A Neural Network Controller for Systems with Unmodeled Dynamics with Applications to Wastewater Treatment

James C. Spall, *Senior Member, IEEE,* and John A. Cristion, *Member, IEEE*

***Abstract*— This paper considers the use of neural networks (NN's) in controlling a nonlinear, stochastic system with unknown process equations. The approach here is based on using the output error of the system to train the NN controller *without* the need to assume or 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. In principle, stochastic approximation algorithms in the standard (Kiefer–Wolfowitz) finite-difference form can be used for this weight estimation since they are based on gradient *approximations* from available system output errors. However, these algorithms will generally require a prohibitive number of observed system outputs. 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. It is shown that this algorithm can greatly enhance the efficiency over more standard stochastic approximation algorithms based on finite-difference gradient approximations. The approach will be illustrated on a simulated wastewater treatment system with stochastic effects and nonstationary dynamics.**

***Index Terms*— Adaptive control, gradient estimation, neural networks, simultaneous perturbation, stochastic approximation.**

I. Introduction

ONE OF THE major problems faced by system designers in many fields is finding a means to control and regulate the system under consideration given uncertainty about the nature of the underlying process. Adaptive control procedures have been designed to address this problem for certain types of system models (e.g., linear or special forms of nonlinear models) where only the parameters (not the model structure) are unknown. These techniques are based on using the known model form to construct a control law with unknown parameters and then using the system data to estimate these parameters. Although adaptive control procedures have been applied in a variety of areas (e.g., robot arm manipulation, aircraft control, etc.) they are limited by the need to assume that the forms of the system equations are known. For a complex process, however, the forms of the system equations may be unknown, making it impossible to determine the required control law for use in existing adaptive control procedures. This provides the motivation for considering the use of neural networks (NN's) in adaptive control.

Artificial neural networks, or more commonly just NN's, 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 NN's 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 [20] for a brief review of the nonlinear control problem). NN's, however, offer the potential for addressing control problems even broader than this, including the control of stochastic systems with unknown nonlinear dynamics.

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 [18], [19], Tulunay [32], Hunt and Sbarbaro [11], Pao, Phillips, and Sobajic [20], or Yamada and Yabuta [34]). Although these methods are useful under certain 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.

The NN in the approach of this paper is used to directly model the resulting unknown control law *without* the need to construct a separate model (NN or other type) for the unknown process dynamics. The basis for this approach is the now well-known fact that any continuous function can be approximated to within any degree of accuracy by some single (or multiple) hidden-layer feed-forward NN (e.g., Funahashi [8] or Hornik, Stinchcombe, and White [10]). Three of the major advantages of such model-free direct control techniques (versus the indirect control approaches mentioned above) are

Manuscript received August 1, 1993; revised August 5, 1994 and March 19, 1996. This work was supported in part by the JHU/APL IRAD Program and U.S. Navy Contract N00039-95-C-0002.

The authors are with the Applied Physics Laboratory, The Johns Hopkins University, Laurel, MD 20723-6099 USA (e-mail: james.spall@jhuapl.edu).

Publisher Item Identifier S 1083-4419(97)02914-2.

that they: i) require no "open loop" training data, ii) tend to be better able to handle changes in the underlying system behavior (since they are not tied to a prior model), and iii) tend to be more robust in the face of widely varying control inputs (i.e., the indirect approach may perform poorly for closed-loop controls outside of the range of open-loop controls used in the a priori training step). Of course, in some systems (such as when the dynamics are well understood) the more traditional model-based approaches may be more effective; see Passino [21, p. 15] for discussion of this (since the technique here is based on formal estimation principles of stochastic approximation, it does, however, seem to eliminate one of the issues raised by Passino, namely that "heuristics are all that is available to perform controller design").

To implement such an adaptive control approach, it is required that the 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, as described below. Therefore, this paper considers in Section III the use of a relatively new stochastic approximation algorithm for this weight estimation, which is based on a "simultaneous perturbation" gradient approximation. Based on a simulated wastewater treatment plant, it is illustrated in Section IV that this algorithm can greatly enhance the efficiency over more standard stochastic approximation algorithms based on finite-difference gradient approximations.

II. Overview of Neural Network Approach to Control

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

$$x_{k+1} = \phi_k(x_k, u_k, w_k) \tag{2.1}$$

where $\phi_k(\cdot)$ is an unknown, nonlinear function governing the dynamics of the system, u_k is the control input applied to govern the system at time $k+1$, and w_k is a random noise vector. Our goal is to choose the control vectors $\{u_k\}$ in a manner such[1] that the state values $\{x_k\}$ are close to a corresponding set of target (reference) vectors $\{t_k\}$, where "close" is relative to the magnitude of the noise and the cost associated with the control. The information being fed into the controller includes the M most recent state values and N most recent controls; in addition other information such as the next target vector or certain known system characteristics (e.g., exogenous inputs) may be included as input to the control. Note from (2.1) that this paper is focusing on the case of direct state measurements (as with most of the cases considered in [18]); this restriction is not critical since the basic ideas here transfer readily to the more general case of separate state and measurement relationships as discussed in Spall and Cristion [29].

In the approach here, the output of the NN will correspond to the value of the control u_k, as shown in Fig. 1 for the case where $M = N = 1$ (for the setting where M and/or N is >1, the figure would be modified in an obvious way). Associated with the NN producing 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 $u_k = u_k(\theta_k; x_k, x_{k-1}, \ldots, x_{k-M+1}; u_{k-1}, u_{k-2}, \ldots, u_{k-N}; t_{k+1})$, given that $\{x_k, x_{k-1}, \ldots; u_{k-1}, u_{k-2}, \ldots\}$ have already occurred, 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

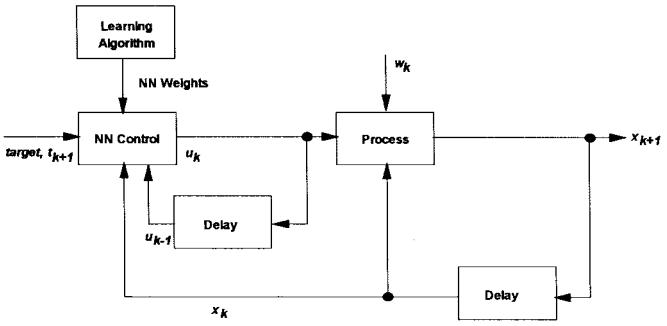

Fig. 1. Control system with NN as approximator to optimal u_k when $M = N = 1$.

$$L_k(\theta_k) = E\left[(x_{k+1} - t_{k+1})^T A_k (x_{k+1} - t_{k+1}) + u_k^T B_k u_k\right] \tag{2.2}$$

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. Note also that if the unconditional loss function $L_k(\theta_k)$ were replaced by a conditional loss, as is sometimes seen in the control literature (e.g., (2.2) replaced by an expected tracking error *conditional* on previous measurements and controls), the same optimal θ_k would typically result. This follows since under standard analysis conditions on the interchange of a derivative and an integral, $E(\partial L_k^{\text{cond}}/\partial\theta_k) = \partial E(L_k^{\text{cond}})/\partial\theta_k = \partial L_k/\partial\theta_k = 0$ at only the optimal θ_k (within the domain of interest) as discussed below, where L_k^{cond} represents the conditional loss. The approach of this paper would apply equally well with other (nonquadratic) 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.

Given the goal of minimizing $L_k(\theta_k)$ at each k, we are seeking the optimal θ_k, say θ_k^*, such that at θ_k^*

$$g_k(\theta_k) \equiv \frac{\partial L_k}{\partial \theta_k} = \frac{\partial u_k^T}{\partial \theta_k} \cdot \frac{\partial L_k}{\partial u_k} = 0 \tag{2.3}$$

where we assume that all indicated derivatives exist. We assume that θ_k^* corresponds to a global minimum or at least a sufficiently good local minimum. Since the system dynamics $\phi_k(\cdot)$ are unknown, the term $\partial L_k/\partial u_k$, which involves the term $\partial\phi_k/\partial u_k$, is not generally computable. Thus the standard "back-propagation" algorithm (i.e., steepest descent—see, e.g., Narendra and Parthasarathy [18], [19]), or any other algorithm involving $\partial L_k/\partial\theta_k$, is not feasible. This contrasts with implementations of indirect feedback controllers where a separate

[1] Note that although (2.2) is a one-step-ahead error function, much of the adaptive control literature focuses on minimizing a loss function over an infinite time horizon, Moden and Soderstrom [17] is one of a number of references that discuss the relationship between the two approaches.

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.

Therefore, we consider a stochastic approximation (SA) algorithm of the form

$$\hat{\theta}_k = \hat{\theta}_{k-1} - a_k(\text{gradient approx.})_k \tag{2.4}$$

to estimate $\{\theta_k^*\}$, where $\hat{\theta}_k$ denotes the estimate at the given iteration, $\{a_k\}$ is a scalar gain sequence (see discussion below), and the gradient approximation is such that it does not require knowledge of $\phi_k(\cdot)$. This algorithm is a stochastic version of the well-known (deterministic) gradient descent algorithm. The next section is devoted to describing in more detail the SA approach to this problem.[2]

III. Weight Estimation by Simultaneous Perturbation Stochastic Approximation

A. Overview of Approach

This subsection gives an overview of the simultaneous perturbation SA (SPSA) approach to the problem in (2.3); the next subsection considers some more detailed aspects of practical implementation. Spall [24] gives a thorough analysis of the SPSA approach to general optimization problems. It is shown that the SPSA algorithm can, under fairly general conditions, achieve the same level of asymptotic accuracy in estimating the NN weights as the standard finite-difference SA (FDSA) approach of Kiefer-Wolfowitz with only $1/p$ the total 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 (2.4), 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

- $\hat{L}_k^{(\pm)}$ are estimated values of $L_k(\hat{\theta}_{k-1} \pm c_k\Delta_k)$ using the observed $x_{k+1}^{(\pm)}$ and $u_k^{(\pm)}$, e.g., for $L_k(\theta_k)$ as in (2.2), $\hat{L}_k^{(\pm)} = (x_{k+1}^{(\pm)} - t_{k+1})^T A_k(x_{k+1}^{(\pm)} - t_{k+1}) + u_k^{(\pm)^T} B_k u_k^{(\pm)}$.
- $u_k^{(\pm)}$ are controls based on a NN with weight vector $\theta_k = \hat{\theta}_{k-1} \pm c_k\Delta_k$, where $\Delta_k = (\Delta_{k1}, \Delta_{k2}, \ldots, \Delta_{kp})^T$, 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 $\forall k, i$ (note that Δ_{ki} can *not* be uniformly or normally distributed; it *can* be symmetrically Bernoulli distributed, which we use in Section IV).
- $x_{k+1}^{(\pm)}$ are state values based on $u_k^{(\pm)}$.
- $\{a_k\}$ and $\{c_k\}$ are sequences of positive numbers. In a setting where the system dynamics and loss function are constant (i.e., $\phi_k(\cdot) = \phi(\cdot), \{w_k\}$ i.i.d., and $A_k = A,\ B_k = B, \forall k$), the SA coefficients $a_k,\ c_k$ should satisfy standard conditions (e.g., $a_k \to 0, c_k \to 0, \sum_{k=0}^{\infty} a_k = \infty, \sum_{k=0}^{\infty}(a_k/c_k)^2 < \infty$) given, say, in [23] or [24] to have $\hat{\theta}_k$ converge to the unique θ^* that generates the optimal control function $u_k(\cdot) = u(\cdot)$ (this assumes that the NN structure is rich enough in its ability to replicate the true optimal control function so that such a unique θ^* exists). On the other hand, if the dynamics or loss function are changing, it is best to pick constant coefficients, $a_k = a, c_k = c\ \forall k$, (in order to track the time-varying solution θ_k^*) as discussed in [3], [13], and [14].

The key fact to observe is that at any iteration only *two* state measurements are needed (i.e., the numerators are the same for all p components of $\hat{g}_k(\hat{\theta}_{k-1})$). This is in contrast to the standard FDSA approach where $2p$ measurements are needed to construct the approximation to $g_k(\hat{\theta}_{k-1})$ (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 [23] for general discussion or Bayard [2] for an application of such techniques in adaptive control). 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 IV, although at the cost of additional measurements. 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 [28] for a thorough discussion of smoothing in SPSA-based direct adaptive control). Finally, it should be noted that SPSA is generally guaranteed to yield only a local minimum of the loss function (as is the case with back propagation when that type of algorithm can be applied); extension of the approach to the global optimization setting seems possible using ideas such as in Styblinski and Tang [30], Chin [6] or Yakowitz [33], although these algorithms are naturally more complicated to implement.

B. Implementation Strategies and Use of Prior Information

This subsection presents two ways in which (3.1a) and (3.1b) can be implemented in practical problems. As discussed below, the variations in implementation strategies depend on whether a nominal state is or is not being produced. We also consider a theoretical issue associated with the validity of the SPSA gradient approximation in the control framework here. The subsection closes with a brief discussion on the use of prior information on the system (if available).

[2] Chen [5], Goldenthal and Farrel [9], and Milito *et al.* [16] have also described techniques for NN weight estimation in adaptive control when the gradient is not available, but their techniques still require considerable information about the process dynamics; in particular they require knowledge of the sign of the terms that appear in the gradient $\partial L_k/\partial u_k$ in (2.3). Spall and Cristion [26] includes a more detailed analysis of this type of technique.

Although every iteration of (3.1a) and (3.1b) requires that $x_{k+1}^{(\pm)}$ be produced, it may sometimes be desirable to also produce a "nominal" state x_{k+1} based on a control u_k with $\theta_k = \hat{\theta}_k$. Such a state provides an indicator as to how well the estimation procedure is performing since it relies on the updated weight estimate in the controller (versus perturbed *previous* weight estimates for use in generating $u_k^{(\pm)}$ and hence $x_{k+1}^{(\pm)}$). Thus two possible sequences of states would be

i) $\{x_0, x_1^{(+)}, x_1^{(-)}, x_1, x_2^{(+)}, x_2^{(-)}, x_2 \ldots\}$ (with nominal state).

ii) $\{x_0, x_1^{(+)}, x_1^{(-)}, x_2^{(+)}, x_2^{(-)}, \ldots\}$ (without nominal state).

In many practical applications it would be desirable to adopt a hybrid of i) and ii). For example, in the standard SA setting where $c_k \to 0$, the perturbations $c_k\Delta_k$ would be relatively large during the early iterations, and so it may be desirable to produce x_{k+1} in order to monitor the performance of the estimation algorithm. On the other hand, in the later iterations, when the perturbations are small and $\hat{\theta}_k$ has nearly converged, it may not be necessary to produce the nominal states since the $x_{k+1}^{(\pm)}$ values would (to within the noise level) differ little from x_{k+1}. There may also, of course, be periods when only nominal states might be generated (i.e., no updating of $\hat{\theta}_k$), e.g., when $\hat{\theta}_k$ has effectively converged and the process dynamics are stationary.

The specific method by which (3.1a) and (3.1b) would be implemented depends on the type of system being controlled and whether or not a nominal state is being produced. We outline below a general implementation strategy. This discussion assumes that $M = N = 1$ in (2.1); obvious modifications cover other cases.

In particular, when the nominal state is being generated [i.e., i) above applies], there are three basic operational steps in going from x_k to x_{k+1}.

1) Determine $u_k^{(+)}$ based on $\hat{\theta}_{k-1} + c_k\Delta_k, x_k, u_{k-1}$, and t_{k+1}. Generate a corresponding output $x_{k+1}^{(+)}$ and observed loss $\hat{L}_k^{(+)}$.
2) Given the system at $x_{k+1}^{(+)}$, use $u_k^{(-)}$ based on $\hat{\theta}_{k-1} - c_k\Delta_k, x_{k+1}^{(+)}, u_k^{(+)}$, and t_{k+1}, to generate an output $x_{k+1}^{(-)}$ and observed loss $\hat{L}_k^{(-)}$.
3) Use algorithm (3.1a,b) to generate $\hat{\theta}_k$. Then based on $\hat{\theta}_k, x_{k+1}^{(-)}, u_k^{(-)}$, and t_{k+1}, generate u_k and the corresponding x_{k+1}.[3]

If the nominal state is not being produced (i.e., ii) above applies), then these three steps are modified in an obvious way. (That is, operationally, with the system at state $x_k^{(-)}$, this implementation would then proceed to $x_{k+1}^{(+)}$ by applying $u_k^{(+)}$ based on $\hat{\theta}_{k-1} + c_k\Delta_k, x_k^{(-)}, u_{k-1}^{(-)}$ and t_{k+1}; after observing to $x_{k+1}^{(+)}$, it then generates $x_{k+1}^{(-)}$. Now $\hat{\theta}_{k-1}$ is updated to $\hat{\theta}_k$ by (3.1a,b) and the process is repeated).

[3] If the gradient averaging method mentioned in Section III-A is used, then steps 1–3 are modified in the obvious ways to accommodate the fact that at each iteration, we generate $2q$ (say) values of the control (instead of just two), with each pair of controls based on a new (independent) value of Δ_k.

In light of the constant evolution of the state vector, let us now discuss an issue associated with the theoretical viability of the SPSA-based approach. In particular, since we seek a stochastic version of a gradient descent algorithm, it is important to establish that $\hat{g}_k(\hat{\theta}_{k-1})$ is related to the true gradient $g_k(\hat{\theta}_{k-1})$ in a meaningful way. Under conditions such as those given below (3.1b), Spall [24] establishes the near unbiasedness of the SP gradient estimate (the bias is $O(c_k^2), c_k \to 0$) in a *static optimization* context (i.e., $L_k(\cdot) = L(\cdot)\ \forall k$ with no feedback). Although this paper will not present a formal convergence theory (see Spall and Cristion [29]), there is an intuitive basis for the gradient approximation described above to be appropriate in the *control* context. Namely, if the dynamic system is nearly statistically stationary over the period in which one gradient approxmation is generated, then that gradient approximation will be nearly an unbiased estimate of the true (unknown) gradient. This near-unbiasedness is a factor in the convergence analysis. We use "nearly" here to encompass nonstationarities due to c_k. In particular, if the nonstationarities in going from $\hat{L}_k^{(+)}$ to $\hat{L}_k^{(-)}$ are decreasing in c_k in a manner such that $E(\hat{L}_k^{(+)} - \hat{L}_k^{(-)}) = O(c_k^3)$, then $\hat{g}_k(\hat{\theta}_k)$ will retain the $O(c_k^2)$ bias of the static optimization context. With additional conditions very similar to [24] this is sufficient for convergence of $\hat{\theta}_k$ to the best possible weight vector—when such a weight vector exists—given the chosen NN structure and number of terms within the information set being provided to the controller. In fact, however, convergence can be shown in some cases where the nonstationarities do not decrease in c_k (see [29]).

Let us close this subsection with a brief discussion of how certain types of prior information on the system, if available, can be used in the algorithm to help improve system performance. The most obvious way is through prespecifying certain values of $\hat{\theta}_0$ to produce a desirable value of u_0. For example, if it is known that a nominal value of the control, say u^*, will typically produce a state value relatively close to the target, then certain bias weights on the output layer of the NN (corresponding to certain elements within $\hat{\theta}_0$) could be set to yield a u_0 near u^*. Another way in which prior information can be introduced is through the self-tuning method discussed in Spall and Cristion [26]–[29]. Here the prior information corresponds to *partial* knowledge of the system equation (2.1). The self-tuning method is a slight modification of the approach of this paper in that the NN output is no longer u_k directly; rather it corresponds to an approximation of certain unknown functions within the system equation, which are then used in a control law that is derived from the prior information. Numerical studies in [28] and [29] indicate that the self-tuning method can yield a lower level of tracking error than the baseline method of this paper *if* the required prior information is available and reliable.

IV. Empirical Study: Wastewater Treatment

This section presents the results of a study on a wastewater treatment model from Dochain and Bastin [7]. Similar wastewater treatment models may also be found in the biore-

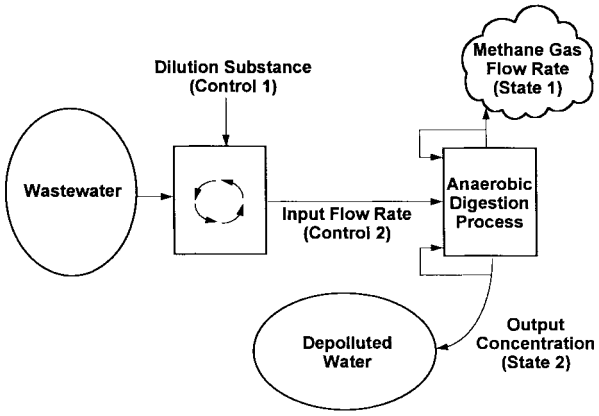

Fig. 2. Wastewater treatment system.

mediation literature (e.g., Andrews [1], Cardello and San [4], Metcalf and Eddy [15], Renard *et al*. [22], and Thibault, Van Breusegem, and Cheruy [31]). This is a model of affine-nonlinear multiplicative control form (e.g., Chen [5]) where the aim is to control the output substrate concentration and the methane production rate based on adjusting the input concentration and the input flow rate at the input to a anaerobic digestion process (see Fig. 2). (Note that [7] controlled only the output substrate concentration *or* the methane production rate using the input flow rate as their only control and that they used an indirect controller, where a general form for the model of the wastewater treatment system was assumed to be known with unknown parameters that were estimated.)

In the wastewater treatment system of Fig. 2, the influent wastewater is first mixed (as determined by a controller) with a dilution substance to provide a mixture with a desired concentration of contaminants. This diluted mixture is then sent to a second tank at a controlled flow rate. In the second tank the mixture goes through an anaerobic digestion process, where the organic material in the mixture is converted by bacteria into depolluted water and byproducts such as methane (Metcalf and Eddy [15], p. 420). Therefore, the system consists of two controls (the mix of wastewater/dilution substance and the input flow rate) and results in two outputs (an effluent depolluted water with a lower concentration of contaminants than the influent and methane gas, which is useful as a fuel [7]). Since this system relies on biological processes, the dynamics are nonlinear and usually time-varying (Thibault, Van Breusegem, and Cheruy [31] and Cardello and San [4]). Also, the system is subject to constraints (e.g., the input and output concentrations, the methane gas flow rate, and the input flow rate all must be greater than zero), which presents an additional challenge in developing a controller for the system.

The study here is based on $A_k = A$ (a constant weighting matrix) and $B_k = 0$ in the loss function (2.2) (i.e., a minimum variance regulator). The performance of the technique will be evaluated by presenting the observed weighted root-mean-square (rms) measurement error, i.e., the observed weighted rms at time k is $[(x_k - t_k)^T A (x_k - t_k)]^{1/2}$, where, for our studies here, we used a two-dimensional diagonal weight matrix A with a value .01 as the first diagonal element and .99 as the second diagonal element (reflecting the relative emphasis to be given to methane production and water purity, respectively). The (feedforward) NN's considered here have nodes that are scaled logistic functions (i.e., $1/(1+e^{-y})$ for input 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 in Chen [5]. 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 $\hat{g}_k(\cdot)$. For the SPSA algorithms we take the perturbations Δ_{ki} to be Bernoulli ± 1 distributed, which satisfies the relevant regularity conditions mentioned in Section III.

The model we used for producing the measurements closely follows that of [7, eqs. (8) and (9)] with the addition of additive (independent) process noise, i.e.,

$$x_{1,k+1} = x_{1,k} + T\mu_k x_{1,k} - T u_{1,k} x_{1,k} + w_{1,k}, \quad \text{(methane gas flow rate)} \quad (4.1a)$$

$$x_{2,k+1} = x_{2,k} - .3636 T x_{1,k} + T u_{1,k}(u_{2,k} - x_{2,k}) + w_{2,k}, \quad \text{(substrate concentration)} \quad (4.1b)$$

$$\mu_k = \frac{(.425 + .025\sin(2\pi k/96))x_{2,k}}{.4 + x_{2,k}}, \quad \text{(bacterial growth function)} \quad (4.1c)$$

where we used $w_k \sim N(0, \sigma^2 I)$ and a sampling period of $T = .5$. (The control algorithm, of course, has no knowledge of (4.1a)–(4.1c).) The bacterial growth function in (4.1c) is in the so-called Monod form, which is the most popular form considered in the literature [7]. For our target sequence t_k we used a periodic square wave, with values $(.97, .13)^T$ for the first 46 updates and $(1, .1)^T$ for the second 46 updates, similar to Fig. 4 in [7]. We modeled the controller using a NN with two hidden layers, one of 20 nodes and one of 10 nodes. The inputs to the controller were the current and most recent state $(M = 2)$, the most recent control $(N = 1)$, and the target vector for the next state, yielding a total of eight input nodes (so an $N_{8,20,10,2}$ network was used for the controller in the notation of [18], which has $180 + 210 + 22 = 412$ weights to be estimated).

Fig. 3 shows the main results for our study of the model in (4.1a)–(4.1c) based on the three-step procedure in Section III-B (including the generation of the optional nominal state for purposes of plotting the weighted rms error). The rms error curves in the figure 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. The value x_0 was set to $(.5, 1.6375)^T$ and $\sigma = .01$, so the initial weighted rms error is 1.5 and the minimum achievable long-run weighted rms error is .01. To further smooth the resulting error curves and to show typical performance (not just case-dependent variation), we applied an expanding window smoother (which allows for rapid changes in the early iterations and little change in later iterations) to the error values based on the average of ten runs. The

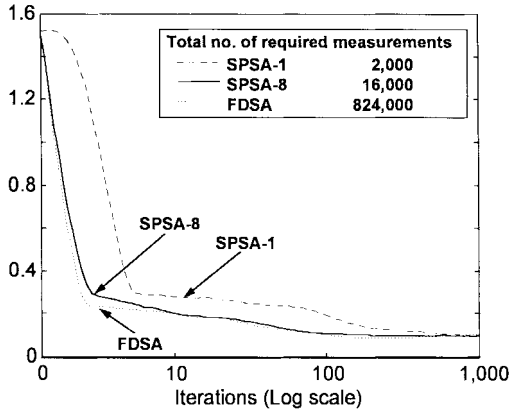

Fig. 3. Weighted rms error for SPSA and FDSA algorithms (minimum achievable long-run weighted rms $= 0.01$).

curves shown in the figure are based on this combination of across-realization averaging and across-iteration smoothing.

For this study, we used constant SA gains of the form $a_k = a$ and $c_k = c$ with $a, c > 0$ (constant gains were used to accommodate the time-varying nature of the system as discussed in Section III). We attempted to tune a and c to maximize the rate of decay in weighted RMS error (as would typically be done in practice); the values satisfied $a = .1$ and $.05 \leq c \leq .1$. 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).

Fig. 3 shows that both the SPSA and FDSA algorithms yield controllers with decreasing rms tracking error over time. We see that the long-run performance of SPSA-8 and FDSA is virtually identical and SPSA-1 performs almost as well as both FDSA and SPSA-8. 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-8 required only two measurements and 16 measurements, respectively, while each iteration of FDSA needed 824 measurements. Hence Fig. 3 illustrates that SPSA-8 yielded the same level of tracking error as the standard FDSA algorithm with a 51-fold savings in state measurements and the performance of SPSA-1 was comparable to FDSA with a 412-fold savings in state measurements. The data savings seen in Fig. 3 is typical of that for a number of other studies involving SPSA and FDSA that we have conducted on model (4.1a)–(4.1c) as well as on other nonlinear models (see e.g., [26] or [29]); in fact even greater data savings are typical with more complex NN's (as might be needed in higher-dimensional systems). Note also that the curves in Fig. 3 have the typical shape of many optimization algorithms in that there is a sharp initial decline followed by a slow decline. Hence over 80% of the possible reduction in RMS error, which may be all that is required in some applications, is achieved by all three algorithms with fewer than ten iterations.

V. Concluding Remarks

This paper has discussed an approach for direct adaptive control that does not require a model (NN or otherwise) for the process dynamics. In particular, the method here addresses the shortcoming noted in Narendra and Pathasarathy [18, p. 19] that "At present, methods for directly adjusting the control parameters based on the output error (between the plant and reference [target] outputs) are not available." The approach applies under general conditions in control problems with a one-step-ahead loss function. (Although this paper considers the case of direct state measurements, extensions to indirect measurements are possible as in [29].) Perhaps the main restriction is that the system dynamics be approximately stationary while an individual SPSA gradient approximation is being generated for one iteration of the estimation algorithm (the dynamics can be nonstationary over longer time intervals). A further restriction is one common to perhaps all control techniques that rely on imprecise *a priori* system knowledge: namely, that the system be able to tolerate suboptimal controls as the learning process takes place.

The approach was illustrated on a simulated nonstationary wastewater treatment plant where the control objective was to achieve time-varying target values for both clean water and methane gas production. It was found that effective control was achieved in this challenging problem at much lower cost than the more standard finite-difference-type method. There remain, however, several areas for further study that could enhance the approach. These include a careful stability/controllability analysis and the implementation of an accelerated (say, second order) stochastic approximation technique (see Spall [25]). Nevertheless, the approach as it currently stands has broad applicability when little is known about the equations governing the system.

References

[1] J. F. Andrews, "Dynamic models and control strategies for wastewater treatment processes," *Water Res.*, vol. 8, pp. 261–289, 1974.
[2] D. S. Bayard, "A forward method for optimal stochastic nonlinear and adaptive control," *IEEE Trans. Automat. Contr.*, vol. 36, pp. 1046–1053, 1991.
[3] A. Benveniste and G. Ruget, "A measure of the tracking capability of recursive stochastic algorithms with constant gains," *IEEE Trans. Automat. Contr.*, vol. AC-27, pp. 639–649, 1982.
[4] R. J. Cardello and K.-Y. San, "The design of controllers for batch bioreactors," *Biotech. Bioeng.*, vol. 32, pp. 519–526, 1988.
[5] F. C. Chen, "Back-propagation neural networks for nonlinear self-tuning adaptive control," *IEEE Control Syst. Mag.*, pp. 44–48, Apr. 1990.
[6] D. C. Chin, "A more efficient global optimization algorithm based on Styblinski and Tang," *Neural Networks*, vol. 7, pp. 573–574, 1994.
[7] D. Dochain and G. Bastin, "Adaptive identification and control algorithms for nonlinear bacterial growth systems," *Automatica*, vol. 20, pp. 621–634, 1984.
[8] K. I. Funahashi, "On the approximate realization of continuous mappings by neural network," *Neural Networks*, vol. 2, pp. 183–192, 1989.
[9] W. Goldenthal and J. Farrell, "Application of neural networks to automatic control," in *Proc. AIAA Guidance, Navigation and Control Conf.*, 1990, pt. 2, pp. 1108–1112.
[10] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Networks*, vol. 2, pp. 359–366, 1989.
[11] K. J. Hunt and P. Sbarbaro, "Neural networks for nonlinear model control," *IEE (Great Britain) Proc.-D*, vol. 138, pp. 431–438, 1991.

[12] Y. Iiguni, H. Sakai, and H. Tokumaru, "A nonlinear regulator design in the presence of system uncertainties using multilayered neural networks," *IEEE Trans. Neural Networks*, vol. 2, pp. 410–417, 1991.
[13] C. M. Kuan and K. Hornik, "Convergence of learning algorithms with constant learning rates," *IEEE Trans. Neural Networks*, vol. 2, pp. 484–489, 1991.
[14] H. J. Kushner and H. Huang, "Asymptotic properties of stochastic approximations with constant coefficients," *SIAM J. Contr. Optim.*, vol. 19, pp. 87–105.
[15] Metcalf and Eddy, Inc., *Wastewater Engineering: Treatment, Disposal, and Reuse*, 3rd ed. New York: McGraw-Hill, 1972.
[16] R. A. Milito, I. Guyon, and S. A. Solla, "Neural network implementation of admission control," in *Advances in Neural Information Processing Systems 3*. San Mateo, CA: Morgan-Kaufmann, 1991.
[17] P. E. Moden and T. Soderstrom, "Stationary performance of linear stochastic systems under single step optimal control," *IEEE Trans. Automat. Contr.*, vol. AC-27, pp. 214–216, 1982.
[18] K. S. Narendra and K. Parthasarathy, "Identification and control of dynamical systems using neural networks," *IEEE Trans. Neural Networks*, vol. 1, pp. 4–26, 1990.
[19] ———, "Gradient methods for the optimization of dynamic systems containing neural networks," *IEEE Trans. Neural Networks*, vol. 2, pp. 252–262, 1991.
[20] Y.-M. Pao, S. M. Phillips, and D. J. Sobajic, "Neural-net computing and the intelligent control of systems," *Int. J. Control*, vol. 56, pp. 263–289, 1992.
[21] K. M. Passino, "Bridging the gap between conventional and intelligent control," *IEEE Control Syst. Mag.*, vol. 13, pp. 12–18, June 1993.
[22] P. Renard, V. Van Breusegem, M.-T. Nguyen, H. Naveau, and E.-J. Nyns, "Implementation of an adaptive controller for the startup and steady-state running of a biomethanation process operated in the CSTR mode," *Biotech. Bioeng.*, vol. 38, pp. 805–812, 1991.
[23] D. Ruppert, "Kiefer–Wolfowitz procedure," in *Encyclopedia of Statistical Science*, S. Kotz and N. L. Johnson, Eds. New York: Wiley, 1983, vol. 4, pp. 379–381.
[24] J. C. Spall, "Multivariate stochastic approximation using a simultaneous perturbation gradient approximation," *IEEE Trans. Automat. Contr.*, vol. 37, pp. 332–341, 1992.
[25] ———, "Stochastic version of second-order optimization using only function measurements," in *Proc. 13th IFAC World Congr.*, 1996.
[26] J. C. Spall and J. A. Cristion, "Neural networks for control of uncertain systems," in *Proc. Test Technology Symp. IV*, sponsored by U.S. Army Test and Evaluation Command, 1991, pp. 575–588.
[27] ———, "Direct adaptive control of nonlinear systems using neural networks and stochastic approximation," in *Proc. IEEE Conf. Dec. Control*, 1992, pp. 878–883.
[28] ———, "Nonlinear adaptive control using neural networks: Estimation based on a smoothed form of simultaneous perturbation gradient approximation," *Statistica Sinica*, vol. 4, pp, 1–27, 1994.
[29] ———, "Model-free control of nonlinear stochastic systems in discrete-time," *IEEE Trans. Automat. Contr.*, submitted for publication. Condensed version in *Proc. IEEE Conf. Dec. Control*, 1995, pp. 2199–2204.
[30] M. A. Styblinski and T. S. Tang, "Experiments in nonconvex optimization: Stochastic approximation with function smoothing and simulated annealing," *Neural Networks*, vol. 3, pp. 467–483, 1990.
[31] J. Thibault, V. Van Breusegem, and A. Cheruy, "On-line prediction of fermentation variables using neural networks," *Biotech. Bioeng.*, vol. 36, pp. 1041–1048, 1990.
[32] E. Tulunay, "Introduction to neural networks and their application to process control," *Neural Networks: Advances and Applications*, E. Gelene, Ed. Amsterdam, The Netherlands, Elsevier, 1991.
[33] S. Yakowitz, "A globally convergent stochastic approximation," *SIAM J. Contr. Optim.*, vol. 31, pp. 30–40, 1993.
[34] T. Yamada and T. Yabuta, "Dynamic system identification using neural networks," *IEEE Trans. Syst., Man, Cybern.*, vol. 23, pp. 204–211, 1993.

James C. Spall (S'82–M'83–SM'90) received the B.S. degree in systems engineering in 1979 from Oakland University, Rochester, MI, the S.M. degree in technology and policy in 1981 from the Massachusetts Institute of Technology, Cambridge, and the Ph.D. in systems engineering in 1983 from the University of Virginia, Charlottesville.

Since 1983, he has been with the Applied Physics Laboratory, The Johns Hopkins University, Laurel, MD, where he leads several projects in the areas of statistical modeling and control. In 1991, he was appointed to the Principal Professional Staff of the Laboratory. He has published many papers in the areas of statistics and control, including articles on subjects such as time series, optimization, parameter identification, and neural networks.

Dr. Spall is an Associate Editor of the IEEE TRANSACTIONS ON AUTOMATIC CONTROL, is a contributing editor for the *Current Index to Statistics*, and served as editor and coauthor of the book *Bayesian Analysis of Time Series and Dynamic Models* (1988). He is a member of the American Statistical Association and Sigma Xi, and a Fellow of the engineering honor society Tau Beta Pi. In 1990, he received the R. W. Hart Prize as principal investigator of the most outstanding Independent Research and Development project at JHU/APL.

John A. Cristion (S'85–M'86) received the B.S. degree in electrical engineering from Drexel University, Philadelphia, PA, in 1986, and the M.S. degree in electrical engineering from The Johns Hopkins University, Laurel, MD, in 1991.

Since 1986, he has been with the Applied Physics Laboratory, The Johns Hopkins University, where his current research interests include sonar signal processing, biomedical engineering, estimation theory, statistical modeling, and neural networks. He has published papers in the areas of control and biomedical engineering, including articles on subjects such as epileptic seizure detection, polygraph analysis, optimization, parameter estimation, and neural networks.

Mr. Cristion is a member of Tau Beta Pi and Eta Kappa Nu.