a second-order polynomial approximation, resulting in a total of 110 unknown coefficients. One disadvantage associated with the introduction of hidden-layer neurons is the increased complexity and conservatism of the stability proof, especially the need for the $\bar{\nu}$ term.

Figure 8 shows the angle of attack and sideslip responses for the engagement shown in Fig. 6. Excellent tracking is achieved, in spite of the nondimensional moment coefficients being approximated as constants in computing the nonlinear inverse control. Turn coordination is preserved by maintaining zero sideslip throughout the intercept. The roll angle histories presented in Fig. 9 indicate that similar performance is achieved in the roll channel.

Figures 10 and 11 illustrate the neural network's ability to reconstruct nonlinear inversion errors in the pitch rate and yaw rate dynamics. The neural-network response illustrated in Fig. 11 indicates that the most severe inversion errors are associated with the yaw rate dynamics. These errors are caused by variations in the aerodynamic moment coefficients with Mach number and angle of attack, which were not modeled in the nonlinear control design. Based on the sideslip response in Fig. 8, however, it is unnecessary to modify the network for improved yaw rate error reconstruction.

Figure 12 shows the neural-network reconstruction of roll rate inversion errors. Modifications that tend to improve the response of the network include adding hidden-layer neurons and increasing the learning rates. No significant improvements were observed, however, for greater than 10 hidden-layer neurons. Five neurons were ultimately used because this number provided a balance between network size and accuracy. Similarly, increasing the learning rates improves the network's approximation of Δ up to the point at which

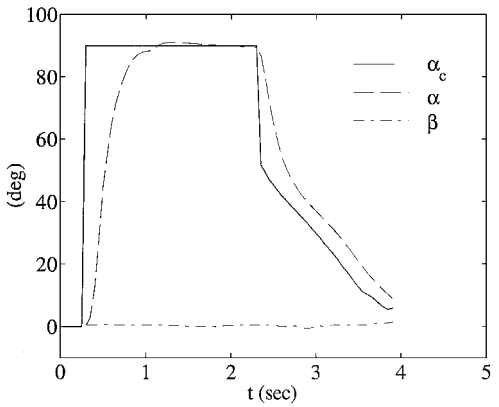

Fig. 8 Angle-of-attack and sideslip responses.

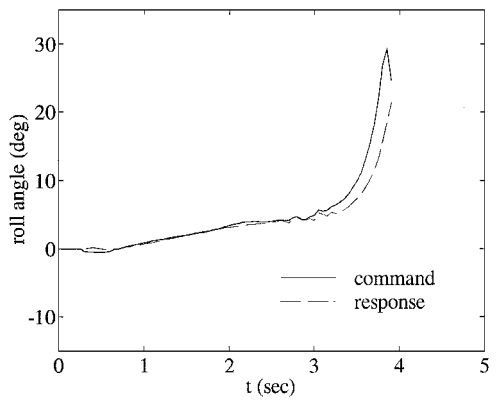

Fig. 9 Roll angle response.

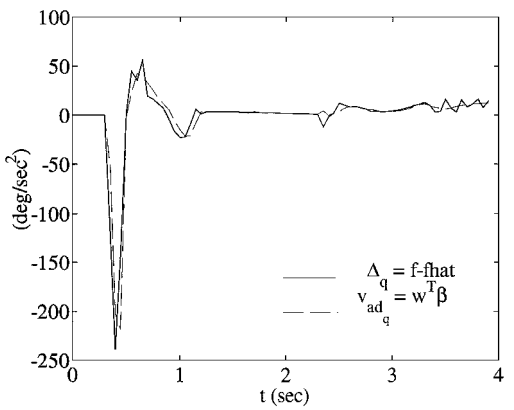

Fig. 10 Pitch rate inversion error reconstruction.

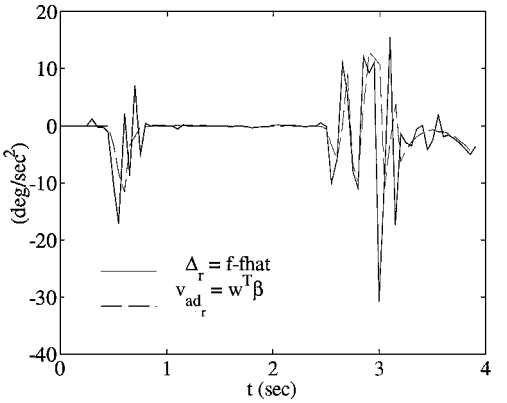

Fig. 11 Yaw rate inversion error reconstruction.

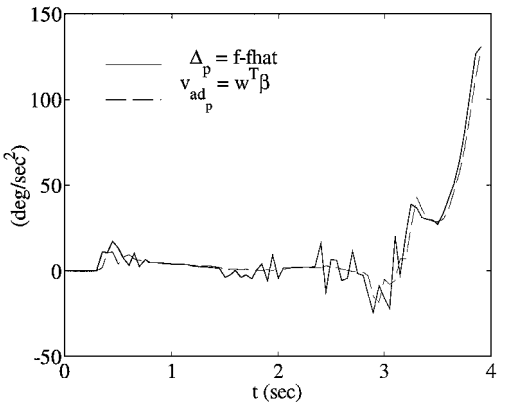

Fig. 12 Roll rate inversion error reconstruction.

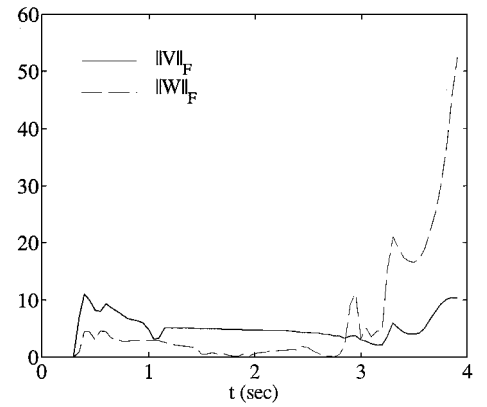

Fig. 13 Neural-network weight update.

the weight update becomes too fast to be adequately represented in the discrete-time simulation.

Finally, Fig. 13 is included simply to illustrate the variation of the weight matrices over time, as captured by $\|\hat{V}\|_F$ and $\|\hat{W}\|_F$. Because the neural network provides only a local approximation of the inversion error, the weights do not approach steady-state values. Instead, learning occurs when conditions vary most rapidly.

Conclusions

A nonlinear bank-to-turn missile autopilot based on approximate dynamic inversion has been proposed. With the aid of an SHL neural network, this autopilot tracks guidance commands in angle of attack and bank angle while holding sideslip angle near zero. A more traditional skid-to-turn autopilot design could be developed simply by replacing the bank angle state with body-axis roll angle and regulating body-axis roll rate to zero.

The effectiveness of this control system has been demonstrated in a nonlinear 6-DOF agile antiair-missile simulation. Numerical

results indicate distinct improvements over LIP networks used in previous work. Moreover, such improvements are achieved using fewer adjustable weights. The price of improved inversion error reconstruction is the need for additional design parameters. Lyapunov stability theory dictates only that the various design gains should be sufficiently large, but excessive conservatism in assigning their values may degrade performance. Nonetheless, we have demonstrated that SHL neural networks enable an approximate nonlinear controller to efficiently adapt on-line to uncertain nonlinear aerodynamic phenomena. This is an especially important feature when, as in the missile autopilot example, these phenomena are difficult or impossible to model accurately for purposes of design and simulation. Furthermore, SHL neural networks possess distinct advantages over simpler LIP networks for this application.

Neural-network-based adaptive control of nonlinear systems is a maturing technology area that promises to be applicable to a wide range of systems. To date, challenging nonlinear control problems characterized by simultaneous nonaffine dependence on plant inputs and nonlinearly parameterized uncertainties have been addressed successfully using these techniques. Meanwhile, other research has made progress in the area of robustness to unmodeled dynamics. Currently active research is directed toward uniting neural networks with hidden layers and robustness to unmodeled dynamics in a single control design.

Acknowledgment

This research was partially sponsored by the U.S. Air Force Research Laboratory under Contract F08630-95-1-0006.

References

[1]Tahk, M., Briggs, M., and Menon, P. K. A., "Application of Plant Inversion via State Feedback to Missile Autopilot Design," *Proceedings of the 27th Conference on Decision and Control*, 1986, pp. 730–735.

[2]Innocenti, M., and Thukral, A., "Simultaneous Reaction Jet and Aerodynamic Control of Missile Systems," *Proceedings of the AIAA Guidance, Navigation, and Control Conference*, AIAA, Washington, DC, 1993, pp. 347–354.

[3]McDowell, D. M., Irwin, G. W., and McConnell, G., "Online Neural Control Applied to a Bank-to-Turn Missile Autopilot," *Proceedings of the AIAA Guidance, Navigation, and Control Conference*, AIAA, Washington, DC, 1995, pp. 1286–1294.

[4]Calise, A. J., and Rysdyk, R. T., "Nonlinear Adaptive Flight Control Using Neural Networks," *IEEE Control Systems Magazine*, Vol. 18, No. 6, 1998, pp. 14–25.

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

[6]Leitner, J., Calise, A., and Prasad, J. V. R., "Analysis of Adaptive Neural Networks for Helicopter Flight Controls," *Journal of Guidance, Control, and Dynamics*, Vol. 20, No. 5, 1997, pp. 972–979.

[7]McFarland, M. B., and Calise, A. J., "Neural Networks for Stable Adaptive Control of Air-to-Air Missiles," *Proceedings of the AIAA Guidance, Navigation, and Control Conference*, AIAA, Washington, DC, 1995, pp. 1280–1285.

[8]McFarland, M. B., and Calise, A. J., "Neural-Adaptive Nonlinear Autopilot Design for an Agile Anti-Air Missile," AIAA Paper 96-3914, Aug. 1996.

[9]Lewis, F. L., Yesildirek, A., and Liu, K., "Multilayer Neural-Net Robot Controller with Guaranteed Tracking Performance," *IEEE Transactions on Neural Networks*, Vol. 7, No. 2, 1996, pp. 388–399.

[10]McFarland, M. B., "Adaptive Nonlinear Control of Missiles," Ph.D. Dissertation, School of Aerospace Engineering, Georgia Inst. of Technology, Atlanta, GA, Sept. 1997.

[11]Krstic, M., Kanellakopoulos, I., and Kokotovic, P., *Nonlinear and Adaptive Control Design*, Wiley, New York, 1995, pp. 70–73.

[12]McFarland, M. B., and Calise, A. J., "Robust Adaptive Control of Uncertain Nonlinear Systems Using Neural Networks," *Proceedings of the American Control Conference*, Inst. of Electrical and Electronics Engineers, Piscataway, NJ, 1997, pp. 1996–2000.

[13]Rysdyk, R. T., Nardi, F., and Calise, A. J., "Robust Adaptive Nonlinear Flight Control Applications Using Neural Networks," *Proceedings of the American Control Conference*, Inst. of Electrical and Electronics Engineers, Piscataway, NJ, 1999, pp. 2595–2599.

[14]Wise, K. A., and Broy, D. J., "Agile Missile Dynamics and Control," AIAA Paper 96-3912, Aug. 1996.