CONTROL OF COMPLEX DYNAMIC SYSTEMS BY NEURAL NETWORKS

James C. Spall and John A. Cristion
The Johns Hopkins University
Applied Physics Laboratory
Johns Hopkins Road
Laurel, Maryland 20723-6099 U.S.A.

ABSTRACT

This paper considers the use of neural networks (NN's) in controlling a nonlinear, stochastic system with unknown process equations. The NN is used to model the resulting unknown control law. The approach here is based on using the output error of the system to train the NN controller without the need to construct a separate model (NN or other type) for the unknown process dynamics. To implement such a direct adaptive control approach, it is required that connection weights in the NN be estimated while the system is being controlled. As a result of the feedback of the unknown process dynamics, however, it is not possible to determine the gradient of the loss function for use in standard (back-propagation-type) weight estimation algorithms. Therefore, this paper considers the use of a new stochastic approximation algorithm for this weight estimation, which is based on a "simultaneous perturbation" gradient approximation that only requires the system output error. It is shown that this algorithm can greatly enhance the efficiency over more standard stochastic approximation algorithms based on finite-difference gradient approximations.

1. INTRODUCTION

One of the major problems faced by system designers is finding a means to control and regulate a system when there is uncertainty about the nature of the underlying process. Adaptive control procedures have been developed in a variety of areas for such problems (e.g., robot arm manipulation, materials handling, quality control, etc.), but are typically limited by the need to assume that the forms of the system equations are known (and usually linear) while the parameters may be unknown. In complex physical, socioeconomic, or biological systems, however, the forms of the system equations (typically nonlinear) are often unknown as well as the parameters, making it impossible to determine the control law needed in existing adaptive control procedures. This provides the motivation for considering the use of a neural network (NN) as a controller.

The approach here uses the observed system output error (actual output – target output) to train the NN-based controller without the need to identify or assume a separate model for the system. As we will show, it is not generally possible to train the NN via well-known back-propagation-type algorithms since the required gradient depends on a model for the underlying system. Thus, this paper shows how the simultaneous perturbation stochastic approximation algorithm can be used as a practical weight estimation technique in such a model-free setting. It is shown that this algorithm is much more efficient than more standard finite-difference-based algorithms.

The direct control approach here is based on using a feed-forward NN to approximate the unknown control law. The basis for this approach is the now well-known fact that any measurable function can be

This work was partially supported by the JHU/APL IRAD Program and U.S. Navy Contract N00039-91-C-0001. A more complete version of this paper is available upon request.

approximated to within any degree of accuracy by some single (or multiple) hidden-layer feed-forward NN (e.g., Funahashi [10] or Hornik, Stinchcombe, and White [12]). Our approach will proceed in one of two ways: one method will be based on making almost no assumptions about the nature of the underlying process while the other method will be based on assuming that some information (but still incomplete) is available on the form of the process equations. In the first (basically no structure information) method, the output of the NN will be used to directly approximate the elements of the control vector; in the second (partial structure information), we create a control functional that depends on unknown functions describing the system dynamics and then use a NN to approximate the unknown functions. The second method is reminiscent of the self-tuning regulator approach to adaptive control (e.g., Davis and Vinter [7, pp. 309-312]), except that we are concerned with estimating functions in a control law functional as opposed to estimating parameters in a control law with a fully known structure.

A number of others have considered using NN's for the problem of controlling uncertain nonlinear (usually deterministic) systems (see, e.g., the April 1990 and April 1992 special issues of the IEEE Control Systems Magazine, Narendra and Parthasarathy [22,23], Hunt and Sbarbaro [15], or Iiguni, Sakai, and Tokumaru [16]). Although these methods are useful under certain (fairly restrictive) conditions, they often lack the ability to control systems with minimal prior information. In particular, they require an explicit model (either NN or other parametric type) for the underlying process equations; this model is assumed to be equivalent to the "true" process equations so that it is possible to calculate the gradient needed in back-propagation-type learning algorithms. These techniques (esp. those of Narendra and Parthasarathy) also require off-line identification of the process model before implementation of the adaptive control algorithm. In contrast, our direct control approach uses the NN strictly as a model for use in the control law (no additional NN or parametric model is used for the process directly); the weights in the NN are estimated adaptively based only on the output error of the process (no prior identification is required).[1] As stated in Hoskins, Hwang, and Vagners [13], one of the major advantages of direct control techniques (versus indirect control) is that they are better able to adapt to changes in the underlying system since they are not heavily based on a prior model. Our approach addresses the shortcoming noted in Narendra and Parthasarathy [22, p. 19] that "At present, methods for directly adjusting the control parameters based on the output error (between the plant and [target] outputs) are not available."

Because it is not possible in our framework to obtain the derivatives necessary to implement standard gradient-based search techniques such as back-propagation, we will consider stochastic approximation (SA) algorithms based on approximations to the required gradient. Usually such algorithms are based on standard finite-difference approximations to the gradient (i.e., as in the multivariate Kiefer-Wolfowitz algorithm–see, e.g., Ruppert [26]). These, however, can be very costly in terms of the amount of data required, especially in high-dimensional problems such as estimating a NN weight vector (which easily has dimension of order 10^2 or 10^3). We will, therefore, consider an SA algorithm based on a "simultaneous perturbation" gradient approximation (Spall [29, 30]), which is typically much more efficient than the standard SA algorithms mentioned above in the amount of data required.

The remainder of this paper is organized as follows. Section 2 describes the two related methods for using NN's to control nonlinear systems. This section also describes why it is not possible to determine the gradient of the loss function, in contrast to the approaches of Narendra and others mentioned above where they either assume that the process dynamics is of known structure or introduce an additional NN to model the

---

[1]Chen [4] and Goldenthal and Farrell [11] have also described techniques for NN weight estimation in adaptive control when the gradient is not available, but their techniques have only been developed for particular deterministic model structures and still require considerable information about the process dynamics; in particular they require knowledge of the signs of the terms that appear in the process dynamics. Spall and Cristion [31] includes a more detailed analysis of these techniques.

dynamics. Section 3 discusses the SA approach to weight estimation using a simultaneous perturbation gradient approximation. Section 4 presents a numerical study on a nonlinear system.

2. OVERVIEW OF NEURAL NETWORK APPROACH TO CONTROL

This section describes how the NN will be implemented for the control of uncertain systems. We will describe two methods: one applies when essentially nothing is known about the dynamics of the system and the other applies when partial information on the dynamics is available. The section closes with a discussion of why the well-known "back-propagation" algorithm (or any other algorithm requiring the gradient of the loss function) can not be used for connection weight estimation in this type of general (direct) control problem, which motivates the use of stochastic approximation as discussed in Section 3.

Consider a system output vector at time $k+1$ given by

$$x_{k+1} = \phi_k(f_k(x_k, x_{k-1}, \ldots, x_{k-s}), u_k, w_k), \quad s \geq 0, \tag{2.1}$$

where $\phi_k(\cdot)$ and $f_k(\cdot)$ are generally unknown, nonlinear functions governing the dynamics of the system, u_k is the control input applied to affect the system at time $k+1$, and w_k is random noise ($f_k(\cdot)$ may also depend on an arbitrary number of previous controls and/or noise terms, but we omit this generalization for ease of notation). The most important special case of (2.1) is the Markov formulation where $f_k(\cdot) = f_k(x_k)$. Our goal is to choose the sequence of control vectors $\{u_k\}$ in a manner such that the system output is close to a sequence of target vectors $\{t_k\}$, where "close" is relative to the magnitude of the noise and the cost associated with the control.

More formally, given the measurements up to time point k we attempt to find the control that minimizes the one-step ahead loss function:

$$L_k \equiv E\,[(x_{k+1} - t_{k+1})^T A_k (x_{k+1} - t_{k+1}) + u_k^T B_k u_k \,|\, \mathcal{F}_{k-1}], \tag{2.2}$$

where A_k, B_k are positive semi-definite matrices reflecting the relative weight to put on deviations from the target and on the cost associated with larger values of u_k, and $\mathcal{F}_{k-1}$ is the σ-algebra generated by $\{x_1, x_2, \cdots, x_k, u_0, u_1, \cdots, u_{k-1}\}$. An important special case of (2.2) is the minimum variance regulator, where $A_k = I$ and $B_k = 0$.

We will consider two methods to the problem of constructing a controller u_k in the face of uncertainty about the dynamics of the system, as illustrated in Figs. 2.1a,b for the important special case where $f_k(\cdot) = f_k(x_k)$ (for the more general case as shown in (2.1), the diagrams would be modified in an obvious way). Both of the methods here correspond to <u>direct control</u> approaches as defined (without a solution) in Narendra and Parthasarathy [22] in that NN learning is based directly on the output error, $x_k - t_k$; these are in contrast to the <u>indirect control</u> methods of Narendra and Parthasarathy (and others), which are based on the off-line identification of a model of the system based on the error between the system output and model output (not system output and target) for a set of prespecified $0u_k$ inputs. In the direct approximation method of Fig. 2.1a, the output of the NN will correspond directly to the elements of the u_k vector, i.e. the inputs to the NN will be x_k and t_{k+1} and the output will be u_k. This approach is appropriate, e.g., when both $\phi_k(\cdot)$ and $f_k(\cdot)$ are unknown functions. In contrast to the direct approximation method of Fig. 2.1a, the NN in the self-tuning method of Fig. 2.1b is used to approximate the unknown dynamics $f_k(\cdot)$, which is then used in a known functional π_k to obtain u_k. Since this method requires that π_k (the functional minimizing (2.2)) be known, it requires that the overall relationship between f_k, u_k and w_k, i.e., $\phi_k(\cdot)$ in (2.1), be known. A very important type of process to which this second method can apply is an affine-nonlinear system as in Chen [4]. When prior information associated with knowledge of $\phi_k(\cdot)$ is available, the self-tuning method of Fig. 2.1b is often able to yield a superior controller. For both the direct approximation and self-tuning methods, it is required that it be known which arguments appear in $f_k(\cdot)$, i.e., for the general setting of (2.1) it is required that s be known.

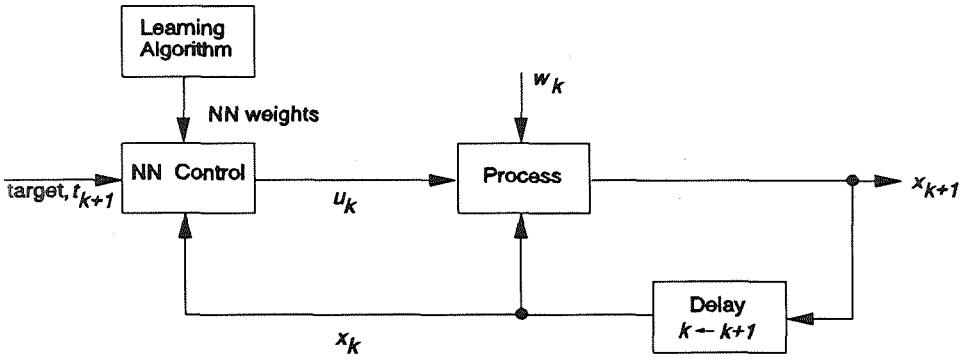

Fig. 2.1a. Control System with NN as Direct Approximator to Optimal u_k when $f_k(\cdot) = f_k(x_k)$

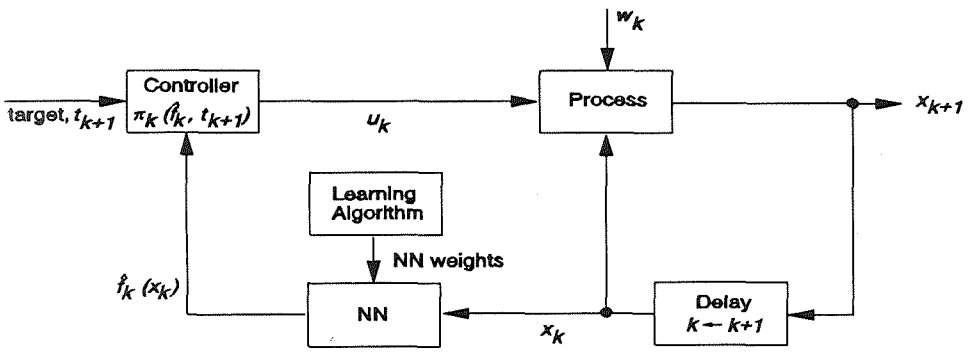

Fig. 2.1b. Self-Tuning Control System with NN as Approximator to $f_k(\cdot)$ when $f_k(\cdot) = f_k(x_k)$

The NN's to be considered here are feed-forward with at least one hidden layer of nodes (neurons) between the input and output nodes. The nodes between (but not within) adjacent layers are all connected and each connection has an associated weight, which is to be estimated from system data. It is this type of NN to which we will restrict our attention, although our method would also apply to other types of NN's (e.g., reccurent). (Since NN's have been discussed in a number of previously published control papers, we will not go into detail here on their development and theory.)

Based on the error criterion in (2.2), we wish to determine the optimal configuration for the NN. Since we assume here that the number of layers and nodes (i.e., network structure) is given, this reduces to a problem of determining the optimal values for the connection weights (determining the NN structure is an important problem in its own right, and has been considered, e.g., in Huang and Huang [14]). Letting $\theta_k \in \mathbb{R}^p$ be the vector of these weights for use in u_k, we are seeking the value of θ_k, say θ_k^*, that minimizes (2.2) given the control as found in Figs 2.1a,b. Thus for each k, we are seeking

$$\left\{\theta_k^*: \frac{\partial L_k}{\partial \theta_k} = \frac{\partial u_k{}^T}{\partial \theta_k} \cdot \frac{\partial L_k}{\partial u_k} = 0\right\}. \tag{2.3}$$

Since $f_k(\cdot)$ (and possibly $\phi_k(\cdot)$) are unknown functions, the term $\partial L_k/\partial u_k$ (and possibly $\partial u_k^T/\partial \theta_k$) in (2.3), which involves the term $\partial \phi_k/\partial u_k$, is not generally computable.[2] Thus the standard "back-propagation" algorithm (i.e., steepest descent – see, e.g., Narendra and Parthasarathy [23] or White [34]), or any other algorithm involving $\partial L_k/\partial \theta_k$, is not feasible.

To illustrate further why $\partial L_K/\partial \theta_K$ is not available in our direct control setting, consider a simple scalar-deterministic version of system (2.1). Then,

$$\frac{\partial L_k}{\partial \theta_k} = \frac{\partial (x_{k+1} - t_{k+1})^2}{\partial \theta_k} = 2(x_{k+1} - t_{k+1}) \frac{\partial \phi_k}{\partial u_k} \frac{\partial u_k}{\partial \theta_k}. \tag{2.4}$$

When neither $\phi_k(\cdot)$ nor $f_k(\cdot)$ is known (as in the direct approximation method of Fig. 2.1a), then neither of the derivatives on the right-hand side of (2.4) will be known. When $f_k(\cdot)$ is unknown and $\phi_k(\cdot)$ and $u_k = \pi_k(\cdot)$ are known (as in the self-tuning method of Fig. 2.1b), then $\partial u_k/\partial \theta_k$ will be known but $\partial \phi_k/\partial u_k$ will, in general, still be unknown since it will depend on $f_k(\cdot)$.[3] Thus we see that in either of the direct control settings in Fig. 2.1a,b, $\partial L_k/\partial \theta_k$ is not generally available. The same principles apply in the more general multivariate stochastic version of model (2.1).

Because back-propagation-type algorithms are not generally feasible in the direct control setting here, we consider a stochastic approximation (SA) algorithm of the form

$$\hat{\theta}_k = \hat{\theta}_{k-1} - a_k \, (\textit{gradient approx.})_k \tag{2.5}$$

to estimate $\{\theta_k^*\}$, where $\hat{\theta}_k$ denotes the estimate at the given iteration, $\{a_k\}$ is a scalar gain sequence satisfying certain regularity conditions, and the gradient approximation is such that it does not require knowledge of $f_k(\cdot)$ (and $\phi_k(\cdot)$, if appropriate). The next section is devoted to describing in more detail the SA approach to this problem.

3. WEIGHT ESTIMATION BY SIMULTANEOUS PERTURBATION STOCHASTIC APPROXIMATION

Recall that we are seeking the NN weight vector at each time point that minimizes (2.2), i.e., we are seeking the minimizing θ_k, θ_k^*, such that

$$g_k(\theta_k^*) \equiv \left. \frac{\partial L_k}{\partial \theta_k} \right|_{\theta_k^*} = 0,$$

where θ_k is for use in the control u_k. Recall also that since back-propagation (or other derivative-based) algorithms are not applicable, we will consider an SA-based approach. This subsection describes the simultaneous perturbation SA (SPSA) approach to this problem and mentions how this approach contrasts with the more standard finite-difference SA (FDSA) approach of Kiefer-Wolfowitz. Spall [30] gives a detailed analysis of the SPSA approach to optimization. It is shown that the SPSA algorithm can achieve the same level of

---

[2] This contrasts with the "open loop" identification problems in, e.g., Narendra and Parthasarathy [22, Sect. 5], where in estimating the connection weights no unknown functions appear in the gradient. This also contrasts with implementations of so-called indirect feedback controllers (e.g., Narendra and Parthasarathy [22, Sect. 6]) where a NN is used to model the unknown system dynamics and the identification and adaptive control is performed as if the NN model was identical in structure to the true system dynamics.

[3] One special case where $\partial L_k/\partial \theta_k$ <u>can</u> be computed is in the self-tuning setting of Fig. 2.1b where $u_k(\cdot)$ is known to enter $\phi_k(\cdot)$ additively (since $\partial \phi_k/\partial u_k$ then does not depend on $f_k(\cdot)$). Of course, in the more general setting of direct approximation control (Fig. 2.1a) $\partial L_k/\partial \theta_k$ would still be unavailable.

asymptotic accuracy as FDSA with only $1/p$ the number of system measurements. This is of particular interest in neural network problems since p can easily be on the order of 10^2 or 10^3.

In line with (2.5), the SPSA algorithm has the form

$$\hat{\theta}_k = \hat{\theta}_{k-1} - a_k \hat{g}_k(\hat{\theta}_{k-1}) \tag{3.1a}$$

where $\hat{g}_k(\hat{\theta}_{k-1})$ is the simultaneous perturbation approximation to $g_k(\hat{\theta}_{k-1})$. In particular the ℓth component of $\hat{g}_k(\hat{\theta}_{k-1})$, $\ell = 1,2,\ldots,p$, is given by

$$\hat{g}_{k\ell}(\hat{\theta}_{k-1}) = \frac{\hat{L}_k^{(+)} - \hat{L}_k^{(-)}}{2c_k\Delta_{k\ell}}, \tag{3.1b}$$

where

- $\hat{L}_k^{(\pm)} = (x_{k+1}^{(\pm)} - t_{k+1})^T A_k (x_{k+1}^{(\pm)} - t_{k+1}) + u_k^{(\pm)T} B_k u_k^{(\pm)}$,
- $u_k^{(\pm)} = u_k(x_k,\ldots,x_{k-s},t_{k+1},\hat{\theta}_{k-1} \pm c_k\Delta_k)$, i.e., a control based on a NN with weight vector $\theta_k = \hat{\theta}_{k-1} + c_k\Delta_k$ or $\theta_k = \hat{\theta}_{k-1} - c_k\Delta_k$,
- $x_{k+1}^{(\pm)}$ is system output based on $u_k^{(\pm)}$, $\Delta_k = (\Delta_{k1},\Delta_{k2},\ldots,\Delta_{kp})^T$, with the $\{\Delta_{ki}\}$ independent, symmetrically distributed (about 0) random variables $\forall k,i$, identically distributed at each k, with $E(\Delta_{ki}^{-2})$ uniformly bounded $\forall k,i$,
- $\{c_k\}$ is a sequence of positive numbers satisfying certain regularity conditions.

The key fact to observe is that at any iteration only two measurements are needed (i.e., the numerators are the same for all p components). This is in contrast to the standard FDSA approach where $2p$ measurements are needed to construct the approximation to $g_k(\cdot)$ (i.e., for the ℓth component of the gradient approximation, the quantity Δ_k is replaced by a vector with a positive constant in the ℓth place and zeroes elsewhere; see, e.g., Ruppert [26]). A variation on the form in (3.1b) is to average several gradient approximations, with each vector in the average being based on a new (independent) value of Δ_k and a corresponding new pair of measurements; this will often enhance the performance of the algorithm as illustrated in Section 4. A further variation on (3.1b) is to smooth across time by a weighted average of the previous and current gradient estimates (analogous to the "momentum" approach in back-propagation); such smoothing can often improve the performance of the algorithm (see Spall and Cristion [32] for a thorough discussion of smoothing in SPSA-based direct adaptive control).

The complete version of this paper gives a much fuller account of the theory behind SPSA together with some of the practical issues associated with its implementation in adaptive control.

4. EMPIRICAL STUDY

4.1 Preliminaries

This section presents the results of our study on a stochastic generalization of a model in Narendra and Parthasarathy [22] (N & P hereafter in this section). We will compare the SPSA and FDSA weight estimation algorithms.

The study here is based on $A_k = I$ and $B_k = 0$ in the loss function (2.2) (i.e., a minimum variance regulator). The performance of the various techniques will be evaluated by comparing the root-mean-square (RMS) tracking error as normalized by the dimension of x_k, i.e. RMS at time k is

$[(x_k - t_k)^T(x_k - t_k)/\dim(x_k)]^{1/2}$. The (feedforward) NN's considered here have an input layer, two hidden layers, and an output layer, as in N & P and Chen [4]. The hidden layer nodes are hyperbolic tangent functions (i.e., $(e^x - e^{-x})/(e^x + e^{-x})$ for input x) while the output nodes are linear functions (simply x). Each node takes as an input (x) the weighted sum of outputs of all nodes in the previous layer plus a bias weight not connected to the rest of the network (hence an $N_{4,20,10,2}$ network, in the notation of N & P, has 100 + 210 + 22 = 332 weights to be estimated). For the weight estimation, we will consider different forms of the SPSA algorithm, denoted SPSA-q, where q denotes the number of individual gradient approximations of the form (3.1b) that are averaged to form $\hat{g}_k(\cdot)$ (hence an SPSA-q algorithm uses $2q$ measurements to form $\hat{g}_k(\cdot)$). For the SPSA algorithms we take the perturbations Δ_{ki} to be Bernoulli ± 1 distributed, which satisfies the relevant regularity conditions of Section 3.

4.2 Results of Numerical Study

The model we consider is a generalization of the two-dimensional model with additive control given in N & P to include additive (independent) noise, i.e.,

$$x_{k+1} = f(x_k) + u_k + w_k, \quad w_k \sim N(0,\sigma^2 I), \tag{4.1}$$

where, as in eqn. (18) of N & P, the data are generated according to

$$f(x_k) = \frac{1}{1 + x_{k2}^2} \begin{bmatrix} x_{k1} \\ x_{k1}x_{k2} \end{bmatrix}$$

with x_{ki} the ith $(i = 1,2)$ component of x_k. Analogous to N & P we take the two-dimensional target sequence to be generated by the deterministic difference equation

$$t_{k+1} = \begin{pmatrix} .6 & .2 \\ .1 & -.8 \end{pmatrix} t_k + \begin{pmatrix} \sin(2\pi k/25) \\ \cos(2\pi k/25) \end{pmatrix}, \quad t_0 = 0.$$

Because (4.1) is an additive control model, we will only consider the DA method in this study (see footnote 3). To implement DA the analyst is assumed to know that $s=0$ (i.e., that this is a Markov-type model) and that $\dim u_k = 2$. As with N & P we used NN's with two hidden layers, one of 20 nodes and one of 10 nodes (so an $N_{4,20,10,2}$ network was used for the controller). The indicated RMS errors throughout this study are normalized for the two-dimensional setting as discussed in Subsection 4.1; therefore, since $\mathrm{cov}(w_k) = \sigma^2 I$, we know that long-run RMS can at best equal σ.

Fig. 4.1 presents the main results for our study of the model in (4.1). The RMS curves in the figures are based on the sample mean of four independent runs with different initial weights $\hat{\theta}_0$, where the elements of $\hat{\theta}_0$ were generated randomly from a uniform (-.1, .1) distribution. To effect a fair comparison of the algorithms the same four sets of initial weight vectors were used for the three different curves. To further smooth the resulting error curves and to show typical performance (not just case-dependent variation), we applied the MATLAB low-pass interpolating function INTERP to the error values based on the average of four runs. The curves shown in the figures are based on this combination of across-realization averaging and across-iteration interpolation. Each of the curves was generated by using SA gains of the form $a_k = A/k^{.7501}$ and $c_k = C/k^{.25}$ with $A, C>0$ (the exponents were chosen to satisfy standard SA conditions and to afford a_k the effectively slowest rate of decay consistent with these conditions; a slow decay rate tended to accelerate the rate of decrease in RMS error). For each curve we attempted to tune A and C to maximize the rate of convergence of $\hat{\theta}_k$ (as would typically be done in practice); the values satisfied $.037 \le A \le .12$ and $.20 \le C \le .25$. The value x_0 was set to $(1.5, 1.5)^T$ for all studies, so the initial RMS error is 1.5.

Fig. 4.1 shows that both the SPSA and FDSA algorithms yield controllers with decreasing RMS tracking error over time. The RMS error curves for both algorithms show the characteristic shape of first order (steepest-descent-type) algorithms in that there is a sharp initial decline followed by slow decline. We see that the long-run performance of SPSA-4 is slightly better than that of FDSA, with SPSA-4 and FDSA achieving terminal RMS errors of .32 and .33 respectively (vs. the theoretical limit of .25); for the SPSA-1 algorithm the terminal error was .47. The critical observation to make here is that the SPSA algorithms achieved their performance with a large savings in data: each iteration of SPSA-1 and SPSA-4 used only three measurements and nine measurements, respectively, while each iteration of FDSA used 665 measurements (these measurement counts include the one operational measurement for each iteration in addition to the measurements generated for purposes of constructing the gradient approximation). Hence Fig. 4.1 illustrates that SPSA-4 yields a slightly lower level of long-run tracking error then the standard FDSA algorithm with a 74-fold savings in system measurements. The data savings seen in Fig. 4.1 is typical of that for a number of other studies involving SPSA and FDSA that we have conducted on model (4.1) as well as on other nonlinear models; in fact even greater data savings are typical with more complex NN's (as might be needed in higher-dimensional systems or in systems where u_k is not simply additive).

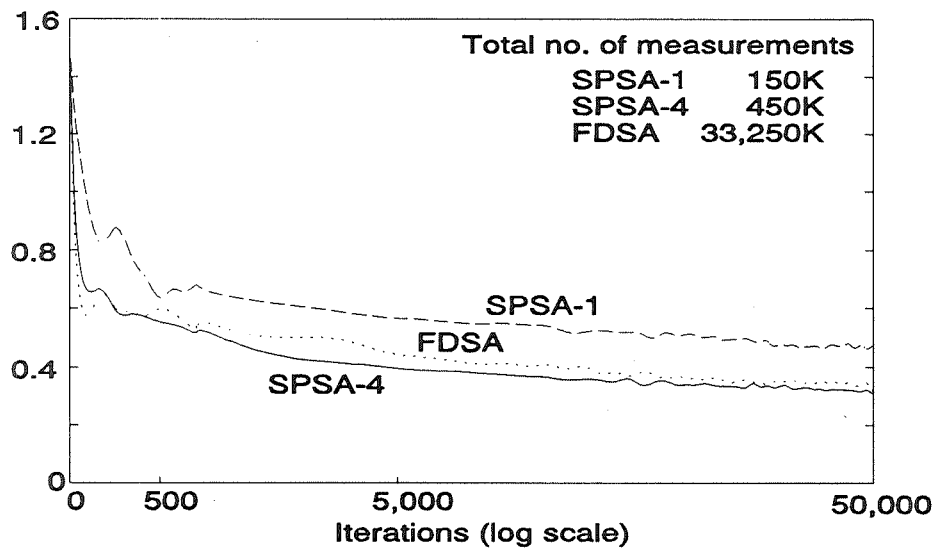

Fig. 4.1 RMS Error for DA Controller with SPSA and FDSA Algorithms in Additive Control Model with σ = .25

Our other numerical study for (4.2) illustrates the relative performance of SPSA and FDSA in a deterministic (σ=0) system (to complement the stochastic comparison in the study for (4.1)). As with (4.1), we used the DA control method. The mean and terminal RMS errors for SPSA-1 were, respectively, .12 and .087 versus .14 and .103 for FDSA. Thus SPSA outperforms FDSA with less than 1/220 the number of system measurements.

REFERENCES

(The list below includes all items cited in the full version of this paper, which is available from the authors upon request.)

[1] Bayard, D.S. [1991], "A Forward Method for Optimal Stochastic Nonlinear and Adaptive Control," IEEE Trans. Auto. Control, vol. 36, pp. 1046-1053.

[2] Benveniste, A. and Ruget, G. [1982], "A Measure of the Tracking Capability of Recursive Stochastic Algorithms with Constant Gains," IEEE Trans. Auto. Control, vol. AC-27, pp. 639-649.

[3] Boguslavskij, I.A [1988], Filtering and Control, Optimization Software Pubs. Div., New York.

[4] Chen, F.C. [1990], "Back-Propagation Neural Networks for Nonlinear Self-Tuning Adaptive Control," IEEE Control Syst. Mag., April, pp. 44-48.

[5] Chin, D.C. [1992], "A More Efficient Global Optimization Algorithm Based on Styblinski and Tang (1990)," Neural Nets., submitted.

[6] Chung, K.L. [1974], A Course in Probability Theory, Academic, New York.

[7] Davis, M.H.A and Vinter, R.B. [1985], Stochastic Modeling and Control, Chapman and Hall, New York.

[8] Evans, S.N. and Weber, N.C. [1986], "On the Almost Sure Convergence of a General Stochastic Approximation Procedure," Bull. Austral. Math. Soc., vol. 34, pp. 335-342.

[9] Fabian, V. [1971], "Stochastic Approximation," in Optimizing Methods in Statistics (J.J. Rustagi, ed.), Academic, New York, pp. 439-470.

[10] Funahashi, K.I. [1989], "On the Approximate Realization of Continuous Mappings by Neural Networks," Neural Nets., vol. 2, pp. 183-192.

[11] Goldenthal, W. and Farrell, J. [1990], "Application of Neural Networks to Automatic Control," Proc. AIAA Guidance, Navigation and Control Conf., Part 2, pp. 1108-1112.

[12] Hornik, K., Stinchcombe, M. and White, H. [1989], "Multilayer Feedforward Networks are Universal Approximators," Neural Nets., vol. 2, pp. 359-366.

[13] Hoskins, D.A., Hwang, J.N., and Vagners, J. [1992], "Iterative Inversion of Neural Networks and its Applications to Adaptive Control," IEEE Trans. Neural Nets., vol. 3, pp. 292-301.

[14] Huang, S.-C. and Huang, Y.-F. [1991], "Bounds on the Number of Hidden Neurons in Multilayer Perceptrons," IEEE Trans. Neural Nets., vol. 2, pp. 47-55.

[15] Hunt, K.J. and Sbarbaro, P. [1991], "Neural Networks for Nonlinear Model Control," IEEE Proc.-D, vol. 138, pp. 431-438.

[16] Iiguni, Y, Sakai, H. and Tokumaru, H. [1991], "A Nonlinear Regulator Design in the Presence of System Uncertainties Using Multilayered Neural Networks," IEEE Trans. Neural Nets., vol. 2. pp. 410-417.

[17] Isidori, A. and Byrnes, C.I. [1990], "Output Regulation of Nonlinear Systems," IEEE Trans. Auto. Control, vol. 35, pp. 131-140.

[18] Kuan, C.-M. and Hornik, K. [1991], "Convergence of Learning Algorithms with Constant Learning Rates," IEEE Trans. Neural Nets., vol. 2, pp. 484-489.

[19] Kushner, H. J. and Huang, H. [1981], "Asymptotic Properties of Stochastic Approximations with Constant Coefficients," SIAM J. Control Optimiz., vol. 19, pp. 87-105.

[20] Macchi, O. and Eweda, E. [1983], "Second Order Convergence Analysis of Stochastic Adaptive Linear Filtering," IEEE Trans. Auto. Control, vol. 28, pp. 76-85.

[21] Moden, P.E. and Soderstrom, T. [1982], "Stationary Performance of Linear Stochastic Systems Under Single Step Optimal Control," IEEE Trans. Auto. Control, vol. AC-27, pp. 214-216.

[22] Narendra, K.S. and Parthasarathy, K. [1990], "Identification and Control of Dynamical Systems Using Neural Networks," IEEE Trans. Neural Nets. vol. 1, pp. 4-26.

[23] Narendra, K.S. and Parthasarathy, K. [1991], "Gradient Methods for the Optimization of Dynamic Systems Containing Neural Networks," IEEE Trans. Neural Nets., vol. 2, pp. 252-262.

[24] Nijmeijer, H. and van der Schaft, A. J. [1990], Nonlinear Dynamical Control Systems, Springer-Verlag, New York.

[25] Psaltis, D., Athanasios, S., and Yamamura, A.A. [1988], "A Multilayered Neural Network Controller," IEEE Control Syst. Mag., vol. 8, April, pp. 17-21.

[26] Ruppert, D. [1983], "Kiefer-Wolfowitz Procedure," Encyclopedia of Statistical Science, vol. 4 (S. Kotz and N.L. Johnson, eds.), pp. 379-381, Wiley, New York.

[27] Ruppert, D. [1985], "A Newton-Raphson Version of Multivariate Robbins-Monro Procedure," Ann. Stat., vol. 13, pp. 236-245.

[28] Saridis, G.N. [1977], Self-Organizing Control of Stochastic Systems, Marcel Dekker, New York.

[29] Spall, J.C. [1988], "A Stochastic Approximation Algorithm for Large-Dimensional Systems in the Kiefer-Wolfowitz Setting," Proc. IEEE Conf. Dec. Control, pp. 1544-1548.

[30] Spall, J.C. [1992], "Multivariate Stochastic Approximation Using a Simultaneous Perturbation Gradient Approximation," IEEE Trans. Auto. Control, vol. 37, pp. 332-341.

[31] Spall, J.C. and Cristion, J.A. [1991], "Neural Networks for Control of Uncertain Systems," Proc. Test Technology Symp. IV (sponsored by U.S. Army Test and Evaluation Command), pp. 575-588.

[32] Spall, J.C. and Cristion, J.A. [1992], "Nonlinear Adaptive Control Using Neural Networks: Estimation Based on a Smoothed Form of Simultaneous Perturbation Gradient Approximation," Statistica Sinica, submitted.