Statistica Sinica 4(1994), 1-27

NONLINEAR ADAPTIVE CONTROL USING NEURAL NETWORKS: ESTIMATION WITH A SMOOTHED FORM OF SIMULTANEOUS PERTURBATION GRADIENT APPROXIMATION

James C. Spall and John A. Cristion

The Johns Hopkins University

Abstract: Neural networks (NNs) have recently attracted much attention in the mathematical modeling community. One of the most promising areas for the use of NNs is in the control of complex (multivariate) dynamic systems when the nonlinear equations governing the system are not known. In this paper, a NN is used to model the resulting unknown control law *without* the need to construct a separate model (NN or other type) for the unknown process dynamics. This is a challenging statistical problem in that the weights (parameters) of the NN are estimated concurrently with controlling the system. The weight estimation uses a form of stochastic approximation that relies on an approximation to the gradient of the underlying loss function. The implementation here uses a simple smoothing operation when constructing the gradient approximation: namely, the gradient approximation at any iteration is formed as a combination of the previous approximation and a new simultaneous perturbation gradient estimate. This paper shows that this smoothing idea is often useful in improving the ability of the NN-based controller to have the system perform in the desired way. Aside from presenting the smoothing idea, this paper includes brief introductions to the fields of nonlinear adaptive control, neural networks, and stochastic approximation.

Key words and phrases: Nonlinear control systems, stochastic approximation, neural network learning, gradient estimation.

1. Introduction

1.1. Scope of paper

This is an interdisciplinary paper spanning the fields of control, statistics, neural networks, and optimization. The goals of this paper are four-fold: (1) to consider the problem of developing adaptive controllers for general dynamic systems with *unknown* (usually nonlinear) governing equations — which has been an essentially unsolved problem — and develop a solution for an important class of such problems, (2) to briefly review the growing area of neural networks (NNs) and show how a NN used as a function approximator can be employed in the

above-mentioned adaptive control problem, (3) to review stochastic approximation (SA) as a general optimization technique in the presence of noisy data and show how the simultaneous perturbation SA technique can be used in the problem of "training" the NN for use in the control problem, and (4) to introduce a modification to simultaneous perturbation SA that is based on smoothing gradient approximations across iterations and illustrate this modification on the NN-based control problem. Because of the interdisciplinary nature of this paper, the reader may find certain aspects to be of greater interest than others. In fact, many of the ideas discussed here (e.g., the smoothed SA algorithm) are generic and could apply in areas other than NN-based adaptive control.

1.2. Adaptive control and neural networks

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 (i.e., those that can learn and adapt over time) have been developed in a variety of areas for such problems. Examples of applications in areas such as robot arm manipulation, aircraft control, macroeconomic policy making, and biological systems regulation are given in 1986 and 1991 IEEE Press books (see list of references) and journals such as the *IEEE Transactions on Automatic Control, Automatica*, the *Journal of Economic Dynamics and Control*, and the *International Journal of Adaptive Control and Signal Processing* (to name a few); an excellent brief introduction to adaptive control is given by M. Gupta on pp. xv-xxii of the above-mentioned 1986 IEEE Press book. Existing adaptive control procedures are typically limited by the need to assume that the *forms* of the system equations are known (and usually linear) while the parameters within the assumed equations are estimated. Further, existing procedures require that the assumed equation forms do not change as the system evolves in time. In complex physical, socioeconomic, or biological systems, however, the forms of the system equations (typically nonlinear) are often unknown and may also be changing in time, making it impossible to determine the control law in existing adaptive control procedures. This provides the motivation for considering the use of artificial neural networks.

Artificial neural networks, or more commonly just neural networks, have recently attracted much attention for their potential to address a number of difficult problems in modeling. One of the areas receiving a significant portion of the attention is the use of NNs for controlling and regulating nonlinear dynamic systems. Traditionally, developing controllers for nonlinear systems has been extremely difficult, even in deterministic settings where the equations governing the system dynamics are fully known. (See Pao, Phillips, and Sobajic (1992) for a brief review of the nonlinear control problem.) NNs, however, offer the potential

for addressing control problems even broader than this, including the control of stochastic systems with unknown nonlinear dynamics.

The approach here uses the observed system output error (i.e., actual output – target output) to train the NN-based controller *without* the need to estimate or assume a separate model for the equations governing the dynamics of the system. As we will show, it is then not generally possible to train the NN (i.e., estimate the connection weights) via well-known back-propagation (steepest descent)-type algorithms since the required gradient depends on a model for the underlying system. Thus, this paper shows how the simultaneous perturbation SA algorithm (Spall (1992)) can be used as a practical weight estimation technique in such a model-free setting.

The control approach here is based on using a NN to approximate the unknown control law. The basis for this approach is the now well-known fact that any measurable function can be approximated to within any degree of accuracy by some single (or multiple) hidden-layer feed-forward NN (e.g., Funahashi (1989) or Hornik, Stinchcombe, and White (1989)). 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, and 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 law 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 (1985, pp. 309-312)), except that we are concerned with estimating unknown functions, as opposed to unknown parameters, in a control law.

A number of others have considered using NNs 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 (1990), Tulunay (1991), Hunt and Sbarbaro (1991), or Pao, Phillips, and Sobajic (1992)). 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 such as linear or nonlinear ARMA) 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 typically require off-line identification of the process model *before* implementation of the adaptive control algorithm.

In contrast to the above, our direct approach applies the NN strictly as a model for use in the control law (no additional NN or parametric model is used for the process); the weights in the NN are estimated adaptively based only on the output error of the process as it operates in "closed-loop" mode (no prior open-loop model estimation is required). Thus the approach here addresses the shortcoming noted in Narendra and Parthasarathy (1990, 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." One of the major advantages of such a direct approach is that it will tend to better adapt to changes in the underlying system, since it is not based on a prior model for the system; further, it will tend to be more robust to extreme values of the control even when the underlying system does not change (i.e., the prior-model-based approach may perform poorly for closed-loop controls outside of the range of open-loop controls used in the off-line estimation step).

1.3. The optimization problem

There is a critical difference between the optimization problem for control and that associated with typical "model fitting" (such as regression). In control problems, the output being produced by the model (a NN here) is a control *input* to the system. The system output is the quantity of ultimate interest, rather than the value of the control (the model output). In contrast, in standard model fitting the output produced by the model *is* of ultimate interest and is compared with actual system output for purposes of fitting (estimating) the model parameters. We now elaborate a little on the optimization problem here.

Our goal is to find optimal values of the NN weight parameters, i.e., those parameters such that the NN-based control leads to optimal system output. Because it is not possible in our framework to obtain the derivatives necessary to implement standard gradient-based search techniques such as back-propagation (this is discussed in more detail in Subsection 3.2), 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 (for examples of such algorithms in control, see Saridis (1977, pp. 375-376) or Bayard (1991)). The finite-difference approach, 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 (1992)), which is typically much more efficient than the standard SA algorithms mentioned above in the amount of data required. Spall and Cristion (S&C) (1992, 1993) also consider the use of simultaneous perturbation SA (SPSA) in general control problems. To

make this paper self-contained, some of the discussion here will be a review of material in S&C (1992, 1993).

The main new result of this paper is to show that the SPSA-based approach of S&C (1992, 1993) can often be enhanced by a certain across-iteration smoothing for the required gradient approximation. The SA algorithm is used to estimate the NN weights while the system is being controlled. To do this estimation, an approximation to the gradient of the loss function is formed at every iteration. In S&C (1992, 1993) this gradient approximation is formed independently at every iteration, as is typically done in most SA techniques. In contrast here, we consider a smoothing technique where the gradient approximation at every iteration is formed as a convex combination of the previous gradient approximation and a new approximation based on the SP technique. This smoothing is in the spirit of the conjugate gradient type SA algorithm in, e.g., Ruszczynski and Syski (1983). It is shown here that this smoothed implementation of SPSA can often offer improved performance over the standard unsmoothed algorithm with no additional cost to the user, either in terms of the number of measurements needed or in terms of computational burden.

The remainder of this paper is organized as follows. Section 2 presents background information on NNs and SA. Section 3 presents an overview of our approach to control and describes why it is not possible to determine the gradient of the loss function for use in a standard back-propagation algorithm (in contrast to the approaches of Narendra and others where they either assume that the process dynamics is of known structure or introduce an additional NN to model the dynamics). Section 4 discusses the SA approach to weight estimation using the smoothed implementation of the simultaneous perturbation gradient approximation and presents a theoretical result on the convergence of the weight estimate. Section 5 presents a numerical study on two different nonlinear systems taken from the control and statistics literature and Section 6 offers some concluding remarks.

2. Background Material: Brief Reviews of Neural Networks and Stochastic Approximation

This section is composed of two subsections. Subsection 2.1 presents a brief introduction to certain essential neural network concepts and Subsection 2.2 does the same for stochastic approximation.

2.1. Neural networks

Feed-forward artificial neural networks (simply neural networks or NNs here) are mathematical models that attempt to achieve some predetermined goal via

the adjustment of weighted interconnections between simple computational elements (referred to here as nodes). Our interest in NNs in this paper is as an approximator for an unknown nonlinear function. The NN model was inspired by its biological counterpart, the human brain. Just as humans attempt to process all information received by their sensors (e.g., eyes, ears, etc.) to determine appropriate actions (e.g., talk, move, etc.), NNs attempt to use input information to derive useful outputs. Of course, as with humans, there must be a learning process that improves the performance of the NN over time. It is important that this learning process produce a NN model that not only is able to reproduce a set of known input-output relations, but is also able to produce reasonable outputs for new input data.

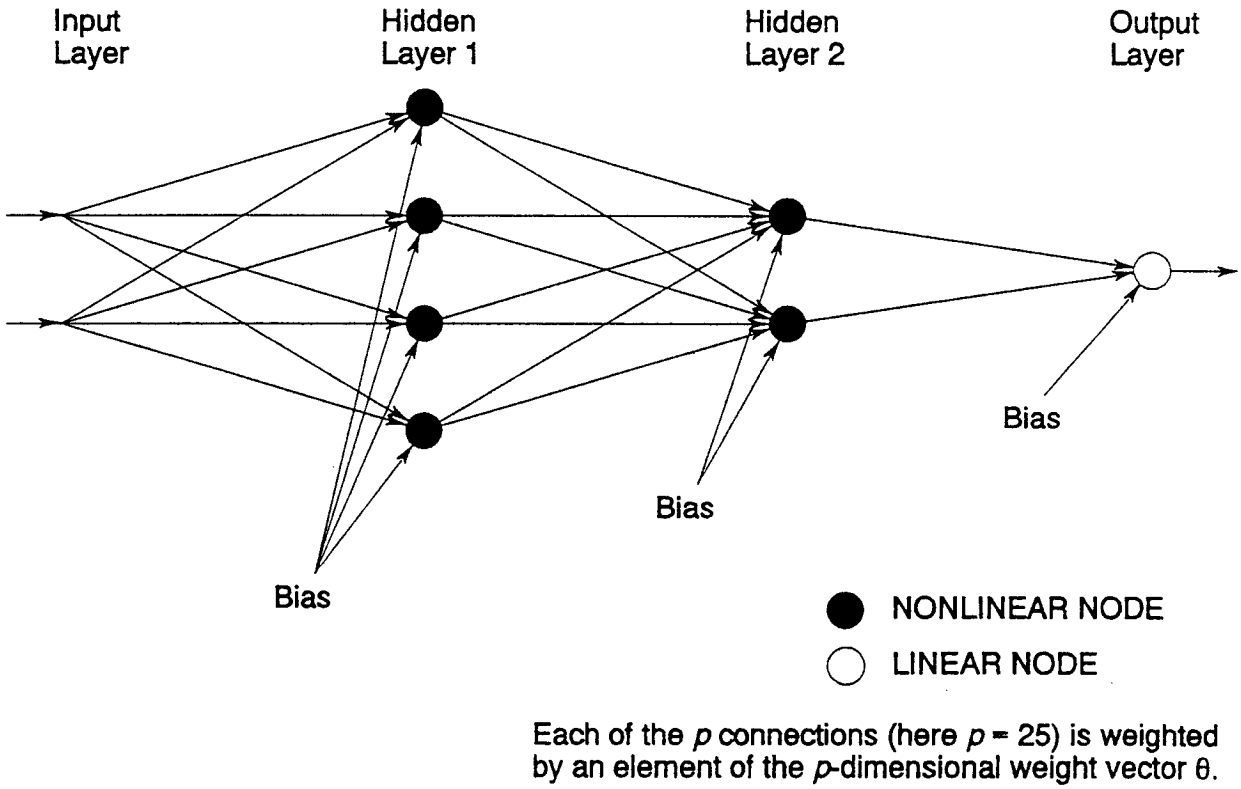

Figure 2.1. Example of simple neural network ($N_{2,4,2,1}$)

NNs typically consist of many nodes arranged in layers (see Fig. 2.1 for a simplified example). Each node in a layer is connected to every node in the next layer; other configurations are certainly possible, but we will only consider the standard feed-forward NN of the generic form shown in Fig. 2.1. There are three types of layers: input layer, output layer, and one or more "hidden" layers, where the hidden layers are between the input and output layers and are necessary for learning general nonlinear input-output relationships. The shorthand notation used here (as in Narendra and Parthasarathy (1990)) to denote a NN with a

certain configuration is $N_{(\cdot)}$, where $(\cdot)$ is an ordered listing of the number of nodes in each layer beginning with the input layer; in this notation the NN of Fig. 2.1 is $N_{2,4,2,1}$. Each node generates an output value that is based on a weighted sum of its input signals and a bounded nonlinear (e.g., output $=\tanh(\cdot)$) or a linear (e.g., output $=$ input) operation. Bounded nonlinear nodes are typically used throughout the NN with the exception of the output layer, where linear nodes are sometimes used to allow for outputs with unknown ranges (this is the case in Fig. 2.1 as well as in the examples of Section 5). The input signals to a node are provided from previous layers (e.g., in Fig. 2.1 the input signals for each node in Hidden Layer 2 are provided by Hidden Layer 1). They are processed by the node and passed on to the next layer. This process is repeated for each node in each layer. Each node takes as an input the weighted (by elements of the weight vector θ) sum of output signals from the previous layer (so, e.g., the input to a node in Hidden Layer 2 of Fig. 2.1 is a weighted sum of the four output signals from Hidden Layer 1 and a bias weight, described below). The value of θ is determined through some learning process (typically by the so-called "back-propagation" algorithm, although this technique can not be used in the control problem here, as described in Section 3 below). Often it is useful to include an additional weighted bias input signal (a constant, data-independent input signal). These bias inputs are also elements of the weight vector θ and must be derived by the same learning process. A more detailed general discussion of NNs and their properties may be found in, e.g., Wasserman (1989). NNs used in practice typically have many more nodes than shown in Fig. 2.1 in order to provide the richness needed to capture a complex nonlinear relationship.

2.2. Stochastic approximation

Stochastic approximation refers to a general set of recursive algorithms for finding minima (or roots) of functions in the presence of noisy observations. In particular, suppose we wish to minimize some differentiable loss function $L(\theta)$, $L : R^p \rightarrow R^1$. Thus we are searching for a minimizing point θ^* satisfying the gradient equation:

$$g(\theta) \equiv \frac{\partial L}{\partial \theta} = 0.$$

There are two main classes of SA algorithms for solving this equation; these are outlined in the pioneering papers of Robbins-Monro (1951) and Kiefer-Wolfowitz (1952). The Robbins-Monro class assumes that (noisy) observations are available on $g(\theta)$ directly, i.e., data are obtained in the form $g(\theta)+noise$ for various levels of θ. Kiefer-Wolfowitz techniques, on the other hand, only require observations on the loss function itself, i.e., data are obtained in the form $L(\theta)+noise$ for various levels of θ. In either class, the basic form for the recursive algorithm to find θ^*

is

$$\hat{\theta}_k = \hat{\theta}_{k-1} - a_k\,(approx.\ of\ \ g(\hat{\theta}_{k-1})), \tag{2.1}$$

where a_k is a positive (scalar) gain coefficient. Under appropriate regularity conditions on $\{a_k\}$ and the gradient approximation, $\hat{\theta}_k$ converges to θ^* (in some probabilistic sense) as $k \to \infty$ (see, e.g., Blum (1954)). The essential difference between the Robbins-Monro and Kiefer-Wolfowitz classes of algorithms is in the information used to form the gradient approximation in (2.1). As mentioned in Section 1 (and shown in more detail in Section 3), observations of $g(\theta)$ are not available in our control problem; hence the Robbins-Monro class (which includes NN back-propagation — see White (1989)) is not applicable.

Within the Kiefer-Wolfowitz class, the standard approach to constructing the gradient approximation in (2.1) is to use a finite-difference method (Ruppert (1983)). In particular, at each iteration of (2.1), we form an approximation based on observations of $L(\cdot)$ at design levels of θ representing a positive and negative perturbation of each component within $\hat{\theta}_{k-1}$ taken one at a time. Thus, since θ is p-dimensional, we need a total of $2p$ observations of $L(\cdot)$ per gradient estimate.

An alternate approach to approximating the gradient in (2.1) is the simultaneous perturbation (SP) method of Spall (1988, 1992). The SP gradient estimate is formed from *two* observations of $L(\cdot)$ *independent* of the dimension p (vs. $2p$ observations for the finite-difference method). In particular, let $\Delta_k = (\Delta_{k1}, \Delta_{k2}, \ldots, \Delta_{kp})^T$ be a vector of independent mean 0 random variables satisfying certain regularity conditions and let c_k be a positive scalar. The two required observations of $L(\cdot)$, say $\hat{L}_k^{(\pm)}$, are obtained at design levels $\hat{\theta}_{k-1} \pm c_k\Delta_k$. Then the SP approximation to $g_k(\hat{\theta}_{k-1})$ is

$$\hat{g}_k(\hat{\theta}_{k-1}) = \begin{bmatrix} \dfrac{\hat{L}_k^{(+)} - \hat{L}_k^{(-)}}{2c_k\Delta_{k1}} \\ \vdots \\ \dfrac{\hat{L}_k^{(+)} - \hat{L}_k^{(-)}}{2c_k\Delta_{kp}} \end{bmatrix}. \tag{2.2}$$[1]

Note that the numerators in the p components of $\hat{g}_k(\hat{\theta}_{k-1})$ are identical (in contrast to finite-difference). Spall (1992) gives a detailed discussion of the regularity conditions for the SPSA algorithm of using (2.2) in (2.1) (one interesting condition is that certain inverse moments of the Δ_{ki} must exist, which *prevents* them from having a uniform or normal distribution). It is shown in Spall (1992) that $\hat{g}_k(\hat{\theta}_{k-1})$ is an unbiased estimator of $g(\hat{\theta}_{k-1})$ to within $O(c_k^2)$, $c_k \to 0$. Further,

[1]Note that the indexing in (2.1) and (2.2) differs slightly from that in Spall (1992); this was done to be consistent with standard adaptive control indexing.

it is shown that the SPSA algorithm has the usual almost sure (a.s.) convergence and asymptotic normality properties of standard finite-difference (FDSA) SA algorithms, but that the asymptotic normality result indicates that SPSA can achieve the same level of asymptotic accuracy in estimating θ^* as FDSA with only $1/p$ the number of measurements of $L(\cdot)$. This is of particular interest in neural network problems since p can easily be on the order of 10^2 or 10^3.

There are two main SPSA innovations in this paper (considered in Sections 4 and 5). The first is allowing for the loss functions to be time-varying (i.e., $\{L_k(\cdot)\}$ replaces $L(\cdot)$), which is a consequence of the adaptive control problem here. The second innovation is in using a smoothed version of the SP gradient approximation, which involves a weighted average of gradient estimates across iterations. In particular, the gradient approximation at any iteration will be a convex combination of the previous gradient approximation and a new SP estimate. Section 4 includes an a.s. convergence result for the smoothing-based algorithm with time-varying loss functions and Section 5 demonstrates the improvements in efficiency resulting from the smoothed approach.

For control applications, such as in this paper, improvements in the efficiency of the SA algorithm can translate into direct improvements in the performance of the underlying system. For instance, in a manufacturing process control setting, achieving a certain percentage reduction in number of iterations required for the controller to have the product meet quality specifications will lead to the same percentage reduction in number of products that are of unacceptable quality and must be discarded. Hence, to the extent that the above-mentioned smoothing is effective at increasing the efficiency of SPSA, we can expect an improvement in the system performance over that possible with non-smoothed SPSA.

3. Overview of Neural Network Approach to Control

This section summarizes our overall approach to the control of nonlinear systems in discrete-time form; more specific aspects of the approach associated with the use of SPSA will be given in Section 4. Subsection 3.1 describes the two methods for NN-based control that were mentioned in Subsection 1.2 above: one method applies when essentially nothing is known about the dynamics of the system and the other method applies when partial information on the dynamics is available. Subsection 3.2 is 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 control problem, which motivates the use of SPSA as discussed in Section 4.

3.1. The model and an overview of NN-based approach to control

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

$$x_{k+1} = \phi_k(x_k, x_{k-1}, \ldots, x_{k-s}, u_k, w_k), \quad s \geq 0, \tag{3.1}$$

where $\phi_k(\cdot)$ is a generally unknown, nonlinear function 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 serially independent random noise (the $\phi_k(\cdot)$ 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 (3.1) is the Markov formulation where $s = 0$. 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.

In the approach here, a NN will be used to produce the control u_k. Associated with the NN generating u_k will be a vector of connection weights $\theta_k \in R^p$ that must be estimated. We will assume that the NN structure (e.g., number of nodes and layers) is given. Hence the adaptive control problem of finding the optimal $u_k = u_k(\theta_k; x_k, x_{k-1}, \ldots, x_{k-s}; t_{k+1})$ is equivalent to finding the θ_k that minimizes some loss function $L_k(\theta_k)$ measuring system performance. A common function is the one-step-ahead quadratic loss:

$$L_k(\theta_k) = E\Big[(x_{k+1} - t_{k+1})^T A_k (x_{k+1} - t_{k+1}) + u_k^T B_k u_k\Big], \tag{3.2}$$

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. (The approach of this paper would apply equally well with other [non-quadratic] loss functions, as might arise, e.g., in constrained problems where penalties are included on certain values of x_{k+1} and/or u_k.) An important special case of (3.2) is the minimum variance regulator, where $A_k = I$ and $B_k = 0$. Note that although (3.2) is a one time-step error function, much of the adaptive control literature focuses on minimizing a loss function over an infinite horizon; Saridis (1977, pp. 291-296) is one of a number of references that discuss the relationship between the two approaches.

We will consider two methods for the problem of constructing a controller u_k in the face of uncertainty about the dynamics of the system, as illustrated in Figs. 3.1a,b for the important special case where $s = 0$ (for the more general case, as shown in (3.1), the diagrams would be modified in an obvious way). Both methods here correspond to *direct adaptive control* approaches, as defined (without a solution) in Narendra and Parthasarathy (1990), in that NN learning is based directly on the output error, $x_k - t_k$, during system operation. These are in contrast to the *indirect adaptive control* methods of Narendra and Parthasarathy

(and others mentioned in Section 1), which are based on the off-line prior 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 u_k inputs, which is then used in building the adaptive controller. As discussed in Section 1, there are distinct disadvantages to the indirect control method, aside from the obvious one of having to collect prior input-output data on the system, which may be costly or unavailable.

In the method of Fig. 3.1a, 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 when virtually nothing is known about $\phi_k(\cdot)$. In contrast to the direct approximation method of Fig. 3.1a, the self-tuning method of Fig. 3.1b requires that some prior information exists about the form of $\phi_k(\cdot)$. In particular, it requires that enough information be available to write $u_k = \pi_k(f_k(\cdot), t_{k+1})$, where $\pi_k(\cdot)$ is some known control law and $f_k(\cdot)$ is an unknown function that is to be approximated by a NN.

As we will see in Section 5, when prior information associated with knowledge of $\phi_k(\cdot)$ is available, the self-tuning method of Fig. 3.1b may yield a superior controller. Both the direct approximation and self-tuning methods require knowledge of which arguments appear in $\phi_k(\cdot)$, i.e., for the general setting of (3.1) it is required that s be known since $x_k, x_{k-1}, \ldots, x_{k-s}$ and t_{k+1} will be the input to the controller.

3.2. Challenges in connection weight estimation

Based on the error criterion in (3.2), 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 (1991)). Thus, we are seeking the value of θ_k, say θ_k^*, that minimizes (3.2) given the control as found in Figs. 3.1a,b. So, for each k, we are seeking the minimizing θ_k^* such that

$$\frac{\partial L_k}{\partial \theta_k} = \frac{\partial u_k^T}{\partial \theta_k} \cdot \frac{\partial L_k}{\partial u_k} = 0 \quad \text{at} \quad \theta_k = \theta_k^*. \tag{3.3}$$

Since $\phi_k(\cdot)$ is an unknown function, the term $\partial L_k/\partial u_k$ in (3.3), which involves the term $\partial \phi_k/\partial u_k$, is not generally computable. To illustrate further why $\partial L_k/\partial \theta_k$ is not available in our setting, consider a simple scalar-deterministic version of system (3.1). Then, under the standard squared-error loss,

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

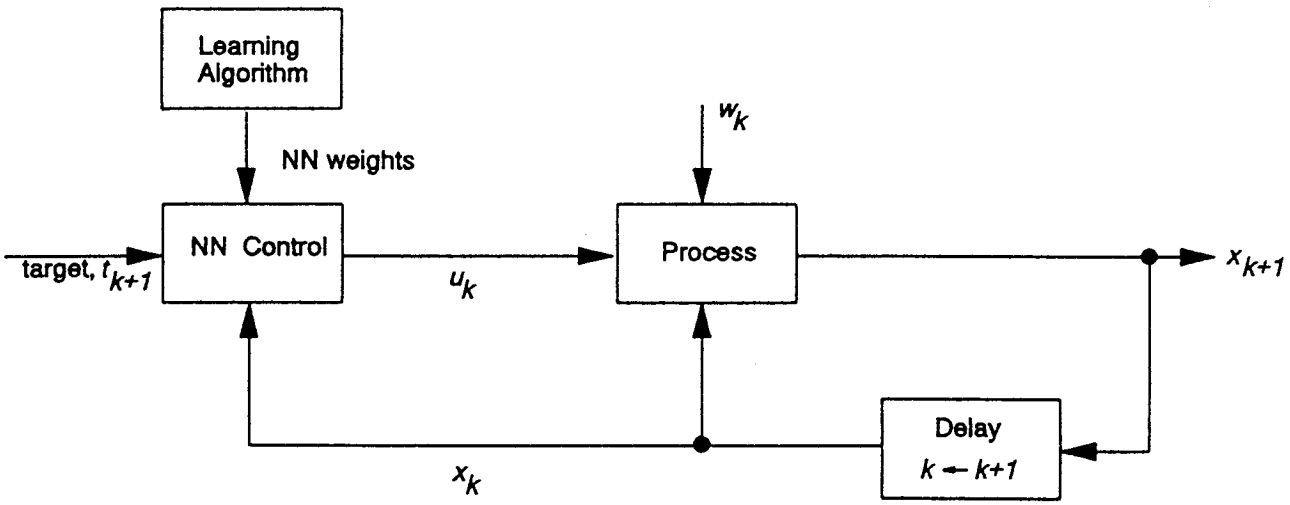

Figure 3.1a. Control system with NN as direct approximator to optimal control (assuming $s = 0$)

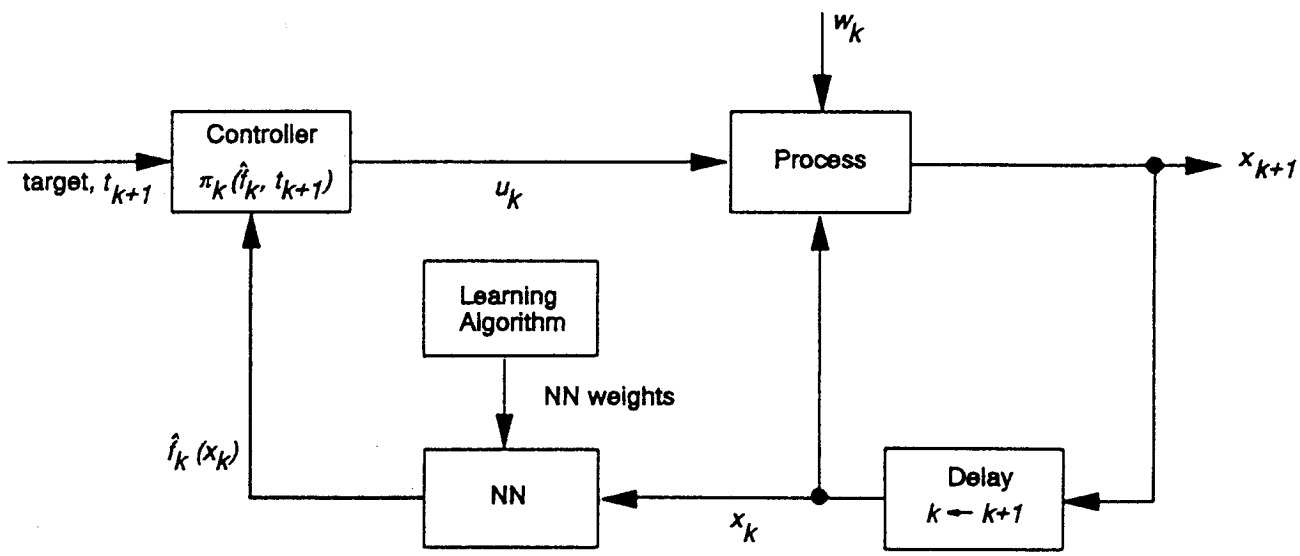

Figure 3.1b. Self-tuning control system with NN as approximator to $f_k(x_k)$ (assuming $s = 0$)

Since $\partial\phi_k/\partial u_k$ depends on knowledge of $\phi_k(\cdot)$ and on $f_k(\cdot)$ for the self-tuning controller, we see that in neither of the methods in Figs. 3.1a,b is $\partial L_k/\partial\theta_k$ generally available.[2] The same principles apply in the more general multivariate stochastic version of model (3.1). Thus the standard "back propagation" algorithm (i.e., steepest descent or Robbins-Monro stochastic approximation — see, e.g., Narendra and Parthasarathy (1991) or White (1989)) or any other algorithm requiring a direct observation of $\partial L_k/\partial\theta_k$ is not feasible.

The fact that $\partial L_k/\partial\theta_k$ (or a noisy observation of $\partial L_k/\partial\theta_k$, à la Robbins-Monro) is not available is inherently connected to the fact that feedback of the system dynamics is present. This contrasts with standard model fitting or identification problems in, e.g., White (1989, Eqns. (2.2)-(2.4)) or Narendra and Parthasarathy (1990, Sec. 5), where in estimating the connection weights no unknown functions appear in the gradient (i.e., the gradient of the squared-error loss function is derived using input-output data from the system and from the NN together with the known gradient of the NN with respect to the weights). This also contrasts with implementations of indirect feedback controllers as discussed above, 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.

To address the fact that $\partial L_k/\partial\theta_k$ is not computable, we consider a stochastic approximation algorithm of the generic form in (2.1), where $\hat{\theta}_k$ denotes the estimate of θ_k 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 $\phi_k(\cdot)$. The next section is devoted to describing in more detail the SA approach to this problem.

4. Weight Estimation by Stochastic Approximation with a Smoothed Gradient Approximation

This section presents the SA algorithm for use in estimating the NN connection weights. The algorithm applies in either the direct approximation control of Fig 3.1a or self-tuning control of Fig. 3.1b. It is based on a smoothed (across iteration) implementation of the simultaneous perturbation gradient approximation that was described in unsmoothed form in Subsection 2.2. Subsection 4.1 gives an overview of the algorithm and comments on how it is much more efficient (in terms of number of system operations) than the more standard finite-difference

[2] One special case where $\partial L_k/\partial\theta_k$ *can* be computed is in the self-tuning setting of Fig. 3.1b where $u_k(\cdot)$ is known to enter ϕ_k additively (since $\partial\phi_k/\partial u_k$ then does not depend on unknown quantities). Of course, in the more general setting of direct approximation control (Fig. 3.1a) $\partial L_k/\partial\theta_k$ would still be unavailable.

SA (FDSA) algorithm of Kiefer-Wolfowitz. This subsection includes a brief discussion on how the algorithm would be implemented in a practical control problem. Subsection 4.2 presents regularity conditions under which the algorithm yields a strongly convergent weight estimate.

4.1. Overview of SA algorithm based on smoothed gradient approximation

Recall that we are seeking the NN weight vector at each time point that minimizes (3.2), 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. In line with (2.1), the simultaneous perturbation SA (SPSA) algorithm here has the form

$$\hat{\theta}_k = \hat{\theta}_{k-1} - a_k G_k, \tag{4.1a}$$

where G_k is the smoothed approximation to $g_k(\hat{\theta}_{k-1})$. In particular for some $0 \leq \rho_k \leq 1$,

$$G_k = \rho_k G_{k-1} + (1-\rho_k)\hat{g}_k(\hat{\theta}_{k-1}), \quad G_0 = 0, \tag{4.1b}$$

where $\hat{g}_k(\cdot)$ is the simultaneous perturbation approximation to $g_k(\cdot)$, as in (2.2). That is, the ℓth component of $\hat{g}_k(\hat{\theta}_{k-1}), \ell = 1, 2, \ldots, p$, is given by

$$\hat{g}_{k\ell}(\hat{\theta}_{k-1}) = \frac{\hat{L}_k^{(+)} - \hat{L}_k^{(-)}}{2c_k\Delta_{k\ell}}, \tag{4.1c}$$

where

- $\hat{L}_k^{(\pm)} = (x_{k+1}^{(\pm)} - t_{k+1})^T A_k (x_{k+1}^{(\pm)} - t_{k+1}) + u_k^{(\pm)^T} B_k u_k^{(\pm)}$,
- $u_k^{(\pm)} = u_k(\hat{\theta}_{k-1} \pm c_k\Delta_k, t_{k+1}, x_k, \ldots, x_{k-s})$, 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, bounded, 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 to compute $\hat{g}_k(\cdot)$ (i.e., the numerators are the same for all p components) and hence to compute G_k. This is in contrast to the standard FDSA approach where $2p$ measurements are needed to construct the approximation to $g_k(\cdot)$, as

discussed in Subsection 2.2. Spall (1992) gives a detailed analysis of the SPSA approach to optimization in the classical setting of a fixed loss function $L(\cdot)$ and corresponding fixed minimum when no smoothing is used (i.e., $\rho_k = 0 \ \forall \ k$). As mentioned in Subsection 2.2, it is shown there that SPSA can achieve the same level of asymptotic accuracy as FDSA with only $1/p$ (or some low multiple of $1/p$) the number of system measurements. Of course, in the control setting here the loss function $L_k(\cdot)$ is not generally fixed and hence it can not be automatically assumed that the results of Spall (1992) would apply; further, the gradient smoothing introduces additional across-iteration dependence in the gradient estimate. Therefore, we show below that the SPSA estimation error $\hat{\theta}_k - \theta_k^*$ converges almost surely (a.s.) to 0 as in the fixed $L_k(\cdot)$ and unsmoothed setting.

There are several ways in which a practical control strategy could be implemented using the SPSA algorithm in (4.1a,b,c). These differences result from whether or not it is desired to produce a "nominal" state x_{k+1} based on a control with updated $\theta_k = \hat{\theta}_k$ (only $x_{k+1}^{(\pm)}$ are required in implementing (4.1a,b,c), which use $\theta_k = \hat{\theta}_{k-1} \pm c_k\Delta_k$) and whether or not it is possible to reset the system from a given state value to the previous state value. S&C (1992 [extended version], 1993) elaborate on these strategies for the unsmoothed SPSA algorithm; the S&C (1992, 1993) discussion would apply equally well to the smoothed gradient setting here since no additional quantities are needed to implement the SPSA recursion. As an illustration of these strategies, suppose that we wish to produce a sequence of system measurements that includes nominal states, i.e., the sequence to be produced is $\{x_0, x_1^{(+)}, x_1^{(-)}, x_1, x_2^{(+)}, x_2^{(-)}, x_2, \ldots\}$. Then, if the system cannot be readily reset from one state to the previous state (as is quite common), each of $u_k^{(+)}$, $u_k^{(-)}$, u_k are produced using the previous $s+1$ state measurements. Thus, for example, if $s = 0$, then $u_k^{(-)}$ is based on $\theta_k = \hat{\theta}_{k-1} - c_k\Delta_k, t_{k+1}$, and $x_{k+1}^{(+)}$. On the other hand if the system *can* be reset from one state to the previous state (perhaps as with the motion of a robot arm), the nominal x_k can be used in forming all of $u_k^{(+)}$, $u_k^{(-)}$, and u_k. We will illustrate both the non-reset and reset procedures in Section 5. There may also, of course, be periods when only nominal states might be generated (i.e., no SPSA updating of $\hat{\theta}_k$), e.g., when $\hat{\theta}_k$ has adequately converged (say, the error in tracking $\{t_k\}$ has reduced 80 percent from its initial value) and the process dynamics are stationary.

4.2. The convergence of the weight estimate $\hat{\theta}_k$

Let us now show that, as in the fixed loss function case of Spall (1992), the difference $\hat{\theta}_k - \theta^*$ is a.s. convergent to 0 where θ^* is such that $\theta_k^* \to \theta^*$ as $k \to \infty$. This result is presented in the Proposition below. Note that having $\theta_k^* \to \theta^*$ does *not* imply the (say) pointwise convergence of $L_k(\cdot)$ to some fixed $L(\cdot)$. In fact,

$L_k(\theta_k^*)$ may be perpetually varying even when $\theta_k^* = \theta^* \; \forall \, k$, as results, say, when t_k is perpetually varying. Below we let $\|\cdot\|$ denote any vector norm. The regularity conditions for the Proposition are:

C.0. $E(\epsilon_k^{(+)} - \epsilon_k^{(-)} | \hat{\theta}_1, \hat{\theta}_2, \ldots, \hat{\theta}_{k-1}; \Delta_k) = 0$ a.s. $\forall \, k$, where $\epsilon_k^{(\pm)}$ is the effective SA measurement noise, i.e., $\epsilon_k^{(\pm)} \equiv \hat{L}_k^{(\pm)} - L_k^{(\pm)}$, with $L_k^{(\pm)}$ the values of L_k based on the controls $u_k^{(\pm)}$.

C.1. $a_k, c_k > 0 \; \forall \, k$; $a_k \to 0$, $c_k \to 0$ as $k \to \infty$; $\sum_{k=1}^{\infty} a_k = \infty$, $\sum_{k=1}^{\infty} (a_k/c_k)^2 < \infty$.

C.2. $0 \leq \rho_k \leq 1 \; \forall \, k$; $\rho_k \to 0$ with $\rho_k / a_k = O(k^{-r})$ for some $r > 0$ such that $c_k^2 = O(k^{-r})$.

C.3. For some $\beta > 0$ and $\forall \, k$, $E(\epsilon_k^{(\pm)^2}) \leq \beta$, $E(L_k(\hat{\theta}_{k-1} \pm c_k \Delta_k)^2) \leq \beta$, $E(\Delta_{k\ell}^{-2}) \leq \beta$, $|\Delta_{k\ell}| \leq \beta$, and $\Delta_{k\ell}$ is symmetrically distributed about 0.

C.4. For some $\beta > 0$, there exists a $K < \infty$ such that for each $k \geq K$ and almost all $\hat{\theta}_{k-1}$ the function $L_k(\cdot)$ is continuously thrice differentiable with uniformly (in k) bounded third derivative for all θ in a neighborhood of $\hat{\theta}_{k-1}$ such that $\|\hat{\theta}_{k-1} - \theta\| \leq \beta$.

C.5. For some $\beta, \tau > 0$ and $K < \infty$ and each $k \geq K$, $\|\hat{\theta}_k - \hat{\theta}_{k-1}\| \leq \beta$ a.s. and given the $r > 0$ in C.2, $E\|\hat{\theta}_k - \theta^*\|^{\tau + 1/r} \leq \beta$.

C.6. For any $\beta > 0$ and for each $k \geq 1$ suppose that if $\|\theta - \theta^*\| \geq \beta$, there exists a $\delta_k(\beta) > 0$ such that $(\theta - \theta^*)^T g_k(\theta) \geq \delta_k(\beta)$ where $\delta_k(\beta)$ satisfies $\sum_{k=1}^{\infty} a_k \delta_k(\beta) = \infty$, $c_k^2 \delta_k(\beta)^{-1} \to 0$, and with r, τ as in C.5, $k^{-\epsilon} \delta_k(\beta)^{-\tau - 1/r} = O(1)$ for some $0 < \epsilon < r\tau$.

Let us now comment on conditions C.0-C.6. Condition C.0 is critical in ensuring that $\hat{g}_k(\cdot)$ is an unbiased estimator of $g_k(\cdot)$ to within an $O(c_k^2)$ bias, where $c_k \to 0$ by C.1. Condition C.1 presents some standard conditions on the SA gains (as discussed at the end of this subsection, however, it is generally best to *not* satisfy this condition in a system with nonstationary dynamics). C.2 ensures that, asymptotically, all the weight in the smoothed-gradient estimate is being put on the contribution associated with the current loss function; this helps ensure that G_k is an asymptotically unbiased estimate of the corresponding gradient $g_k(\cdot)$. C.3 ensures that the variability of $\hat{g}_k(\cdot)$ is not so large as to potentially cause divergence of the algorithm; the condition on $E(\Delta_{k\ell}^{-2})$ is particularly important for the validity of $\hat{g}_k(\cdot)$ as an estimator of $g_k(\cdot)$ and for the convergence of $\hat{\theta}_k$, and *disallows*, e.g., taking $\Delta_{k\ell}$ as uniformly or normally distributed (taking $\Delta_{k\ell}$ as symmetrically Bernoulli distributed satisfies this condition and has proven effective in our numerical studies). C.4 is used to ensure that $L_k(\cdot)$ is sufficiently smooth so that $\hat{g}_k(\cdot)$ is nearly an unbiased estimator of $g_k(\cdot)$ based on a third order expansion of $L_k(\cdot)$ about $\hat{\theta}_{k-1}$. Of course, C.4 translates into differentiability

conditions on $\phi_k(\cdot)$ and, with the self-tuning method of Fig. 3.1b, into differentiability conditions on the control law $\pi_k(\cdot)$. C.5 also plays a role in the asymptotic unbiasedness of G_k as well as ensuring that the estimate $\hat{\theta}_k$ does not stray too far from the fixed θ^*. Since the r in C.5 may be near 0, C.5 includes a fairly stringent moment condition. For practical purposes, this moment condition may sometimes be almost equivalent to a requirement that $\hat{\theta}_k$ lie in some compact subspace of R^p. C.6 ensures that if we are at a value $\hat{\theta}_{k-1}$ not at θ^*, the gradient $g_k(\cdot)$ is sufficiently steep (as well as pointing towards θ^*) so that there will be a tendency to push the next value $\hat{\theta}_k$ towards θ^*. Note that the non-uniformity (in k) that is allowed for $\delta_k(\theta)$ permits $L_k(\cdot)$ to "flatten out" in some fixed region around θ^* as k gets large provided that this flattening does not occur too fast. C.6 also effectively requires that the initial condition $\hat{\theta}_0$ be sufficiently close to θ^* so that $\hat{\theta}_k$ does not get stuck in a local minimum of L_k (the same issue arises, of course, in back-propagation-type algorithms); Chin (1993) discusses a technique by which SPSA can be used as a global optimizer for arbitrary initial conditions.

We now present the convergence result and proof. After the proof, we include a few comments on the practical aspects of the Proposition.

Proposition. *Let conditions* C.0-C.6 *hold and suppose there exists a fixed* θ^* *such that* $\theta_k^* \to \theta^*$ *as* $k \to \infty$. *Then*

$$\hat{\theta}_k - \theta^* \to 0 \quad \text{a.s.} \tag{4.2}$$

Proof. Recall that $G_k = \rho_k G_{k-1} + (1-\rho_k)\hat{g}_k(\hat{\theta}_{k-1})$ where G_k is dependent on $\Theta_{k-1} \equiv \{\hat{\theta}_0, \hat{\theta}_1, \ldots, \hat{\theta}_{k-1}\}$. From C.0, C.3, and C.4 it follows that

$$\begin{aligned} E(G_k|\Theta_{k-1}) &= \rho_k E(G_{k-1} \mid \Theta_{k-1}) + (1-\rho_k)(g_k(\hat{\theta}_{k-1}) + b_k) \\ &= \frac{\rho_k}{a_{k-1}}(\hat{\theta}_{k-1} - \hat{\theta}_{k-2}) + (1-\rho_k)(g_k(\hat{\theta}_{k-1}) + b_k) \end{aligned} \tag{4.3}$$

where by Spall (1992, Lemma 1) $c_k^{-2}\|b_k\|$ is uniformly bounded a.s. To show (4.2), we establish that the multivariate versions of conditions (i)-(iv) in Evans and Weber (1986) hold. Relative to (i), we have by (4.3) and (C.6) that for any $\epsilon > 0$

$$\begin{aligned} &\left\{\|\tilde{\theta}_{k-1}\| \geq \epsilon, a_k \tilde{\theta}_{k-1}^T E(G_k \mid \Theta_{k-1}) < 0\right\} \\ &\qquad \subseteq \left\{|\rho_k \tilde{\theta}_{k-1}^T(\hat{\theta}_{k-1} - \hat{\theta}_{k-2})/a_{k-1} + (1-\rho_k)\tilde{\theta}_{k-1}^T b_k| \geq \delta_k(\epsilon)\right\} \end{aligned} \tag{4.4}$$

where $\tilde{\theta}_k = \hat{\theta}_k - \theta^*$. Condition (i) is shown to hold if for the event on the r.h.s. of (4.4), $P(\{\cdot\}$ infinitely often$) = 0$. By the Markov inequality, the probability of

this event is bounded above by

$$\begin{aligned}
&\delta_k(\epsilon)^{-\tau-1/\tau} E|\rho_k \tilde{\theta}_{k-1}^T(\hat{\theta}_{k-1} - \hat{\theta}_{k-2})/a_{k-1} + (1-\rho_k)\tilde{\theta}_{k-1}^T b_k\,|^{\tau+1/\tau} \\
\leq\ &\left(\frac{4}{\delta_k(\epsilon)}\right)^{\tau+1/\tau} \left(E|\rho_k \tilde{\theta}_{k-1}^T(\hat{\theta}_{k-1} - \hat{\theta}_{k-2})/a_{k-1}|^{\tau+1/\tau} + E|(1-\rho_k)\tilde{\theta}_{k-1}^T b_k|^{\tau+1/\tau}\right) \\
=\ &\delta_k(\epsilon)^{-\tau-1/\tau} O(k^{-1-\tau\tau})
\end{aligned}$$

where the last line follows from (4.3) (including the uniform bound on $c_k^{-2} b_k$), C.2, and C.5. Thus for the event on the r.h.s. of (4.4), we have from C.6, $\sum_{k=1}^{\infty} P(\{\cdot\}) < \infty$, which implies (i) by the Borel-Cantelli lemma. Condition (ii) follows from $a_k \to 0$ and the fact that (4.3), C.4, and C.5 imply

$$\limsup_{k\to\infty} \|E(G_k|\Theta_{k-1})\|(1 + \|\tilde{\theta}_{k-1}\|)^{-1} < \infty \quad \text{a.s.}$$

The fact that

$$\sum_{k=1}^{\infty} a_k^2 E\|G_k - E(G_k|\Theta_{k-1})\|^2 < \infty$$

follows easily by C.1 and C.3, which shows (iii). Finally, we can show (iv) by establishing that

$$\begin{aligned}
&P\left(\liminf_{k\to\infty} \|\tilde{\theta}_{k-1}\| > 0,\ \sum_{k=1}^{\infty} a_k \left\|\rho_k(\hat{\theta}_{k-1} - \hat{\theta}_{k-2})/a_{k-1} + (1-\rho_k)(g_k(\hat{\theta}_{k-1}) + b_k)\right\| < \infty\right) \\
&= 0 \qquad (4.5)
\end{aligned}$$

(recall that the expression inside the $\|\cdot\|$ terms in the summands of (4.5) is $E(G_k|\Theta_{k-1})$). For any sample point in the underlying sample space such that $\liminf_{k\to\infty}\|\tilde{\theta}_{k-1}\| > 0$ take $0 < \epsilon < \liminf_{k\to\infty}\|\tilde{\theta}_{k-1}\|$. Then $\exists$ a $k_1(\epsilon)$ such that $\|\tilde{\theta}_{k-1}\| \geq \epsilon\ \forall\ k \geq k_1$, which by C.2 and C.6 indicates that for each $k \geq k_1$ $\exists$ a $\delta_k'(\epsilon) > 0$ such that $(1-\rho_k)\|g_k(\hat{\theta}_{k-1})\| \geq \delta_k'(\epsilon)$. Further, by the a.s. uniform bounds on $\hat{\theta}_{k-1} - \hat{\theta}_{k-2}$ and b_k and the fact that $\rho_k/a_{k-1} \to 0$ and $b_k = O_p(c_k^2)$, we have (except, of course, on a set of measure 0) that $\exists$ a $k_2(\epsilon)$ such that $\|\rho_k(\hat{\theta}_{k-1} - \hat{\theta}_{k-2})/a_{k-1} + (1-\rho_k)b_k\| \leq \delta_k'(\epsilon)/2\ \forall\ k \geq k_2$. Hence, for this sample point, we have

$$\sum_{k=1}^{\infty} a_k \|E(G_k|\Theta_{k-1})\| \geq \sum_{k=k^*}^{\infty} a_k \delta_k'(\epsilon)/2 = \infty$$

by C.1 and C.6 where $k^* = \max\{k_1, k_2\}$. Since $\exists$ such a k^* and $\{\delta_k'\}$ for almost all sample points such that $\liminf\|\tilde{\theta}_{k-1}\| > 0$, this shows (4.5) (and hence (iv)), which completes the proof.

Let us close this section with a discussion of several practical issues associated with the application of the Proposition. The condition $\theta_k^* \to \theta^*$ is quite weak. For example, in the reasonably common situation where the system dynamics are time invariant (i.e., $\phi_k(\cdot) = \phi(\cdot)$ with stationary noise distribution) the even stronger condition $\theta_k^* = \theta^* \ \forall \ k$ generally holds (this follows since the optimal control can generally be expressed as a function independent of k, as discussed, e.g., in Nijmeijer and van der Schaft (1990, Ch. 14)). The more general condition of the Proposition allows for transient effects in $\phi_k(\cdot)$. Further, the critical martingale difference-type condition C.0 (which is perhaps the main condition in ensuring that the gradient estimate is asymptotically unbiased) is satisfied in all settings where the process noise w_k is additive and independent (it can hold in other settings as well, as illustrated in the example from Yaz (1987) discussed in Section 5). Finally, S&C (1992) discuss the use of constant SA gains (i.e., $a_k = a$ and $c_k = c \ \forall \ k$), which do *not* satisfy condition C.1. Such constant gains are useful when θ_k^* is perpetually time varying, as they provide an algorithm that is better able to adjust to changing dynamics than one with more standard decaying gains (see, e.g., Kushner and Huang (1981)). In this paper, however, we will continue to focus on the decaying gain setting.

5. Empirical Studies

5.1. Preliminaries

This section presents the results of our studies on two different nonlinear models. The first model has the control entering multiplicatively and the noise entering additively and the second model has the control entering additively and the noise entering non-additively. For the two models, we will compare the performance of the smoothed ($\rho_k > 0$) and the unsmoothed ($\rho_k = 0$) SPSA algorithms in the direct approximation (DA) controller of Fig. 3.1a. In addition, for the first model (multiplicative control) we will consider the relative performance of the smoothed and unsmoothed algorithms using the self-tuning (ST) controller and also with changes in the level of process noise w_k.

All studies here are based on $A_k = I$ and $B_k = 0$ in the loss function (3.2) (i.e., a minimum variance regulator). S&C (1992 [extended version]) present numerical results for $B_k \neq 0$, showing no significant difference in the relative performance of various algorithms from when $B_k = 0$. The performance of the various techniques will be evaluated by comparing estimates of the root-mean-square (RMS) tracking error as normalized by the dimension of x_k, i.e., the RMS at time k that we estimate is $[E(x_k - t_k)^T(x_k - t_k)/\dim(x_k)]^{1/2}$. The (feedforward) NNs considered here have an input layer, two hidden layers, and an output layer, as in Narendra and Parthasarathy (1990) and Chen (1990). The hidden layer nodes are the

hyperbolic tangent functions (i.e., $(e^y - e^{-y})/(e^y + e^{-y})$ for input y) and the output nodes are linear functions (i.e., output = y). Each node takes as an input (y) the weighted sum of outputs of all nodes in the previous layer plus a bias weight not connected to the rest of the network as illustrated in Fig. 2.1 (hence an $N_{4,20,10,2}$ network, as sometimes used below, has $100 + 210 + 22 = 332$ weights to be estimated). For the SPSA algorithm we take the perturbations Δ_{ki} to be Bernoulli ± 1 distributed, which satisfies the relevant regularity conditions of Section 4.

In the numerical studies of S&C (1992, 1993), several analyses are conducted that would be relevant here as well. For example, these references compare the performance of SPSA and FDSA, demonstrating that they have essentially equivalent long-run performance even though SPSA uses a much lower number of system measurements. This, of course, is the fundamental motivation behind using SPSA. S&C (1992, 1993) also examine the performance of SPSA when several conditionally independent SP gradient approximations are averaged at each iteration (not to be confused with the smoothing here where the gradients are averaged *across* iterations) and show that this can often be of benefit despite the cost of taking additional system measurements at each iteration.

5.2. Results of numerical studies

Let us now present our studies on the two different nonlinear models, where, as mentioned above, the study of the first model will include additional analysis beyond the comparison of smoothing and non-smoothing in the DA method.

Multiplicative Control/Additive Noise Model

The first model we consider for generating state measurements has the popular (e.g., Chen (1990)) multiplicative control form:

$$\begin{aligned} x_{k+1} &= (1 - f^{(0)}(x_k))u_k + w_k, \\ &\equiv f(x_k)u_k + w_k, \quad w_k \overset{\text{i.i.d.}}{\sim} N(0, \sigma^2), \end{aligned} \tag{5.1}$$

where $x_k \in R^1$ and, as in Equation (3.2) of Lai and Zhu (1991) or Sections 4 and 5 of Al-Qassam and Lane (1989), $f^{(0)}(x_k)$ is given by the exponential-based function

$$f^{(0)}(x_k) = \left(-.3 - .8e^{-x_k^2} \right) x_k.$$

The target sequence $\{t_k\}$ will be periodic with one period being a 50 iteration sine wave followed by a 50 iteration square wave where both waves have amplitude one.

In the DA method nothing is assumed about the relationship in (5.1) except that $\dim(u_k) = 1$ and that $s = 0$ (as in Fig. 3.1a). In the ST technique we

assume that enough information is available to know that $f(\cdot)$ and u_k multiply each other, and that w_k enters additively, but, of course, we need not assume that the form of the dynamics $f(\cdot)$ is known since it is $f(\cdot)$ that the NN will approximate; hence, we take $u_k = \pi_k(\hat{f}(x_k), t_{k+1}) = t_{k+1}/\hat{f}(x_k)$ since w_k has mean 0. As with Narendra and Parthasarathy (1990) we used NNs with two hidden layers, one of 20 nodes and one of 10 nodes (so an $N_{2,20,10,1}$ was used for the DA controller while an $N_{1,20,10,1}$ was used for the ST controller).

Figs. 5.1 and 5.2 present the main results for our study of the model in (5.1) in both the noise-free ($\sigma = 0$) and noisy ($\sigma = .5$) setting. We know that the long-run RMS can at best equal σ. The controller was implemented without resetting, as described in Subsection 4.1. The sample RMS value at any time point was computed by taking the square root of the mean of four independent mean-square errors based on four runs with different initial weights $\hat{\theta}_0$. The elements of $\hat{\theta}_0$ were generated randomly from a uniform $(-.5, .5)$ distribution with the exception of the bias weight on the output of the NN in the ST controller, which was set to 1.0 to reflect prior information that $f(x_0) \geq 1$ (this enhanced the performance of the ST controller by reducing the likelihood of division by zero in forming $u_k = \pi_k(\cdot)$). To effect a fair comparison of the algorithms the same four initial weight vectors and four pairs of random number seeds (to initialize the w_k and Δ_k sequences) were used in the DA runs; likewise for the ST runs. To further smooth the resulting RMS error curves and to show typical performance (not just case-dependent variation), we applied the MATLAB low-pass interpolating function INTERP to the RMS 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 using SA gains of the form $a_k = a/k^{.602}$ and $c_k = c/k^{.101}$ (which satisfy condition C.1 of the Proposition). For each comparison we attempted to tune the constants a, c to approximately maximize the rate-of-convergence for the unsmoothed algorithm; the same a, c were then used in the smoothed algorithm in order to be conservative in considering the performance of this approach (it was found for these particular studies that $.005 \leq a \leq .015$, $.04 \leq c \leq .10$).[3] For the figures $x_0 = 10$, so the initial RMS error is 10. When gradient smoothing was used, we took $\rho_k = .5/k^{.603}$ (note that $\rho_k/a_k = O(k^{-r})$, as required by condition C.2, with $r = .001$).

Figs. 5.1 and 5.2 show that without and with process noise both the smoothed

[3]The choice of a, c is important for adequate performance of the algorithm (analogous to choosing the step-size in back-propagation). For example, choosing a too small may lead to an excessively slow reduction in tracking error while choosing an a too large may cause the system to go unstable (so, for practical problems, it might be appropriate to begin with a relatively small a and gradually increase it until there is an adequate convergence rate but little chance of going unstable).

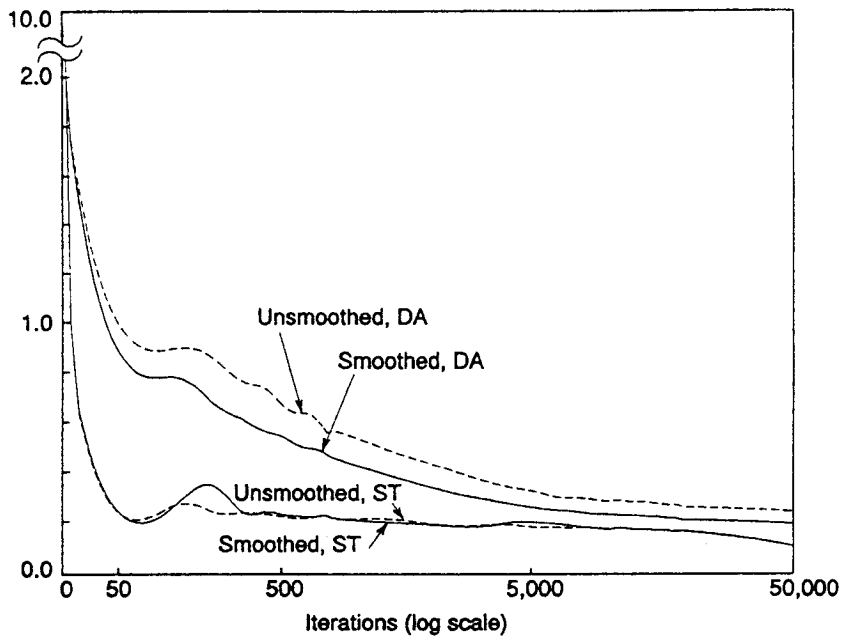

Figure 5.1. RMS error for smoothed and unsmoothed DA and ST controllers in multiplicative control model (5.1) with $\sigma = 0$

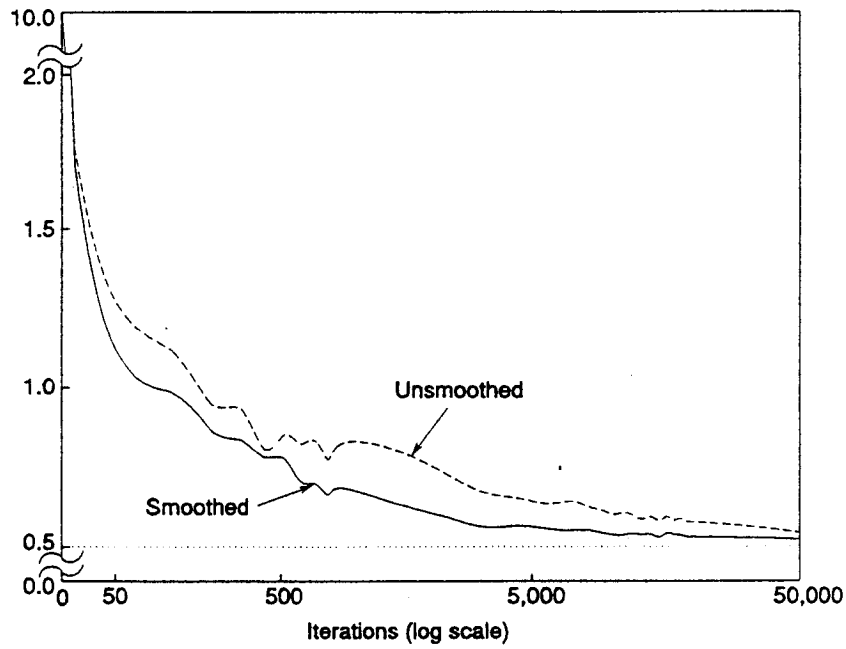

Figure 5.2. RMS errors for smoothed and unsmoothed DA controller in multiplicative control model (5.1) with $\sigma = 0.5$

and unsmoothed implementation yield controllers with decreasing RMS tracking error. In all cases, the RMS error curves show the characteristic shape of first-order (steepest-descent-type) algorithms in that there is a sharp initial decline followed by slow decline. Hence, over 90 percent of the possible reduction in RMS error — which may be all that is required in many applications — occurs within the first 100 iterations in all error curves of the two figures. In the DA cases, the smoothed algorithm outperforms the unsmoothed, but for the ST method there was little long-term difference between smoothed and unsmoothed algorithms. This may result from the fact that the additional information being used in ST overpowers the relative gains of smoothing. Note that the differences between the smoothed and unsmoothed DA algorithms in Figs. 5.1 and 5.2 remain significant throught the whole range of iterations. This apparently is a consequence of the different search direction taken by the smoothed algorithm in the first few iterations since there is a little contribution of previous gradient information to the current gradient approximation in the later iterations (e.g., at $k = 20$, only 8.2% of the contribution to G_k is due to G_{k-1} and at $k = 1000$ this contribution is less than 0.8%).

Additive Control/Non-Additive Noise Model

The second model we consider is one where the control is added to the dynamics and the noise is not additive. In particular, as in Yaz (1987), the data are generated according to

$$x_{k+1} = \begin{pmatrix} -.5 & .3 \\ 0 & 1.1 \end{pmatrix} x_k + \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix} u_k + \begin{pmatrix} \|x_k\| \\ 0 \end{pmatrix} w_k, \quad x_k, u_k \in R^2, \tag{5.2}$$

where w_k is an independent scalar Bernoulli $\pm.5$ noise process and $\|\cdot\|$ denotes the Euclidean norm. As noted by Yaz, (5.2) corresponds to an unstable system. Hence, in contrast to the study for model (5.1), we employed a controller where the system could be reset from one state value to the previous state value as described in Subsection 4.1 (we found that the control/estimation algorithm here performed poorly if the system was not reset). We consider two constant target sequences, $t_k = (0,0)^T \; \forall \; k$ and $t_k = (1,0)^T \; \forall \; k$, which yield long-run best possible RMS errors of 0 and $1/\sqrt{2}$ respectively. Aside from the multiplicative (possibly unstable) mode in which the noise enters (5.2), this model is interesting since only one of the two control elements affects the system and since the first element of x_k can only be affected by a control after a delay of one time period. By the fact that $\{w_k\}$ is independent, the important condition C.0 on the effective SA noise is satisfied here as it is in the simpler additive w_k setting. Also, analogous to the studies above, we used an $N_{4,20,10,2}$ network for the controller.

Fig. 5.3 shows the results of the study with model (5.2). The RMS error curves are formed from the same averaging/interpolation scheme used in the study for model (5.1). The figure includes runs for the above-mentioned two different constant target sequences (which yield two different lower bounds to the long-run RMS error). The SA gains were chosen as before (i.e., approximately optimally for the unsmoothed algorithm and decaying at the rates specified above) with $a = .01$, $c = .75$ for the zero target and $a = .005$, $c = .75$ for the non-zero target; when smoothing is used we take (as above) $\rho_k = .5/k^{.603}$. We see that the smoothed algorithm yields a significantly lower RMS over the full range of iterations in the zero target case and a slightly lower RMS over almost all iterations in the non-zero target case. As with the earlier studies, ρ_k is essentially 0 for most of the range of iterations, yet the benefits gained from non-zero ρ_k in the first few iterations put the algorithm on a more stable path nearer the target for almost the whole range of iterations. Also keep in mind that the SA gains were tuned to optimize the performance of the *unsmoothed* algorithm; we have found that the smoothed algorithm can be made to work even better if the gains are tuned appropriately.

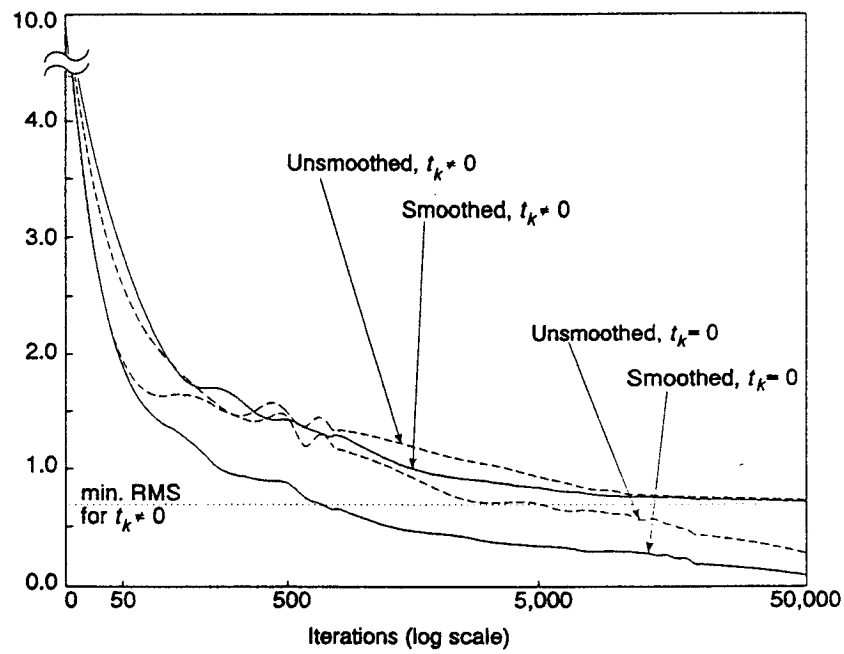

Figure 5.3. RMS errors for smoothed and unsmoothed DA controller in additive control model (5.2) with zero and nonzero target sequences

6. Concluding Remarks

This paper has considered the problem of controlling nonlinear stochastic

systems with unknown process equations. The adaptive control approach here is based on adjusting the weights in a neural network using the output error of the system. Our approach differs from those of a number of others in that it does not rely on an explicit model for the system dynamics (either based on a NN or other parametric form [e.g., nonlinear ARMA]) to be used in conjunction with a NN model for the controller. Related to this, it does not rely on prior identification of such a system model based on specified input-output data.

Since we are not assuming full knowledge of the structure of the equations describing the process dynamics, it is not possible to calculate the gradient of the loss function for use in the usual NN back-propagation-type search algorithms. Therefore, we describe a stochastic approximation-based method for the NN weight estimation, which is based on a "simultaneous perturbation" approximation to the unknown gradient (Spall (1988, 1992)). This method relies on observing the system state at two (or a low multiple of two) levels of the control to construct the gradient approximation and is therefore able to estimate the weights with many fewer measurements than the standard finite-difference SA technique of Kiefer-Wolfowitz and others.

This paper has also introduced a smoothed gradient (SPSA) algorithm for estimating the weights in a neural network. The gradient smoothing of this paper is a first step in developing an accelerated implementation of the SPSA algorithm. The smoothing is in the spirit of the well-known conjugate gradient algorithm of deterministic optimization in that a recursion for the gradient approximation is formed that combines current gradient information with the gradient approximation of the previous iteration. Although the smoothing idea would apply in virtually any use of SPSA as a general optimization technique, it has particular potential in control problems since improved SPSA efficiency can translate directly into improved system performance.

This paper gives conditions under which the smoothed implementation of SPSA shares the almost sure convergence property of the unsmoothed version in Spall (1992). We also describe numerical studies on two different types of nonlinear models. These studies indicate that the smoothed algorithm often works better than the unsmoothed (in terms of reducing the magnitude of the error in tracking the target) and, at worst, seems to work at least as well as the unsmoothed. Further, there are no additional costs involved in using the smoothed approach, either in terms of computational burden or the amount of data required. This improved performance and lack of additional costs suggests that one should consider using a smoothed algorithm in most applications of SA in NN-based control.

Acknowledgement

This work was partially supported by the JHU/APL IRAD Program and U.S. Navy Contract N00039-91-C-0001.

References

Al-Qassam, M. S. and Lane, J. A. (1989). Forecasting exponential autoregressive models of order 1. *J. Time Series Anal.* **10**, 95–113.

Bayard, D. S. (1991). A forward method for optimal stochastic nonlinear and adaptive control. *IEEE Trans. Auto. Control* **36**, 1046–1053.

Blum, J. R. (1954). Multidimensional stochastic approximation methods. *Ann. Math. Statist.* **25**, 737–744.

Chen, F.-C. (1990). Back-propagation neural networks for nonlinear self-tuning adaptive control. *IEEE Control Syst. Mag.*, April, 44–48.

Chin, D. C. (1993). A more efficient global optimization algorithm based on Styblinski and Tang (1990). *Neural Nets.*, to appear 1994 (also JHU/APL report PSA–93–119).

Davis, M. H. A. and Vinter, R. B. (1985). *Stochastic Modeling and Control.* Chapman and Hall, New York.

Evans, S. N. and Weber, N. C. (1986). On the almost sure convergence of a general stochastic approximation procedure. *Bull. Austral. Math. Soc.* **34**, 335–342.

Funahashi, K. I. (1989). On the approximate realization of continuous mappings by neural network. *Neural Nets.* **2**, 183–192.

Hornik, K., Stinchcombe, M. and White, H. (1989). Multilayer feedforward networks are universal approximators. *Neural Nets.* **2**, 359–366.

Huang, S.-C. and Huang, Y. -F. (1991). Bounds on the number of hidden neurons in multilayer perceptions. *IEEE Trans. Neural Nets.* **2**, 47–55.

Hunt, K. J. and Sbarbaro, P. (1991). Neural networks for nonlinear model control. *IEE (Great Britain) Proc.-D* **138**, 431–438.

IEEE Reprint Series (1986). *Adaptive Methods for Control System Design.* IEEE Press, New York.

IEEE Reprint Series (1991). *Advances in Adaptive Control.* IEEE Press, New York.

Kiefer, J. and Wolfowitz, J. (1952). Stochastic estimation of the maximum of a regression function. *Ann. Math. Statist.* **23**, 462–466.

Kushner, H. J. and Huang, H. (1981). Asymptotic properties of stochastic approximations with constant coefficients. *SIAM J. Control Optimiz.* **19**, 87–105.

Lai, T. L. and Zhu, G. (1991). Adaptive prediction in non-linear autoregressive models and control systems. *Statistica Sinica* **1**, 309–334.

Narendra, K. S. and Parthasarathy, K. (1990). Identification and control of dynamical systems using neural networks. *IEEE Trans. Neural Nets.* **1**, 4–26.

Narendra, K. S. and Parthasarathy, K. (1991). Gradient methods for the optimization of dynamic systems containing neural networks. *IEEE Trans. Neural Nets.* **2**, 252–262.

Nijmeijer, H. and van der Schaft, A. J. (1990). *Nonlinear Dynamical Control System.* Springer-Verlag, New York.

Pao, Y.-M., Phillips, S. M. and Sobajic, D. J. (1992). Neural-net computing and the intelligent control of systems. *Internat. J. Control* **56**, 263–289.

Robbins, H. and Monro, S. (1951). A stochastic approximation method. *Ann. Math. Statist.* **22**, 400–407.

Ruppert, D. (1983). Kiefer-Wolfowitz procedure. *Encyclopedia of Statistical Sciences* **4** (Edited by S. Kotz and N. L. Johnson), 379–381, John Wiley, New York.

Ruszczynski, A. and Syski, W. (1983). Stochastic approximation method with gradient averaging for unconstrained problems. *IEEE Trans. Auto. Control* **28**, 1097–1105.

Saridis, G. N. (1977). *Self-Organizing Control of Stochastic Systems.* Marcel Dekker, New York.

Spall, J. C. and Cristion, J. A. (1992). Direct adaptive control of nonlinear stochastic systems using neural networks and stochastic approximation. *Proc. IEEE Conf. Dec. Control*, 878–883 (extended version available upon request).

Spall, J. C. and Cristion, J. A. (1993). A neural network controller for systems with unmodeled dynamics. *IEEE Trans. Syst., Man, and Cyber.*, submitted.

Spall, J. C. (1988). A stochastic approximation algorithm for large-dimensional systems in the Kiefer-Wolfowitz setting. *Proc. IEEE Conf. Dec. Control*, 1544–1548.

Spall, J. C. (1992). Multivariate stochastic approximation using a simultaneous perturbation gradient approximation. *IEEE Trans. Auto. Control* **37**, 332–341.

Tulunay, E. (1991). Introduction to neural networks and their application to process control. *Neural Networks: Advances and Applications* (Edited by E. Gelene), 241–273, Elsevier, Amsterdam.

Wasserman, P. D. (1989). *Neural Computing: Theory and Practice.* van Nostrand Reinhold, New York.

White, H. (1989). Some asymptotic results for learning in single hidden-layer feedforward network models. *J. Amer. Statist. Assoc.* **84**, 1003–1013.

Yaz, E. (1987). A control scheme for a class of discrete nonlinear stochastic systems. *IEEE Trans. Auto. Control* **32**, 77–80.

Applied Physics Laboratory, The Johns Hopkins University, Johns Hopkins Road, Laurel, MD 20723-6099, U.S.A.

(Received March 1992; accepted August 1993)