# Adaptive Feedback Control by Constrained Approximate Dynamic Programming

Silvia Ferrari, *Member, IEEE*, James E. Steck, and Rajeev Chandramohan

***Abstract*—A constrained approximate dynamic programming (ADP) approach is presented for designing adaptive neural network (NN) controllers with closed-loop stability and performance guarantees. Prior knowledge of the linearized equations of motion is used to guarantee that the closed-loop system meets performance and stability objectives when the plant operates in a linear parameter-varying (LPV) regime. In the presence of unmodeled dynamics or failures, the NN controller adapts to optimize its performance online, whereas constrained ADP guarantees that the LPV baseline performance is preserved at all times. The effectiveness of an adaptive NN flight controller is demonstrated for simulated control failures, parameter variations, and near-stall dynamics.**

***Index Terms*—Approximate dynamic programming (ADP), constrained optimization, feedback control, neural networks (NNs).**

## I. Introduction

THE AEROSPACE community has long been interested in applying adaptive neural networks (NNs) to flight control in the hope of handling unexpected failures and emergency maneuvers. This problem is very challenging because, although the adaptation must handle a variety of operating conditions, the controller must maintain a high level of safety and performance at all times. On the other hand, adaptive NN controllers could greatly impact commercial and general aviation aircraft. One goal of the industry is to produce a safer aircraft by eliminating loss of control and adapting to failure. Another goal is to build an aircraft that is easier to fly and cheaper to operate by optimizing its response to pilot command inputs and minimizing control usage.

There is considerable precedent for applying classical control designs and approximate dynamic programming (ADP) to neurocontrol to handle uncertainties and nonlinear plant dynamics [1]–[7]. A common approach consists of superimposing an adaptive neural element onto a classical control structure to handle uncertainties. However, by assuming a special form for the plant dynamics, these designs may display poor performance and robustness elsewhere in the operating domain. In principle, ADP NN controllers can optimize performance subject to any plant dynamics. However, they are often criticized for the lack of closed-loop stability and performance guarantees that characterize classical controllers. A few approaches based on Lyapunov functions [8], [9] and linear difference inclusion [1] have been proposed to prove the closed-loop stability of sigmoidal NNs. However, a methodology is still missing for synthesizing ADP NN controllers that meet classical control objectives and are closed-loop stable. In this paper, a controller entirely composed of a recurrent sigmoidal NN adapts online by a constrained ADP algorithm that preserves classical performance and stability guarantees.

Manuscript received March 5, 2008; revised March 18, 2008. This work was supported by the National Science Foundation under Grant ECS 0300236 and CAREER Award ECS 0448906. This paper was recommended by Guest Editor D. Liu.

S. Ferrari is with the Department of Mechanical Engineering, Duke University, Durham, NC 27708 USA.

J. E. Steck and R. Chandramohan are with the Department of Aerospace Engineering, Wichita State University, Wichita, KS 67260-0044 USA.

Digital Object Identifier 10.1109/TSMCB.2008.924140

## II. NN Control Objectives and Specifications

Modern gain-scheduled controllers are capable of providing satisfactory performance in the linear parameter-varying (LPV) regime of the plant, i.e., $\mathbf{X}_{\mathrm{LPV}}$. However, they are not designed to handle unforeseen operating conditions or failures that cause the plant to abandon $\mathbf{X}_{\mathrm{LPV}}$. Many real plants are characterized by a high-dimensional operating domain $\mathbf{X}$ and experience a wide range of unmodeled dynamics and parameter variations over their lifetime. Therefore, it would be infeasible to design the controller for all possible conditions *a priori*. A NN controller can adapt online and account for non-LPV regimes only if and when they arise. At the same time, it must provide the same performance and safety guarantees as classical linear designs in $\mathbf{X}_{\mathrm{LPV}}$, where the plant operates most of the time.

Consider a plant whose dynamics can be approximated by the following nonlinear differential equation:

$$\begin{cases} \dot{x} = f(x, p_m, u) \\ y = h(x, p_m, u) \end{cases} \tag{1}$$

where $x \in \mathbf{X} \subset \mathbb{R}^{n\times 1}$ is the state, $u \in \mathbf{U} \subset \mathbb{R}^{m\times 1}$ is the control, $p_m \in \mathbb{R}^{\wp\times 1}$ is a vector of parameters, and $y \in \mathbb{R}^{r\times 1}$ is the output. The differential equation structure and parameters $p_m$ are not always known *a priori* and are subject to change. The complete range of dynamics that the plant can experience $\{\mathbf{X}, \mathbf{U}\}$ is its *full operating domain* and is assumed to be observable. The control objectives are expressed by the following integral cost function:

$$J = \lim_{t_f\to\infty} \left\{ \frac{1}{t_f} \int_{t_0}^{t_f} \mathcal{L}\left[x(\tau), u(\tau)\right] d\tau \right\} \tag{2}$$

to be minimized with respect to the control law $u = c(x)$.

Typically, the plant operates in a subset of the state space $\mathbf{X}_{\mathrm{LPV}} \subset \mathbf{X}$ in which plant dynamics can be closely approximated by a class of affine nonlinear systems

$$\begin{cases} \Delta\dot{x} = F(x)\Delta x + G(x)\Delta u \\ \Delta y = H_x(x)\Delta x + H_u(x)\Delta u \end{cases} \tag{3}$$

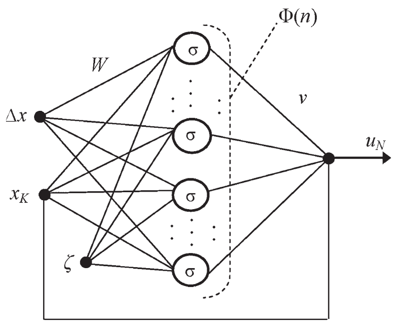


Fig. 1. State-feedback dynamic NN controller.

where $\Delta$ denotes deviations from the equilibrium. The functions $F$, $G$, $H_x$, and $H_u$ can be approximated by a set of matrices $A_j \approx F(\zeta_j)$, $B_j \approx G(\zeta_j)$, $C_j \approx H_x(\zeta_j)$, and $D_j \approx H_u(\zeta_j)$ for $p$ equilibria or *scheduling vectors* $\{\zeta_1, \ldots, \zeta_p\} \in \mathbf{X}_{\rm LPV}$, where $\zeta \in \mathbb{R}^{z \times 1}$ contains time-varying parameters and state variables that significantly influence plant dynamics. Then, there exists a linear time-invariant (LTI) model that provides a satisfactory representation of plant dynamics near $\zeta_j \in \mathbf{X}_{\rm LPV}$ and can be represented by the transfer function $P_j(s)$, where

$$P_j(s): \begin{cases} \Delta\dot{x} = A_j \Delta x + B_j \Delta u \\ \Delta y = C_j \Delta x + D_j \Delta u \end{cases} \quad j = 1, \ldots, p. \tag{4}$$

A dynamic state-feedback control law is given by the NN

$$\begin{cases} \dot{x}_K = A_K(\zeta) x_K + B_K(\zeta) \Delta y \\ \Delta u = u_N := v\Phi(W x_a) \end{cases} \tag{5}$$

with input $x_a := [\,\chi^T \quad \zeta^T\,]^T$, where $\chi := [\,\Delta x^T \quad x_K^T\,]^T \in \mathbb{R}^{\nu \times 1}$, and the controller state $x_K \in \mathbb{R}^{\kappa \times 1}$ is a function of $u_N$ (Fig. 1). The control is assumed to be scalar without loss of generality. The adjustable parameters $v \in \mathbb{R}^{1 \times l}$ and $W \in \mathbb{R}^{l \times (\nu + z)}$, and the matrix functions $A_K(\zeta) \in \mathbb{R}^{\kappa \times \kappa}$ and $B_K(\zeta) \in \mathbb{R}^{\kappa \times r}$, are determined in Section IV. $\Phi$ is a diagonal operator with repeated sigmoids, i.e.,

$$\Phi(n) := [\,\sigma(n_1) \quad \cdots \quad \sigma(n_l)\,]^T \tag{6}$$

where $n_i$ denotes the $i$th component of a signal $n \in \mathbb{R}^{l \times 1}$. In this paper, $\sigma(n_i) := (e^{n_i} - 1)/(e^{n_i} + 1)$; thus, $\sigma : \mathbb{R} \to \mathbb{R}$ is monotonically nondecreasing, is slope restricted, and belongs to the sector $[\alpha, \beta]$, with $\alpha = 0$ and $\beta = 1/2$.

The design objectives of the NN controller are given as follows.

1) Provide the same performance and stability guarantees as a classical linear controller given *a priori* knowledge of the plant dynamics in $\mathbf{X}_{\rm LPV}$.
2) Adapt online to accommodate for plant dynamics that are unknown *a priori*, or arise in $\{\mathbf{X} \backslash \mathbf{X}_{\rm LPV}\}$, by optimizing the same control objectives used in objective 1).
3) Satisfy objective 1) while adapting according to objective 2) to preserve the desired LPV performance baseline at all times.

## III. Review of IQCs

This section reviews the results from [10] on integral quadratic constraints (IQCs) for monotonic and slope-restricted diagonal operators. IQCs are used to analyze feedback interconnections between an LTI operator

$$H(s): \begin{cases} \dot{\chi} = A\chi + Bg \\ n = C\chi + Eg \end{cases} \tag{7}$$

and a bounded causal operator $\Phi$ that is possibly nonlinear, where $\chi \in \mathbb{R}^{\nu \times 1}$, $n \in \mathcal{L}_{2e}^{l}$, $g \in \mathcal{L}_{2e}^{m}$, and $\mathcal{L}_{2e}^{n}$ denotes the linear space of all functions $\phi : (0, \infty) \to \mathbb{R}^n$, which are square integrable on any finite interval.

Assume that $\Phi$ can be described by the following IQC inequality:

$$\int_{-\infty}^{\infty} \begin{bmatrix} N(j\omega) \\ G(j\omega) \end{bmatrix}^A \Pi(j\omega) \begin{bmatrix} N(j\omega) \\ G(j\omega) \end{bmatrix} d\omega \geq 0 \tag{8}$$

where $N(j\omega)$ and $G(j\omega)$ denote the Fourier transforms of $n(t)$ and $g(t)$ at frequency $\omega$, respectively, $(\cdot)^A$ denotes the adjoint of a matrix, and $\Pi$ is a Hermitian matrix function. Then, under appropriate assumptions [11, Th. 1], the IQC stability theorem states that if there exists $\epsilon > 0$ such that

$$\begin{bmatrix} H(j\omega) \\ I \end{bmatrix}^A \Pi(j\omega) \begin{bmatrix} H(j\omega) \\ I \end{bmatrix} \leq -\epsilon I, \qquad \forall \omega \in \mathbb{R} \tag{9}$$

then the feedback interconnection of $H$ and $\Phi$ is globally exponentially stable. The class $\mathbf{\Pi}_\Phi$ of all $\Pi$ that define a valid IQC for a particular operator $\Phi$ is convex and, in some cases, can be readily found in the literature [11].

If $\Phi$ is a monotonic and slope-restricted diagonal operator with repeated nonlinearity, e.g., (6), the search for a suitable $\Pi \in \mathbf{\Pi}_\Phi$ can be restricted to a finite-dimensional subset, and, by applying the Kalman–Yakubovich–Popov lemma [11], the inequality in (9) can be transformed into the following feasibility problem [10]:

$$\begin{aligned} & T = T^T Q = Q^T \\ & T_{ii} \geq \sum_{j=1, j \neq i}^{l} |T_{ij}|, \qquad i = 1, \ldots, l \\ & \begin{bmatrix} A^T Q + QA & QB + C^T T \\ (QB)^T + TC & -2T + E^T T + TE \end{bmatrix} < 0 \end{aligned} \tag{10}$$

where $T \in \mathbb{R}^{l \times l}$ and $Q \in \mathbb{R}^{\nu \times \nu}$ are constant matrices.

## IV. LPV Performance of the NN Controller

Several gain-scheduling techniques, including multivariable control, $\mu$-synthesis, and linear matrix inequalities, have been developed to design parameter-dependent controllers in the form

$$K(s, \zeta): \begin{cases} \dot{x}_K = A_K(\zeta) x_K + B_K(\zeta) \Delta y \\ \Delta u = u_K := C_K(\zeta) x_K + D_K(\zeta) \Delta y \end{cases} \tag{11}$$

that meet multiple design objectives, such as $H_\infty$ and $H_2$ performance and pole placement, for a plant in $\mathbf{X}_{\rm LPV}$ [12]. In gain scheduling, the state-space matrices in (11) are obtained by interpolating a set of LTI controllers $\mathcal{K} = \{K_1, \ldots, K_p\}$, where

$$K_j := \begin{pmatrix} A_{K_j} & B_{K_j} \\ C_{K_j} & D_{K_j} \end{pmatrix} := \begin{pmatrix} A_K(\zeta_j) & B_K(\zeta_j) \\ C_K(\zeta_j) & D_K(\zeta_j) \end{pmatrix} \tag{12}$$

and $j = 1, \ldots, p$ [12]. Here, it is assumed that $\mathcal{K}$ is provided by a gain-scheduling technique and, thus, can be used to specify the performance of the NN controller (5) in $\mathbf{X}_{\mathrm{LPV}}$, i.e.,

*Theorem 1:* Given a set of LTI controllers $\mathcal{K}$, there exists a controller (5), with $l = p$, that is input–output equivalent to (11) at the equilibria $\{\zeta_1, \ldots, \zeta_p\} \in \mathbf{X}_{\mathrm{LPV}}$, i.e., satisfies the *closed-loop* requirements

$$\begin{cases} u_N(\zeta_j) = u_K(\zeta_j) \\ \frac{\partial u_N}{\partial \dot{\chi}}(\zeta_j) = \frac{\partial u_K}{\partial \dot{\chi}}(\zeta_j) \end{cases} \quad j = 1, \ldots, p \tag{13}$$

if there exist a matrix $W = [W_\chi \quad W_\zeta]$ and a vector $v^T \in \mathbb{R}^{l \times 1}$ that satisfy the linear systems

$$\begin{cases} N = W_\zeta Z \\ S v^T = b \\ DVW_\chi = M_2 \end{cases} \tag{14}$$

and provided that the matrices $(I - D_j D_{K_j})$ are invertible, where

$$M_{1_j} := (I - D_j D_{K_j})^{-1} [\, C_j \quad D_j C_{K_j} \,] \tag{15}$$
$$M_{2_j} := [\, 0 \quad C_{K_j} \,] + D_{K_j} M_{1_j} \tag{16}$$
$$M_2 := [\, M_{2_1}^T \quad \cdots \quad M_{2_p}^T \,]^T \tag{17}$$
$$Z := [\, \zeta_1 \quad \cdots \quad \zeta_p \,] \tag{18}$$
$$N := [\, n^1 \quad \cdots \quad n^p \,] \tag{19}$$
$$S := [\, \Phi(n^1) \quad \cdots \quad \Phi(n^p) \,]^T \tag{20}$$
$$D := [\, \Phi'(n^1) \quad \cdots \quad \Phi'(n^p) \,]^T . \tag{21}$$

$V = \mathrm{diag}(v)$, $b := \tilde{b}\imath_p$, $\imath_p := 1_{p \times 1}$, $\tilde{b}$ is a known constant, and $\Phi'$ is defined in terms of $\sigma' = d\sigma(n_i)/dn_i$ as

$$\Phi'(n) := [\, \sigma'(n_1) \quad \cdots \quad \sigma'(n_l) \,]^T . \tag{22}$$

A proof is provided in [13]. The systems in (14) are linear in $N$, $W_\zeta$, $v$, and $W_\chi$ provided that they are solved in this order.

Closed-loop stability conditions are obtained by showing that at $\zeta_j \in \mathbf{X}_{\mathrm{LPV}}$, the feedback interconnection of (3) and (5) is an IQC interconnection [11]. In fact, the closed-loop dynamics of the NN-controlled system at $\zeta_j$ are given by

$$\begin{cases} \dot{\chi_{cl}} = A_{cl}(\zeta_j)\chi_{cl} + B_{cl}(\zeta_j) v g \\ n = W \chi_{cl} \\ g = \Phi(n) \end{cases} \quad j = 1, \ldots, p \tag{23}$$

where

$$A_{cl}(\zeta_j) := \begin{bmatrix} A_j & 0 \\ B_{K_j} C_j & A_{K_j} \end{bmatrix} \quad B_{cl}(\zeta_j) := \begin{bmatrix} B_j \\ B_{K_j} D_j \end{bmatrix}.$$

Although (10) could be used for stability analysis, it is not as useful for synthesis because it is not linear with respect to $v$ and $W$. Thus, the following result is proven in [13].

*Theorem 2:* Assume that $A_{cl}(\zeta_j) \in \mathbb{R}^{\nu \times \nu}$ is asymptotically stable $\forall \zeta_j \in \mathbf{X}_{\mathrm{LPV}}$, and $v$ and $W$ satisfy (14). Then, if there exist $p$ symmetric matrices $P_j \in \mathbb{R}^{p \times p}$ such that $P_j = S T_j S^T$, with $T_j$ diagonally dominant, and

$$\begin{bmatrix} \mathcal{W} & \beta P_j^T \\ \beta P_j & -2P_j + \mathcal{Q}_j^T R_1^{-1} \mathcal{Q}_j \end{bmatrix} \leq 0, \qquad j = 1, \ldots, p \tag{24}$$

where

$$\mathcal{W} := -\left[\Pi \, R_2^{-1} \, \Pi^T\right]^{-1} \tag{25}$$
$$\Pi := \left[(DVS^T)^{-1} M_2 \quad S^{-T} N Z^+\right] \tag{26}$$
$$\mathcal{Q}_j := Q_j B_{cl}(\zeta_j) b^T \tag{27}$$

and $Q_j$ is a known positive-definite matrix obtained from

$$A_{cl}^T(\zeta_j) Q_j + Q_j A_{cl}(\zeta_j) = -R_1 - R_2 \tag{28}$$

the feedback interconnections in (23) are exponentially stable.

## V. Constrained Adaptation in Non-LPV Regimes

ADP aims at overcoming the curse of dimensionality by discretizing $t \in [0, t_f)$ by a fixed interval $\Delta t$ and by embedding the optimization of (2) into the optimization of a value function

$$V_D\left[x(t_k), c\right] = \lim_{t_f \to \infty} \left\{ \frac{1}{t_f} \sum_{t_i = t_k}^{t_f - \Delta t} \mathcal{L}_D\left[x(t_i), u(t_i)\right] \right\} \tag{29}$$

subject to a suitable discrete-time model of (1) [14, p. 144]. From hereon, the subscript $(\cdot)_D$ will be omitted for simplicity. Howard [15] showed that if the control law and value function approximations $c_\ell$ and $V_\ell$ are updated by iterating between a policy improvement routine

$$c_{\ell+1}\left[x(t_K)\right] = \arg \min_{u(t_K)} \left\{ \mathcal{L}\left[x(t_K), u(t_K)\right] + V_\ell\left[x(t_K), c_\ell\right] \right\} \tag{30}$$

and a value determination operation

$$V_{\ell+1}\left[x(t_K), c_{\ell+1}\right] = \mathcal{L}\left[x(t_K), u(t_K)\right] + V_\ell\left[x(t_K), c_{\ell+1}\right] \tag{31}$$

they eventually converge to their optimal counterparts, where $x(t_{k+1})$ is computed from the discrete-time model of (1). Furthermore, at each iteration $\ell$, these two approximations are improved and are closer to optimal than their predecessors. As proven in [16], the above algorithm can be implemented *over time* by letting $\ell = k$ such that $c_\ell$ and $V_\ell$ are improved at every time step $t_k$ until they eventually converge to the optimal functions $c^*$ and $V^*$, respectively.

### A. Constrained Policy Improvement Routine

The ADP algorithms (30) and (31) are locally implemented using only one state sample at every iteration $\ell = k$. The control law approximation is provided by the dynamic NN (5), i.e.,

$$c_\ell\left[x_a(t)\right] := v^{[\ell]} \Phi\left[W^{[\ell]} x_a(t)\right] + u_{0_j}, \quad \ell = 0, 1, 2, \ldots. \tag{32}$$

Thus, it is updated by modifying the parameter values $v^{[\ell]}$ and $W^{[\ell]}$ at every policy improvement iteration $\ell$. $u_{0_j}$ is a constant nominal control vector that is obtained using (4). Every control design objective defines a quadratic cost index

$$J_i = \lim_{t_f \to \infty} \left[ \frac{1}{t_f} \sum_{t_k=0}^{t_f-\Delta t} \xi_i^T(t_K)\xi_i(t_K) \right] \quad (33)$$

formulated in terms of the state and the control. For example, both $H_2$ performance and pole placement can be expressed by a cost index $\xi_i(t_k) = M_i^{1/2}[\,x_a^T(t_k) \quad u^T(t_k)\,]^T$ through a symmetric weighting matrix $M_i$ of design parameters [14, Sec. 6.3]. Suppose that the control objectives define a set of $q$ cost indexes $\{J_1, \ldots, J_q\}$ to be minimized. Then, (29) is readily defined by letting $\mathcal{L}_D = \sum_{i=1}^{q} \xi_i^T(t_k)\xi_i(t_k)$, and the value-function approximation is updated using (31) [16].

At $\ell = 0$, $v^{[0]}$ and $W^{[0]}$ are set equal to the values obtained from the LPV design equations (14) and (24). Then, if the plant leaves $\mathbf{X}_{\mathrm{LPV}}$ or deviates from (3) due to failures or parameter variations, the optimality condition in [3] is violated, setting off the ADP adaptation. To preserve the LPV performance, equality constraints obtained from (14) and (24) are adjoined to (30), leading to the following constrained optimization problem:

$$\begin{aligned} \text{minimize } e_\ell(w) := & \sum_{i=1}^{q} \hat{\xi}_i^T(t_K)\hat{\xi}_i(t_K) \\ & + V_\ell\big[f_D\left[\hat{x}(t_K), p_m(t_K), c_\ell\left[\hat{x}_a(t_K)\right]\right], \\ & \quad c_\ell\left[\hat{x}_a(t_K)\right]\big] \end{aligned} \quad (34)$$

$$\text{subject to } F(w) := \begin{bmatrix} N - W_\varsigma Z \\ Sv^T - b \\ DVW_\chi - M_2 \\ \mathcal{W} + \left[\Pi R_2^{-1}\Pi^T\right]^{-1} \end{bmatrix} = 0 \quad (35)$$

where $(\hat{\cdot})$ denotes a vector evaluated at the actual state value observed from the plant online. $v$ and $W$ are rearranged into a vector of variables $w = [\,w_1 \quad \cdots \quad w_N\,]^T \in \mathbb{R}^{N\times 1}$. The known constants $Z$, $b$, $M_2$, $R_2$, and $\mathcal{W}$ and the matrix functions of $w$ ($N$, $S$, $D$, $V$, and $\Pi$) are all defined in Section IV.

### B. Constrained NN Parameter Update

The constrained training approach presented in [17] can be used to minimize (34) subject to (35). The solution of a constrained optimization problem can be pursued by the method of Lagrange multipliers or by direct elimination [18]. As a first step, $w$ is partitioned into two sets of weights, i.e., $\omega \in \mathbb{R}^{L\times 1}$ and $s \in \mathbb{R}^{S\times 1}$, such that $L + S = N$, and (35) can be written as $F(\omega, s) = 0$. If $(\partial F/\partial \omega)|_s \neq 0$, then the constraints (35) uniquely imply a function

$$\omega = \mathcal{H}(s) \quad (36)$$

which can be used in place of (35) to simplify the solution of the constrained optimization [18]. In this case, the method of direct elimination can be applied by writing (34) as

$$E_\ell(s) = e_\ell\left(\mathcal{H}(s), s\right) \quad (37)$$

such that $s$ can be determined independently of $\omega$. Then, $\omega$ is used to satisfy (36), and $s$ is used to minimize (37). It also follows that the solution of (34) and (35) is an extremum of (37) that obeys $\partial E_\ell/\partial s|_{s^*} = 0$. Once the optimal value $s^*$ is determined, $\omega^*$ can be obtained from $s^*$ using (36). Furthermore, the substitution of (36) into (37) can be circumvented by implementing the *adjoined error gradient* [17]

$$\frac{\partial E_\ell}{\partial s} = \frac{\partial e_\ell}{\partial s} + \frac{\partial e_\ell}{\partial \omega}\frac{\partial \mathcal{H}}{\partial s} := \partial_s e_\ell + \partial_\omega e_\ell \partial_s \mathcal{H} \quad (38)$$

$\partial_x y$ denotes the gradient of $y$ with respect to $x$. $\partial_s e_\ell$ and $\partial_\omega e_\ell$ are obtained by back-propagation, and the Jacobian $\partial_s \mathcal{H}$ is derived from (36) using the properties in [17].

If $(\partial F/\partial \omega)|_s = 0$, the method of Lagrange multipliers can be used to seek the solution of (34) and (35). By this method, (35) is adjoined to (34) obtaining the augmented error function

$$E_{a_\ell}(w) \equiv e_\ell(w) - \lambda^T F(w). \quad (39)$$

$\lambda \in \mathbb{R}^{M\times 1}$ is a vector of Lagrange multipliers, where $M$ is the number of equality constraints in (35). As shown in [14], in the vicinity of an extremum of (39), $\lambda$ takes the value

$$\lambda^* = -(\partial_w F)^{-T}(\partial_w e_\ell)^T \quad (40)$$

where $-T$ denotes the inverse transpose of a matrix. Then, the optimal value of $w$ can be determined from

$$\partial_w e_\ell - (\lambda^*)^T \partial_w F = 0 \quad (41)$$

using quasi-Newton or Newton–Raphson algorithms.

**Algorithm 1** Pseudocode of constrained ADP algorithm

```
given e_ℓ(·), H(·), and (∂E_ℓ/∂s)(·);
compute ω^[0] and s^[0], such that F(ω^[0], s^[0]) = 0;
for ℓ = k = 0, 1, 2, ... do
  obtain x̂_a(t_k);
  if optimality condition is violated, then
    i = 0;
    ω^[i] = ω^[ℓ] and s^[i] = s^[ℓ];
    while e_ℓ(x̂_a(t_k), ω^[i], s^[i]) > e_tol do
      s^[i+1] = s^[i] − η^[i](∂E_ℓ^[i]/∂s);
      ω^[i+1] = H(s^[i+1])
      i = i + 1
    end while
    ω^[ℓ+1] = ω^[i] and s^[ℓ+1] = s^[i];
    obtain c_{ℓ+1}(·) from (32)
    obtain V_{ℓ+1}(·) from (31)
  end if
end for
```

The convergence of the unconstrained ADP algorithms (30) and (31) to $c^*$ and $V^*$ was proven in [16, pp. 89–92]. The parameter update in Algorithm 1 is performed by a gradient-based line search algorithm (e.g., [7]) that obeys the Armijo rule and, thus, is proven to converge in [18, Prop. 1.8]. When direct elimination is applicable, the constraints merely reduce the order of the minimization in the constrained ADP algorithm (34) [14, p. 36]. Otherwise, the method of Lagrange multipliers can be applied, with local convergence that is at least linear or, in some cases, superlinear [18, p. 231].

Since direct elimination and Lagrange multipliers can be effectively implemented by first-order gradient-descent and Lagrangian methods, respectively, the time complexity of Algorithm 1 is dictated by the computation of $\partial_s E_\ell$ or $\partial_w e_\ell$. $\partial_w e_\ell$ can be computed by multiplication of two matrices that are at worse $N \times N$ [19] and, thus, requires time $O(N^3)$,

TABLE I
MSE OF THE LPV DESIGN EQUATIONS

| **Adaptation Method** | $t_0 = 0$ s | $t_{50} = 5$ s | $t_{100} = 10$ s |
|---|---|---|---|
| *Unconstrained* ADP | $1.364 \cdot 10^{-7}$ | $8.850 \cdot 10^{11}$ | $1.584 \cdot 10^{11}$ |
| *Constrained* ADP | $1.364 \cdot 10^{-7}$ | $1.202 \cdot 10^{-7}$ | $2.778 \cdot 10^{-7}$ |

or $O(N^{2.81})$ by Stressen's algorithm. As shown in [17], the computation of $\partial_s \mathcal{H}$ in (38) requires inverting an $(M-1) \times (M-1)$ matrix $M^2$ times, which requires $O(M^{4.81})$ time. Although, typically, $M \ll L, S$, $\partial_s \mathcal{H}$ remains the most expensive computation in (38), and, therefore, the worse-case computation time of $\partial_s E_\ell$ is $O(M^{4.81})$. As an example, for a NN with $L = 15$, $S = 10$, $M = 15$, and $n = m = 2$, this computation requires 0.04 s and 1.6-kB storage on a laptop computer with a 2.2-GHz Intel Core Duo CPU.

## VI. Application to Adaptive Aircraft Control

The approach presented in this paper was used to develop adaptive NN controllers for a business jet and a Hawker Beechcraft Bonanza aircraft in [17] and [20]. The aircraft dynamics in $\mathbf{X}$ are simulated by a 12th-order nonlinear differential equation obtained from mathematical models and wind-tunnel data [13]. The control inputs are the throttle, the elevator, the aileron, and the rudder. The aircraft flight envelope $\mathbf{X}_{\rm LPV}$ and the $p$ scheduling vectors used in the LPV design equations are shown in [17] and [20]. The scheduling variables are the aircraft airspeed $V$ and the altitude $H$. The value of $\hat{x}_a(t_k)$ is observed once every $\Delta t = 0.1$ s and is used by Algorithm 1 to update $s$ and $\omega$ over a few epochs (indexed by $i$). The results summarized in this section are obtained for a NN (5) with $l = p = 36$ sigmoids, $(\nu + \kappa) = 8$ inputs, and $m = 4$ outputs.

The numerical experiments show that the NN controller optimizes the performance during nonlinear maneuvers in $\mathbf{X}$, control failures, parameter variations, and unmodeled dynamics with no prior knowledge of these conditions. Also, anytime the aircraft flies in $\mathbf{X}_{\rm LPV}$ (before or after adaptation), the NN controller performs optimally [17]. The mse of the LPV design equations (14) and (24) is shown in Table I for sample adaptation times during a large-angle maneuver [17]. For comparison, an unconstrained but otherwise equivalent NN controller is implemented during the same maneuver. Whereas the mse of the constrained NN controller remains virtually unchanged, the mse of the unconstrained NN controller increases by several orders of magnitude, illustrating that its LPV performance deteriorates over time as a result of the frequent parameter updates.

A partial elevator failure is simulated in Fig. 2 by changing the parameters $C_{m_{\delta E}}$ and $C_{L_{\delta E}}$, representing pitching moment and lift stability derivatives, respectively, by 10% at $t_0 = 0$. As a result, the LPV performance baseline (11) is no longer optimal, and the ADP adaptation is set off, leading to approximately 50% reduction in the path angle overshoot. The ability of the NN controller to improve performance in the presence of parameter variations is tested by modifying the stability derivative $C_{m_\alpha}$, representing the influence of the angle of attack $\alpha$ on the pitching moment coefficient $C_m$, and the derivative due to the pitching velocity $C_{m_q}$. These aerodynamic parameters determine the pitching moments that are exerted on the aircraft as a result of the aircraft orientation and velocity relative to the

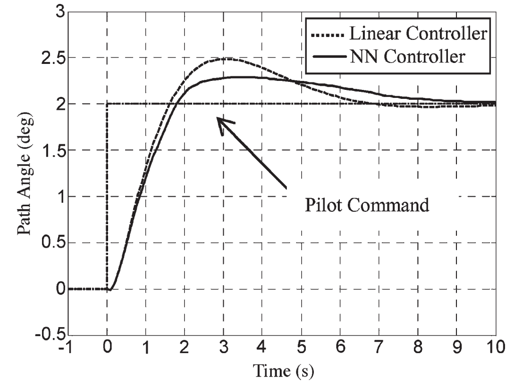


Fig. 2. Aircraft response to a pilot command of 2° path angle, at $H = 3000$ ft and $V = 181.23$ ft/s, in the presence of a partial elevator failure.

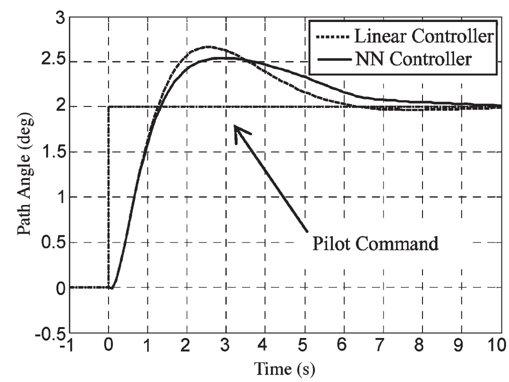


Fig. 3. Aircraft response to a pilot command of 2° path angle, at $H = 3000$ ft, $V = 181.23$ ft/s, and with a 10% variation in $C_{m_\alpha}$ and $C_{m_q}$.

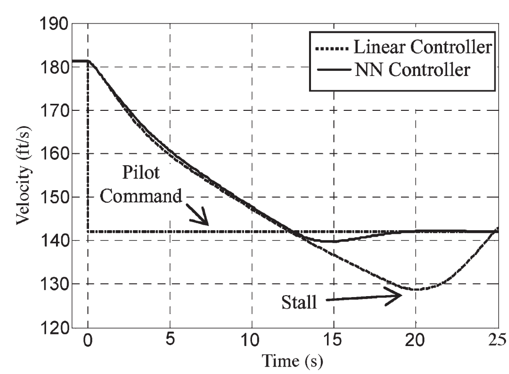


Fig. 4. Aircraft response to a pilot command of −40 ft/s change in airspeed leading to stall.

airflow. As shown in Fig. 3, when $C_{m_\alpha}$ and $C_{m_q}$ are modified by 10%, the adaptive NN controller reduces the path angle overshoot by approximately 25%.

To test the NN controller's ability to handle unmodeled dynamics, a stall model is included in the lift curve of the simulated Bonanza aircraft, without accounting for it in the control design. As shown in Fig. 4, the aircraft is flying at a steady level when, at $t_0 = 0$ s, the pilot commands a $-40$ ft/s decrease in airspeed and brings the aircraft near stall at $t_k \approx 20$ s when $\alpha > 9^\circ = \alpha_{\rm stall}$ (Fig. 5) and $V \approx 135$ ft/s. At this speed, the nonlinearity of the resulting drag and lift effects becomes so significant as to cause the linear controller (11) to loose control of the aircraft (Fig. 6). The NN controller is

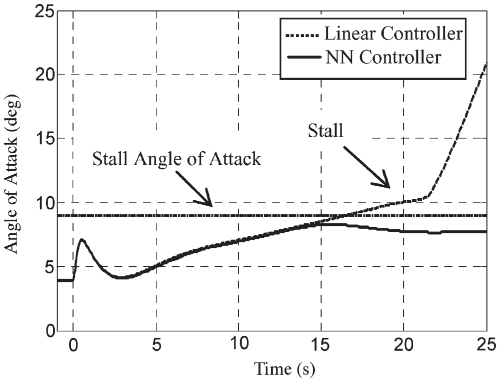


Fig. 5. Adaptive NN controller averts stall during the maneuver in Fig. 4 near $\alpha_{\text{stall}} = 9^{\circ}$ for $V \approx 135$ ft/s.

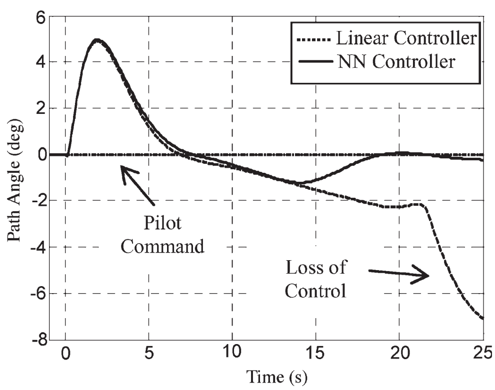


Fig. 6. Adaptive NN controller prevents loss of control that is otherwise experienced by the linearly controlled aircraft during the maneuver in Fig. 4.

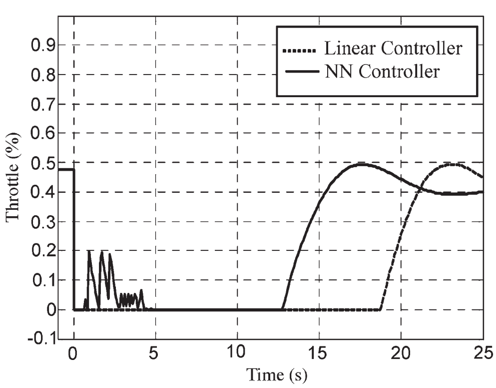


Fig. 7. Throttle input applied by the adaptive NN controller to control the maneuver in Fig. 4 significantly differs from that of the linear controller.

tested using the same command and conditions. Its performance during the maneuver improves so significantly compared to its baseline (11) that the controller is able to not only avert stall but also track the pilot command (Fig. 4) and stabilize the aircraft. Since (11) (dashed line) also specifies the NN controller's performance at $t_0$, the NN clearly learns how to control the maneuver online, ultimately applying a very different throttle input, as shown in Fig. 7.

## VII. Conclusion

A novel approach for designing ADP NN controllers is presented. The control performance and the closed-loop stability in the LPV regime are formulated as a set of design equations that are linear with respect to matrix functions of NN parameters. These design equations are adjoined to the policy improvement routine to preserve the LPV performance baseline while adapting to unmodeled dynamics and control failures. This approach is applied to the adaptive control of aircraft demonstrating that the NN controller significantly improves performance in the presence of partial elevator failure and aerodynamic-parameter variations. Also, the NN controller adapts so quickly as to avert loss of control in the presence of unmodeled near-stall dynamics.

## References

[1] S. Limanond and J. Si, "Neural-network-based control design: An LMI approach," *IEEE Trans. Neural Netw.*, vol. 9, no. 6, pp. 1422–1429, Nov. 1998.
[2] G. G. Lendaris and J. C. Neidhoefer, "Guidance in the use of adaptive critics for control," in *Learning and Approximate Dynamic Programming*, J. Si, A. Barto, and W. Powell, Eds. Hoboken, NJ: Wiley, 2004.
[3] S. N. Balakrishnan and V. Biega, "Adaptive-critic-based neural networks for aircraft optimal control," *J. Guid. Control Dyn.*, vol. 19, no. 4, pp. 893–898, 1996.
[4] G. K. Venayagamoorthy, R. G. Harley, and D. C. Wunsch, "Comparison of heuristic dynamic programming and dual heuristic programming adaptive critics for neurocontrol of a turbogenerator," *IEEE Trans. Neural Netw.*, vol. 13, no. 3, pp. 764–773, May 2002.
[5] P. J. Werbos, "Advanced forecasting methods for global crisis warning and models of intelligence," in *Gen. Syst. Yearbook*, vol. 22, 1977, pp. 25–38.
[6] M. Abu-Khalaf and F. L. Lewis, "Nearly optimal state feedback control of constrained nonlinear systems using a neural networks HJB approach," *Annu. Rev. Control*, vol. 28, no. 2, pp. 239–251, 2004.
[7] S. Ferrari and R. F. Stengel, "On-line adaptive critic flight control," *J. Guid. Control Dyn.*, vol. 27, no. 5, pp. 777–786, Sep./Oct. 2004.
[8] Y. Wang, H. Wang, and J. Jiao, "Nonlinear one-step-ahead predictive mean control of bounded stochastic systems with guaranteed stability," *Int. J. Syst. Sci.*, vol. 35, no. 2, pp. 97–107, Feb. 2004.
[9] J. A. K. Suykens, B. L. R. De Moor, and J. Vandewalle, "$\text{NL}_q$ theory: A neural control framework with global asymptotic stability criteria," *Neural Netw.*, vol. 10, no. 4, pp. 615–637, Jun. 1997.
[10] F. J. D'Amato, M. A. Rotea, A. V. Megretski, and U. T. Jönsson, "New results for analysis of systems with repeated nonlinearities," *Automatica*, vol. 37, no. 5, pp. 739–747, 2001.
[11] A. Megretski and A. Rantzer, "System analysis via integral quadratic constraints," *IEEE Trans. Autom. Control*, vol. 42, no. 6, pp. 819–830, Jun. 1997.
[12] P. Apkarian and P. Gahinet, "A convex characterization of gain-scheduled $H_{\infty}$ controllers," *IEEE Trans. Autom. Control*, vol. 40, no. 5, pp. 853–864, May 1995.
[13] [Online]. Available: http://fred.mems.duke.edu/silvia.ferrari/SMCADPsupplement08.pdf
[14] R. F. Stengel, *Optimal Control and Estimation*. New York: Dover, 1994.
[15] R. Howard, *Dynamic Programming and Markov Processes*. Cambridge, MA: MIT Press, 1960.
[16] S. Ferrari and R. F. Stengel, "Model-based adaptive critic designs," in *Learning and Approximate Dynamic Programming*, J. Si, A. Barto, and W. Powell, Eds. Hoboken, NJ: Wiley, 2004.
[17] S. Ferrari and M. Jensenius, "A constrained optimization approach to preserving prior knowledge during incremental training," *IEEE Trans. Neural Netw.*, vol. 19, no. 6, pp. 996–1009, 2008.
[18] D. P. Bertsekas, *Constrained Optimization and Lagrange Multiplier Methods*. Belmont, MA: Athena Scientific, 1996.
[19] S. Ferrari, "Algebraic and adaptive learning in neural control systems," Ph.D. dissertation, Princeton Univ., Princeton, NJ, Nov. 2002.
[20] R. Chandramohan, "Adaptive critic flight control for a general aviation aircraft: Simulations for the beech bonanza fly-by-wire testbed," M.S. thesis, Whichita State Univ., Wichita, KS, May 2007.

Author photographs and biographies not available at the time of publication.