International Journal of Systems Science
volume 34, numbers 14–15, 15 November–15 December 2003, pages 851–863

Application of stochastic approximation techniques in neural modelling and control

A. VANDE WOUWER†*, C. RENOTTE† and M. REMY†

Learning, i.e. estimation of weights and biases in neural networks, involves the minimization of an output error criterion, a problem which is usually solved using back-propagation algorithms. This paper aims to assess the potential of simultaneous perturbation stochastic approximation (SPSA) algorithms to handle this minimization problem. In particular, a variation of the first-order SPSA algorithm that makes use of several numerical artifices including adaptive gain sequences, gradient smoothing and a step rejection procedure is developed. For illustration purposes, several application examples in the identification and control of nonlinear dynamic systems are presented. This numerical evaluation includes the development of neural network models as well as the design of a model-based predictive neural PID controller.

1. Introduction

Over the past several years, neural networks (NNs) have been increasingly applied to the identification and control of nonlinear systems (e.g. Hunt *et al*. 1992, Suykens *et al*. 1996).

A basic model structure for static nonlinearities is the multilayer feedforward NN, in which learning, i.e. estimation of weights and biases, involves the minimization of an output error criterion $J(\theta)$ using back-propagation (BP) (Rumelhart *et al*. 1986), an analytical procedure in which the error evaluated at the output layer is propagated back through the hidden layers and the input layer.

Although the BP method can be generalized for more complex NN structures (e.g. recurrent NNs, hybrid physical–neural models), which are useful in modelling dynamic nonlinear systems, the resulting algorithms are usually more complicated to implement and more computationally demanding, e.g. dynamic back-propagation (Narendra and Parthasarathy 1990, 1991) and back-propagation through time (Werbos 1990). Hence, it is appealing to develop a more straightforward numerical procedure for computing the gradient of the output error criterion. However, as NNs usually involve a large number of unknown parameters, the evaluation of the criterion gradient by varying the parameters one at a time, as it is required in conventional finite difference approximations, would be extremely costly.

In contrast to standard finite differences, the simultaneous perturbation (SP) approximation of the gradient proposed by Spall (1992) makes use of a very efficient technique based on a simultaneous (random) perturbation in all the parameters and requires only two evaluations of the criterion. This approach has first been applied to gradient estimation in a first-order stochastic approximation (SA) algorithm (Spall 1992), and more recently to Hessian estimation in an accelerated second-order SPSA algorithm (Spall 2000).

These algorithms seem particularly well suited to the NN learning problem, and in this connection, this study aims at assessing the potential of the above-mentioned first- and second-order algorithms (1SPSA and 2SPSA) based on several representative application examples. From a practical point of view, the important issues of convergence, accuracy, computational load, ease of use (tuning) and simplicity of implementation are discussed.

Efficiency, simplicity of implementation and very modest computational costs make 1SPSA particularly attractive, even though it suffers from the classical drawback of first-order algorithms, i.e. a slowing down in the convergence as an optimum is approached. In this study, a variation of this first-order algorithm

Received 10 December 2002. Revised 19 April 2003.

† Service d'Automatique, Faculté Polytechnique de Mons, 31 Boulevard Dolez, B-7000 Mons, Belgium.
*To whom correspondence should be addressed.
e-mail: Alain.VandeWouwer@.fpms.ac.be

is considered that makes use of adaptive gain sequences, gradient smoothing and a step rejection procedure, to enhance convergence and stability. To demonstrate the algorithm efficiency and versatility, attention is focused on a realistic application example, e.g. the development of a predictive control scheme for a two-tank cooling system.

This predictive controller is based on three main components:

- Process emulator in the form of a neural state space model (Suykens *et al.* 1996), which generates prediction of the future process outputs over a specified horizon.
- NN controller, with a PID-like input–output parametrization (Tan 1993).
- Optimization procedure to train the NN model (off-line) and the NN controller (on-line), i.e. our modified 1SPSA algorithm.

The paper is organized as follows. Section 2 introduces the basic principle of the first- and second-order SPSA algorithms. In Section 3, the performances of the several algorithms are compared based on a simple application example, i.e. the training of a series-parallel NN used for modelling a nonlinear dynamic system. On this basis, 1SPSA is selected for its simplicity and efficiency, and a variation of this first-order algorithm, including adaptive gain sequences, gradient smoothing and a step rejection procedure, is presented in Section 4. The performance improvement brought by the proposed numerical artifices is illustrated with another simple example, i.e. the training of a dynamic multilayer perceptron (DMLP) (Ayoubi 1996) used for modelling a nonlinear dynamic system. In Section 5, a more challenging application, i.e. the problem of controlling the output temperature of sulfuric acid in a two-tank cooling system, is studied in some detail. The performance of this control scheme in the face of non-measurable disturbances in the acid inlet temperature and noisy output acid temperature measurements is investigated. Finally, Section 6 has concluding remarks.

2. SA algorithms

Consider the problem of minimizing a possibly noisy, objective function $J(\theta)$ with respect to a vector θ of unknown parameters (in this study, the weights and biases of a NN).

$$\min_{\theta} \frac{1}{N} \sum_{i=1}^{N} \left(y_i - \hat{y}_i(\theta) \right)^2, \tag{1}$$

where y_i is the real-system output and $\hat{y}_i$ is the output produced by the NN depending on the parameters θ. In this work, the minimization problem (1) is handled using several SA algorithms, which have been implemented in MATLAB.m files.

The first-order 1SPSA algorithm is given by the following core recursion for the parameter vector θ (Spall 1992):

$$\hat{\theta}_{k+1} = \hat{\theta}_k - a_k\, \hat{g}_k(\hat{\theta}_k), \tag{2}$$

where a_k is a positive scalar gain coefficient satisfying certain conditions and $\hat{g}_k(\hat{\theta}_k)$ is an approximation of the criterion gradient obtained by varying all the elements of $\hat{\theta}_k$ simultaneously, i.e.

$$\hat{g}(\theta_k) = \begin{bmatrix} \dfrac{J(\hat{\theta}_k + c_k\Delta_k) - J(\hat{\theta}_k - c_k\Delta_k)}{2c_k\Delta_{k1}} \\ \cdots \\ \dfrac{J(\hat{\theta}_k + c_k\Delta_k) - J(\hat{\theta}_k - c_k\Delta_k)}{2c_k\Delta_{kp}} \end{bmatrix}, \tag{3}$$

where c_k is a positive scalar and Δ_k is a user-generated zero-mean random vector satisfying certain regularity conditions (typically, $\Delta_k = (\Delta_{k1}, \Delta_{k2}, \ldots, \Delta_{kp})^T$ with symmetrically Bernouilli distributed random variables $\{\Delta_{ki}\}$). The above-mentioned conditions on the gain sequence a_k and on the random vector Δ_k are imposed to ensure asymptotic convergence of the algorithm. These conditions are detailed in Spall (1992).

It is important to note that this gradient estimate differs from usual finite difference approximations in that the number of criterion evaluations is not proportional to the number of unknown parameters. Instead, only two evaluations of the criterion are required.

The recursion (2) can also be based on a smoothed gradient approximation (Spall and Cristion 1994)

$$G_k = \rho_k G_{k-1} + (1 - \rho_k)\hat{g}_k(\hat{\theta}_k), \quad G_0 = 0, \tag{4}$$

where $\hat{g}_k(\hat{\theta}_k)$ is the simultaneous perturbation gradient estimate (3) and $0 \le \rho_k \le 1$. When the smoothed gradient approximation G_k is used in (2) instead of $\hat{g}_k(\hat{\theta}_k)$, the procedure, which is analogous to the momentum approach in BP, is denoted 1SPSA-GS (where the additional letters 'GS' stand for gradient smoothing).

Alternatively, if direct evaluations of the criterion gradient are available (usually with some added noise), recursion (2) defines a stochastic gradient (SG) algorithm, denoted 1SGSA in the following.

The second-order algorithms 2SPSA (and 2SGSA) are based on the following two core recursions, one for the parameter vector θ, the second for the Hessian $H(\theta)$ of the criterion (Spall 2000):

$$\hat{\theta}_{k+1} = \hat{\theta}_k - a_k \overline{\overline{H}}_k^{-1} \hat{g}_k(\hat{\theta}_k), \quad \overline{\overline{H}}_k = f_k(\overline{H}_k) \tag{5}$$

$$\overline{H}_k = \frac{k}{k+1}\overline{H}_{k-1} + \frac{1}{k+1}\hat{H}_k, \tag{6}$$

where $\hat{H}_k$ is a per-iteration estimate of the Hessian matrix, which is computed from gradient approximations (or direct evaluations) using a simultaneous perturbation approach, $\overline{H}_k$ is a simple sample mean, and f_k is a mapping designed to cope with possible non-positive definiteness of $\overline{H}_k$.

Again, the algorithm requires only a small number of function evaluations—at least four criterion evaluations to construct the gradient and Hessian estimates, or three gradient evaluations in the SG case—independent of the number of unknown parameters.

Important implementation issues of these SA algorithms include initialization, choice of the gain sequences $\{a_k\}$ and $\{c_k\}$ (usually these sequences are chosen in the form $a_k = a(A+k+1)^{-\alpha}$ and $c_k = c(k+1)^{-\gamma}$), gradient/Hessian averaging and step rejection (Spall 2000).

Usually, 1SPSA appears as an efficient, robust algorithm, but suffers from the classical drawback of first-order algorithms, i.e. a slowing down in the convergence as an optimum is approached. 2SPSA includes second-order effects with the aim of accelerating convergence. However, tuning of the several algorithm coefficients is a more delicate task, particularly in noisy environment.

3. Numerical evaluation of the SA algorithms

The above-mentioned algorithms are applied to parameter estimation in an NN in series-parallel mode (or feedforward time delay NN) used as a prediction model for a process given by (this particular example was originally considered in Narendra and Parthasarathy 1990):

$$y(k) = \frac{y(k-1)y(k-2)y(k-3)u(k-2)(y(k-3)-1)+u(k-1)}{1+y(k-2)^2+y(k-3)^2}, \tag{7}$$

where u(k) and y(k) are the input and output sequences, respectively.

Obviously, this simple example does not require any special learning procedure since the application of BP is straightforward. Hence, first- and second-order methods as implemented in the MATLAB NN Toolbox could certainly be used. Our objective at this stage is to assess the relative performance of the SA algorithms presented in Section 2.

The training set consists of 997 data produced by (7) using an input sine wave with increasing frequency (0.1–10 Hz). Test sets used for model validation are produced with other input signals, e.g. a switchback signal. Based on the assumption that the correct system orders are known, the process is modelled using a NN with five inputs $[y(k-1), y(k-2), y(k-3), u(k-1), u(k-2)]$, one hidden layer and one output. To select the number of nodes in the hidden layer n_{hl} and to define 'reference' results, a classical Levenberg–Marquardt (LM) algorithm (Reklaitis *et al.* 1983) is first used to estimate the parameter sets corresponding to NNs with n_{hl} ranging from 1 to 15. As a result, it is observed that the quadratic error criterion is monotonically decreasing. Compromising model accuracy and over-parametrization (a measure of this compromise is given by Akaike's criterion in Ljung 1987), a NN model with $n_{hl} = 7$ is selected.

In all these cases (n_{hl} ranging from 1 to 15), 1SPSA, 1SPSA-GS (with gradient smoothing) and 2SPSA perform successfully. Generally, the SPSA algorithms can be rated as follows with regards to speed of convergence and accuracy: (1) 2SPSA, (2) 1SPSA-GS and (3) 1SPSA. In the case of a NN with $n_{hl} = 7$ and a minimization of the error criterion carried out with 2SPSA, direct and cross-validation results are illustrated in figure 1.

However, as the number of nodes in the hidden layer increases from 1 to 15, the results obtained with the three SPSA algorithms become almost the same, i.e. the accuracy of the results produced by 1SPSA and 1SPSA-GS gradually improves, the improvement being more noticeable for 1SPSA, while the accuracy of the results produced by 2SPSA, which initially also improves, slightly deteriorates for larger values of n_{hl}. We can only conjecture about this last observation, which could be interpreted as a deterioration of the Hessian estimate due to NN over-parametrization.

Convergence and accuracy are usually better and less sensitive to the NN structure when the gradient information, which is readily available in this example, is used in the stochastic gradient algorithms 1SGSA or 2SGSA. Our numerical experiments show that 1SGSA produces results equivalent to standard BP, while 2SGSA slightly supersedes BP with momentum and adaptive learning rate, as implemented in the MATLAB NN Toolbox. As compared with SP algorithms, 1SGSA and 2SPSA display similar performance.

From a computational point of view, the estimation and regularization of the Hessian estimate used in 2SPSA can become very costly for large numbers of parameters. In our application, the CPU time required by 2SPSA ranges from 5 to 10 times the CPU required by 1SPSA, so that, in terms of efficiency, the use of 2SPSA might be questionable.

An attempt is therefore made to reduce the computational expense by evaluating only a diagonal estimate of the Hessian matrix. Indeed, a reduction of about 40% in the computation time is observed, due to savings in the evaluation of the Hessian estimate, as well as in the recursion on θ that only requires a trivial matrix inverse. The performance, in terms of rate of convergence

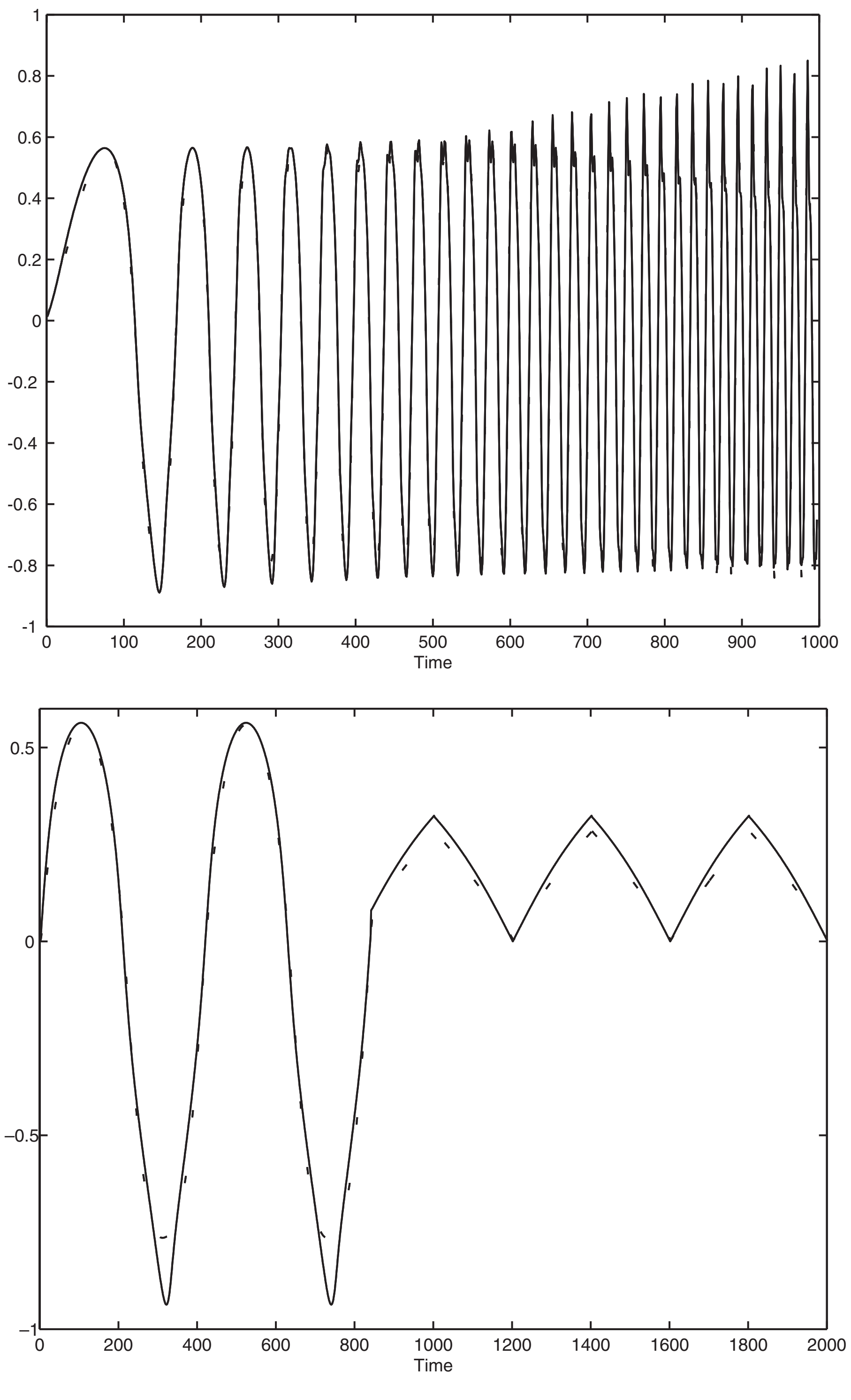

Figure 1. Identification of an NN in a series-parallel mode—direct (above) and cross-validation (below) results. Solid line, real system output; dotted line, NN output.

and accuracy, remains almost unchanged, which demonstrates that the diagonal Hessian estimate still captures potential large scaling differences in the elements of θ. This algorithm variation is denoted 2SPSA-DH (where 'DH' is Diagonal Hessian Estimate). In this latter algorithm, regularization can be achieved in a straightforward way, by imposing the positiveness of the diagonal elements of the Hessian matrix.

Note that Zhu and Spall (2002) have recently developed a more elaborate version of 2SPSA, in which the mapping f_k in (5), which eliminates the non-positive definiteness of the Hessian estimate, preserves key spectral properties (the diagonal eigenvalue matrix Λ_k of $\overline{H}_k$ is first 'corrected' so as to eliminate negative elements, resulting in a new $\hat{\Lambda}_k$ matrix, and the orthogonal matrix P_k of eigenvectors is used to define the mapping $f_k(\overline{H}_k) = P_k \hat{\Lambda}_k P_k^T$). This latter developments are however beyond the scope of this study, and will not be pursued.

Table 1 compares, for 10 independent runs starting from the same initial parameter estimates θ_0, the results obtained with each algorithm in the case of a NN with $n_{hl} = 7$. It gives the minimum (best), maximum (worst) and average values of the mean-square error criterion after 3000 iterations. It is apparent that algorithms based on an SP approximation of the gradient and/or the Hessian produce more 'dispersed' results, i.e. display a larger deviation between the worst and best cases than the SG(BP) algorithms. Figure 2 shows the average mean-square error curves for the SP algorithms. Note that even though the average performance of 1SPSA and 1SPSA-GS are virtually identical, the results produced by 1SPSA are more dispersed from one run to another (table 1).

Table 1. Mean-square errors obtained after 3000 iterations (10 independent runs—NN with $n_{hl} = 7$).

	Best RMS	Worst RMS	Average RMS
1SPSA	0.0051	0.023	0.012
1SPSA-GS	0.011	0.019	0.013
2SPSA	0.0015	0.0030	0.0021
2SPSA-DH	0.0012	0.0067	0.0025
1SGSA	0.0016	0.0016	0.0016
2SGSA	0.000 092	0.000 30	0.000 14
BP	0.0016	0.0016	0.0016
Adaptive BP-GS	0.000 23	0.000 23	0.000 23
LM	0.000 000 24	0.000 000 24	0.000 000 24

4. Modified first-order SP algorithm

Efficiency, simplicity of implementation and very modest computational costs make 1SPSA particularly attractive. In this section, a variation of this first-order algorithm is considered, which makes use of adaptive gain sequences, gradient smoothing and a step rejection procedure, to enhance convergence and stability.

4.1. *Some numerical artifices*

In its original formulation (Spall 1992), 1SPSA makes use of decaying gain sequences $\{a_k\}$ and $\{c_k\}$ in the form:

$$a_k = \frac{a}{(A + k + 1)^{\alpha}}, \qquad c_k = \frac{c}{(k + 1)^{\gamma}}. \tag{8}$$

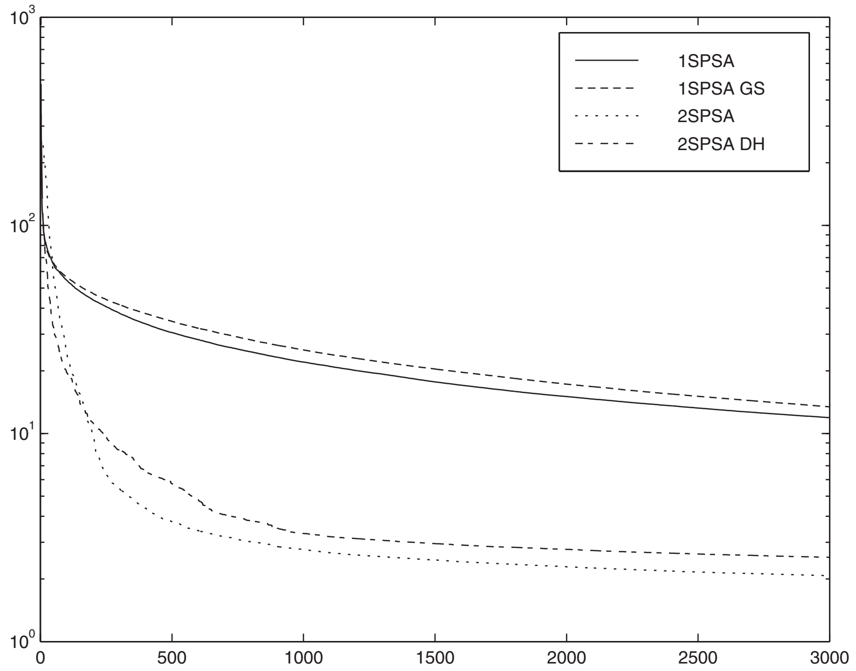

Figure 2. Identification of an NN in a series-parallel mode—average mean-square error curves for the SP-algorithms (NN with $n_{hl} = 7$).

Although the choice of these sequences ensures asymptotic convergence in the case of a convex optimization problem, numerical experiments show that the algorithm can get stuck somewhere in the parameter space if the criterion value becomes significantly worse (due to a poor current gradient approximation, a non-convex optimization problem as it is often the case in NN identification problems, etc.) and the gain sequences are then too small to recover from this situation.

To enhance convergence, the use of an adaptive gain sequence for parameter updating is considered

$$\left.\begin{aligned} a_k &= \eta a_{k-1}, \quad \eta \geq 1, \ \text{if } J(\theta_k) < J(\theta_{k-1}) \\ a_k &= \mu a_{k-1}, \quad \mu \leq 1, \ \text{if } J(\theta_k) \geq J(\theta_{k-1}), \end{aligned}\right\} \tag{9}$$

where, typically, $\eta = 1.05$ and $\mu = 0.7$.

In comparison with the original 1SPSA algorithm, which requires two criterion evaluations $J(\hat{\theta}_k + c_k\Delta_k)$ and $J(\hat{\theta}_k - c_k\Delta_k)$, one extra criterion evaluation $J(\hat{\theta}_k)$ is required to implement (9). This represents a very reasonable overhead compared with the cost of a conventional finite difference approximation of the gradient, where all the parameters have to be varied one at a time. If computational expense is critical, a one-sided approximation of the gradient could be computed based on $J(\hat{\theta}_k)$ and $J(\hat{\theta}_k + c_k\Delta_k)$ (or $J(\hat{\theta}_k - c_k\Delta_k)$) rather than a two-sided (centered) approximation as in (3). However, this approximation would be less accurate, and when computation time allows it, three criterion evaluations are recommended to evaluate (3) and (9) independently.

In addition to gain attenuation, when the value of the criterion becomes worse, 'blocking' mechanisms are also applied, i.e. if $J(\theta_k) \geq J(\theta_{k-1})$:

- Current step is rejected, i.e. θ_k is disregarded.
- Updating gain is reduced according to (9).
- New step is accomplished starting from the previous parameter estimate θ_{k-1} (i.e. the parameter estimate obtained at the previous optimization step is stored in an intermediate workspace for further use).
- New criterion value is checked and the procedure is repeated if required.

A constant gain sequence $c_k = c$ is used for gradient approximation, the value of c being selected so as to overcome the influence of (numerical or experimental) noise. Indeed, it is necessary to select a sufficiently large perturbation $c_k\Delta_k$ of the parameter estimate θ_k in order to effect a significant change in the criterion, from $J(\hat{\theta}_k)$ to $J(\hat{\theta}_k + c_k\Delta_k)$ (or $J(\hat{\theta}_k - c_k\Delta_k)$). Otherwise, the computed differences could be just numerical noise, so as the search direction of the algorithm.

Finally, a gradient smoothing (GS) procedure is implemented, i.e. gradient approximations are averaged across iterations in the following way

$$G_k = \rho_k G_{k-1} + (1-\rho_k)\hat{g}_k(\hat{\theta}_{k-1}), \quad 0 \leq \rho_k \leq 1, \ G_0 = 0, \tag{10}$$

where, starting with a typical $\rho = 0.95$, ρ_k is decreased in a way similar to (9) when step rejection occurs (i.e. $\rho_k = \mu\rho_{k-1}$ with $\mu \leq 1$) and is reset to its initial value after a successful step.

As the following simple numerical example illustrates, the use of these numerical artifices, i.e. adaptive gain sequences, step rejection procedure and gradient smoothing, significantly improves the effective practical performance of the algorithm (which, in the following, is denoted 'adaptive 1SP-GS').

4.2. *Preliminary test*

Consider the problem of modelling a nonlinear process given by:

$$y(k) = \frac{0.875y(k-1) + u(k-1)}{1 + y^2(k-1)} \tag{11}$$

using a dynamic multilayer perceptron (DMLP) (Ayoubi 1996) with one input, four nodes in the hidden layer and one output. DMLPs incorporate dynamic linear elements within each elementary processing units and so allow nonlinear dynamic systems to be modelled without the need for global recursion. As the number of unknown parameters grows rapidly with the NN input dimension (and so, in a feedforward time delay NN, with the number of delayed inputs), DMLPs will be particularly advantageous in reducing the size of the unknown parameter set. In this application example, the hidden and output nodes are associated with second-order dynamic elements, so that there are $n_p = 33$ unknown parameters to estimate.

Table 2 compares, in terms of number of iterations, computational load (normalized CPU with the CPU required by 1SPSA as reference) and mean square error (RMS), the performance of the following:

- Original 1SPSA algorithm (equations (2) and (3) with a step rejection procedure).
- Adaptive 1SP-GS (equations (2), (3), and (8)–(10)).

Table 2. Computational statistics.

	Iterations	CPU	RMS
1SPSA	8000	1	0.003 26–0.004 31
Adaptive 1SP-GS	8000	1.3	0.003 09
Adaptive 1GBP-GS	600	7.2	0.003 10

- Adaptive 1GBP-GS (the same as above with a gradient evaluated analytically using generalized back-propagation).

Clearly, generalized back-propagation (GBP) is very efficient in terms of the number of iterations required to achieved a certain level of accuracy, but it is computationally expensive as it requires the solution of dynamic sensitivity equations at each iteration. On the other hand, 1SPSA has very modest computational requirements, but produces relatively dispersed results (0.00326–0.00431 represents the range of values obtained from 10 independent runs starting from the same initial parameter estimates). The main advantage of our algorithm is that it retains the very modest computational requirement of 1SPSA and usually provides less dispersed, more accurate results.

5. More realistic application

In the remainder of this study, the usefulness of the adaptive 1SP-GS algorithm is illustrated with a more realistic application, i.e. neural modelling and control of a two-tank system used to cool sulfuric acid with a countercurrent water stream (Jenson and Jeffreys 1977) (figure 3).

A nonlinear mathematical model of the cooling system is used for producing simulated data. This model can be derived by expressing the energy balance on each tank, i.e.

$$\left.\begin{aligned} M_{T,1}C_{pa}\dot{T}_{a,1} &= \dot{M}_w C_{pw}(T_{w,2}-T_{w,1}) + \dot{M}_a C_{pa}(T_{a,in}-T_{a,1}) \\ M_{T,2}C_{pa}\dot{T}_{a,2} &= \dot{M}_w C_{pw}(T_{w,in}-T_{w,2}) + \dot{M}_a C_{pa}(T_{a,1}-T_{a,2}), \end{aligned}\right\} \tag{12}$$

where $M_{T,i}$ is the weight of acid in tank i ($i=1,2$), C_{pa} (C_{pw}) is the acid (water) specific heat, $\dot{M}_a$ ($\dot{M}_w$) is the acid (water) mass flow rate, and $T_{a,i}$ ($T_{w,i}$) is the acid (water) temperature from tank i.

Heat transfer is modelled through a log-mean delta T, i.e.

$$\left.\begin{aligned} \dot{M}_w C_{pw}(T_{w,1}-T_{w,2}) &= k_1 A_1 \frac{(T_{a,1}-T_{w,1})-(T_{a,1}-T_{w,2})}{\ln((T_{a,1}-T_{w,1})/(T_{a,1}-T_{w,2}))} \\ \dot{M}_w C_{pw}(T_{w,2}-T_{w,in}) &= k_2 A_2 \frac{(T_{a,2}-T_{w,2})-(T_{a,2}-T_{w,in})}{\ln((T_{a,2}-T_{w,2})/(T_{a,2}-T_{w,in}))}, \end{aligned}\right\} \tag{13}$$

where k_i is the heat transfer coefficient in tank i and A_i is the coil heat transfer area in tank i.

Parameter values and steady-state operating conditions are listed in table 3. As sulfuric acid is assumed to come from an upstream unit, the feed temperature $T_{a,in}$ varies and is considered as a non-measured disturbance.

5.1. *NN modelling*

A neural state space model, as introduced by Suykens *et al.* (1996), is selected:

$$\left.\begin{aligned} \hat{x}_{k+1} &= W_{AB}\tanh(V_A\hat{x}_k + V_B u_k + \beta_{AB}) \\ \hat{y}_k &= W_{CD}\tanh(V_C\hat{x}_k + V_D u_k + \beta_{CD}). \end{aligned}\right\} \tag{14}$$

Table 3. Model parameters.

$M_{T,1} = 4351$ kg	$A_1 = 6.4\,m^2$
$M_{T,2} = 4351$ kg	$A_2 = 8.9\,m^2$
$C_{p,a} = 1.506$ kJ/kg K	$T_{a,in} = 447$ K
$C_{p,w} = 4.183$ kJ/kg K	$\dot{M}_a = 1.26$ kg/s
$k_1 = 1.136$ kJ/m^2 s K	$T_{w,in} = 293$ K
$k_2 = 0.738$ kJ/m^2 s K	$\dot{M}_w = 0.97$ kg/s

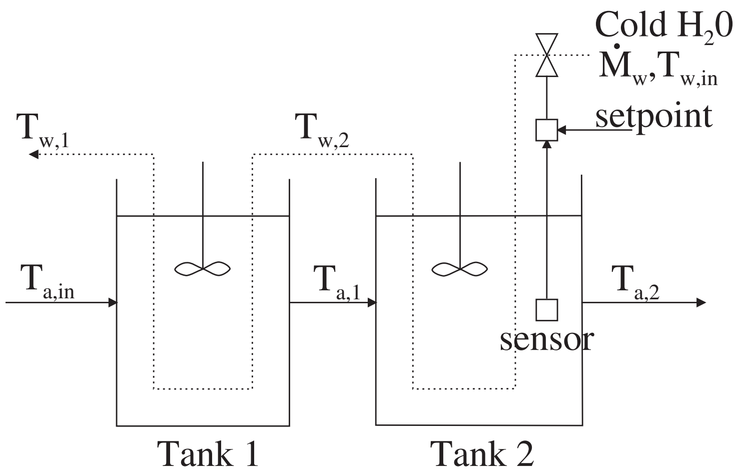

Figure 3. Two-tank cooling process.

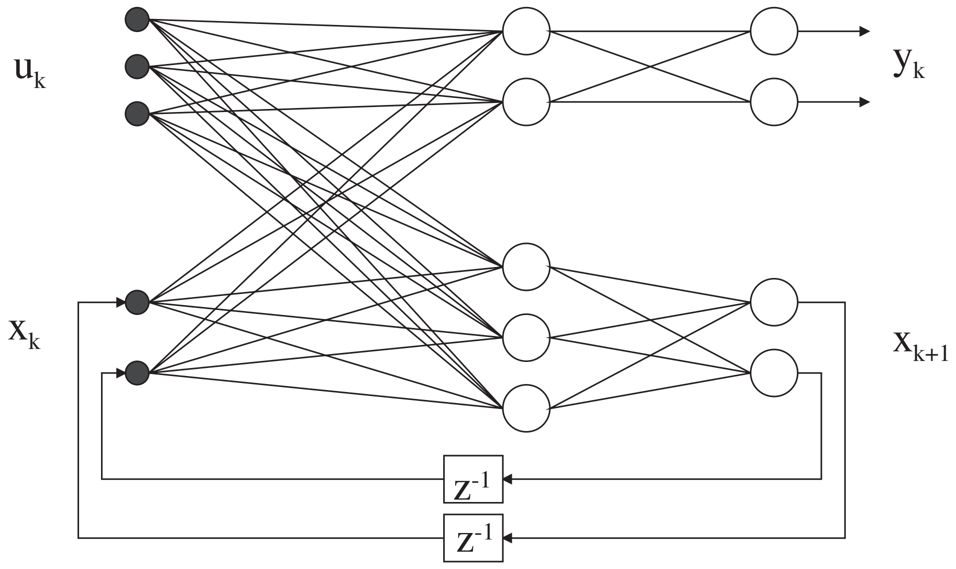

Figure 4. Neural state space model.

As illustrated in figure 4, neural state space models are recurrent NNs. The dimensions of the weight matrices and bias vectors are $W_{AB} \in \Re^{n \times n_{hx}}$, $V_A \in \Re^{n_{hx} \times n}$, $V_B \in \Re^{n_{hx} \times m}$, $\beta_{AB} \in \Re^{n_{hx}}$, $W_{CD} \in \Re^{\ell \times n_{hy}}$, $V_C \in \Re^{n_{hy} \times n}$, $V_D \in \Re^{n_{hy} \times m}$ and $\beta_{CD} \in \Re^{n_{hy}}$, where n, m, ℓ, n_{hx} and n_{hy} are the number of states, inputs, outputs and hidden neurons, respectively.

The 40 weights and biases of a neural state space model with $n = 2$, $m = 1$, $\ell = 1$, $n_{hx} = 5$ and $n_{hy} = 2$ are estimated by minimizing an output error criterion in the form (1). The training set consists of 2270 data produced by applying steps of various amplitudes and durations in the cooling water stream. During these experiments, the inlet acid temperature is constant (table 3). The evolution of the criterion for 10 independent runs starting with random initial estimates (figure 5) illustrates the good performance of the adaptive 1SP-GS algorithm as compared with the original 1SPSA algorithm. Figure 6 shows some cross-validation results demonstrating the good model agreement.

5.2. *NN predictive control*

Once the modelling task has been achieved, the neural state space model can be used as a process emulator in a model-based predictive control scheme (figure 7). The NN state space model generates prediction of future process outputs over a specified prediction horizon, which allows a quadratic performance criterion to be defined, i.e.

$$J = \sum_{i=N_1}^{N_2} (y^r(k+i) - \hat{y}(k+i))^2 + \lambda \sum_{i=1}^{N_2} \Delta u(k+i-1)^2, \tag{15}$$

where $y^r(k)$ is the output of a model reference, and N_1 and N_2 define the horizons over which the tracking errors $e(k) = y^r(k) - \hat{y}(k)$ and control increments $\Delta u(k)$ are considered. The weighting factor λ penalizes the control increments.

The control signal u(k) can be produced in two ways:

- On-line optimization routine is applied to the minimization of J with respect to the control moves u(k) over the prediction horizon, and the optimal control signal is directly applied to the process.
- Minimization of J is accomplished with respect to the weights of a feedforward NN controller, which in some sense mimics the action of the on-line optimization routine.

In this study, the second approach has been chosen as it is conceptually more robust. Indeed, the NN controller keeps track of the optimization procedure and can produce a control signal even during periods where optimization cannot be performed satisfactorily (e.g. in an on-line application, higher priority events and interrupts can prevail on optimization). In our implementation, controller training occurs only during the transient phases in which the system is sufficiently excited by the input signals. Optimization is suspended in steady-state phases to avoid detuning the NN controller, which produces the input signal applied to the process.

Following the line of thought in Tan (1993), the NN controller parametrization is chosen in a similar way as in a classical PID controller, i.e. with three inputs $i_1(k) = e(k)$, $i_2(k) = \sum_{i=1}^{k} e(i)$ and $i_3(k) = e(k) - e(k-1)$, one hidden layer with n_{hl} nodes and one

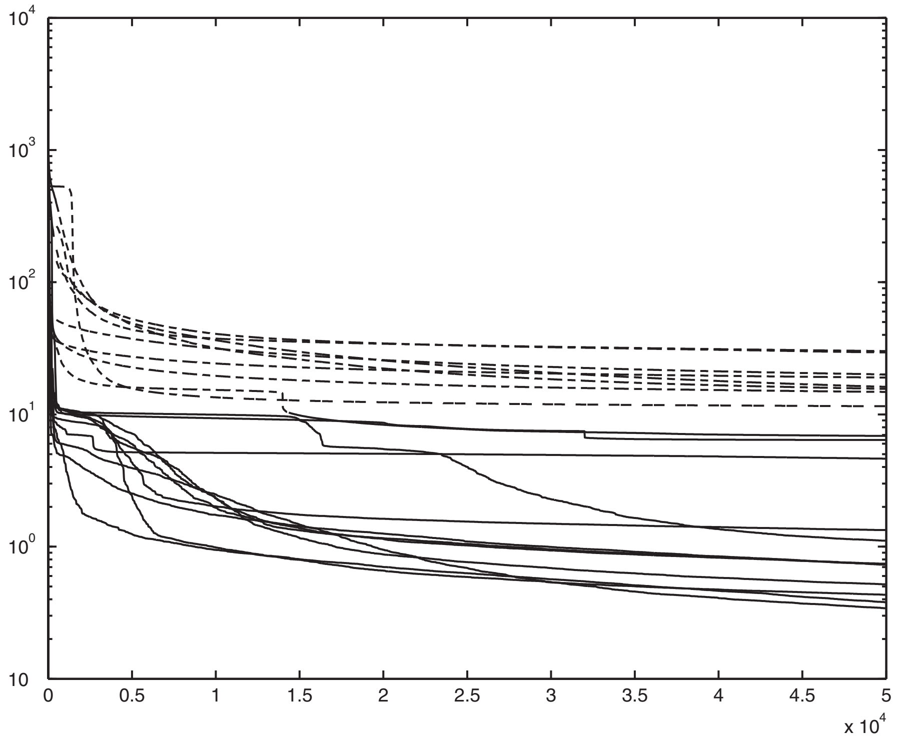

Figure 5. Two-tank system identification: mean-square error curves for 10 independent runs. Solid lines, adaptive 1SP-GS; dashed lines, original 1SPSA.

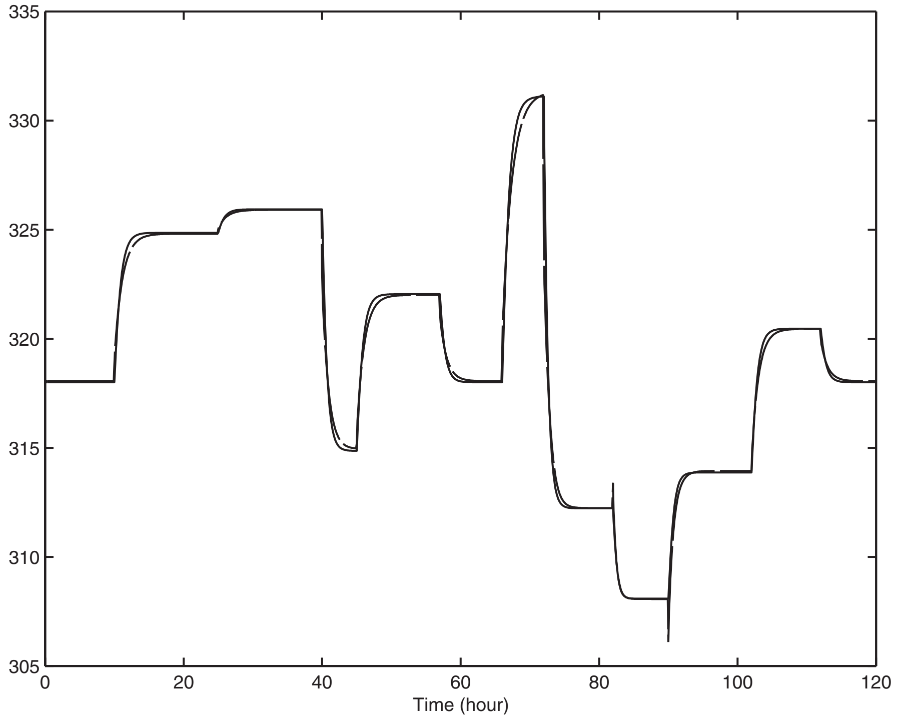

Figure 6. Two-tank system identification: cross-validation results. Solid line, outlet temperature of sulfuric acid; dotted line, NN output.

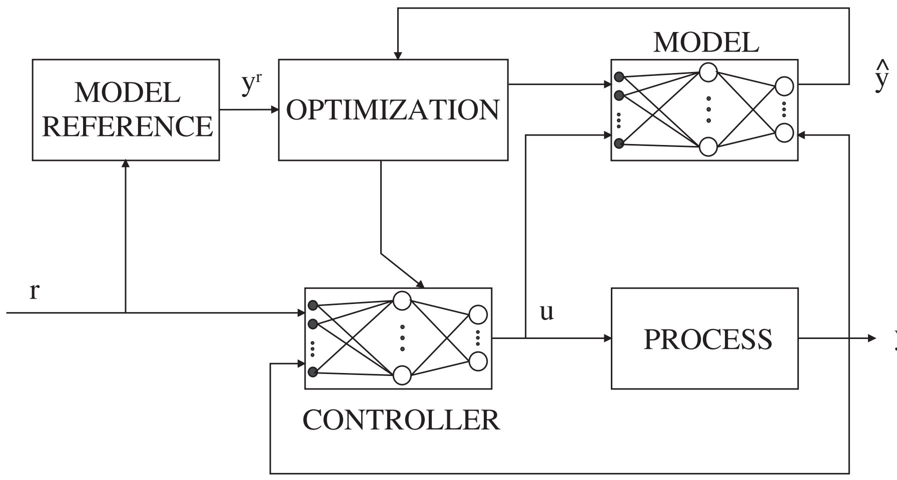

Figure 7. Model-based predictive neural control scheme.

output. As only a weighted sum of $i_1(k)$, $i_2(k)$, $i_3(k)$ is needed to mimic the PID algorithm, the biases of the NN controller are set to zero.

Of course, the prediction of the NN emulator is usually not perfect, and the effect of modelling errors can be treated as an additive disturbance which can be estimated at the kth sampling instant in a manner similar to dynamic matrix control (DMC) (Patwardhan *et al.* 1990), i.e.

$$\left.\begin{aligned} y^m(k) &= \hat{y}(k) + d(k-1) \\ d(k) &= y(k) - \hat{y}(k), \end{aligned}\right\} \qquad (16)$$

where $y^m(k)$is substituted for $\hat{y}(k)$ in the expression of the performance criterion (15), which allows modelling errors to be compensated.

An NN PID controller with $n_{hl} = 2$ is used to control the acid temperature from the second tank $T_{a,2}(t)$ by acting on the cooling water stream $\dot{M}_w(t)$. The eight unknown NN weights are estimated by minimizing the performance criterion (15) with $N_1 = 1$, $N_2 = 20$ and the output $y^r(k)$ of a second-order model reference with $\xi = 0.8$ and $\omega_n = 4$.

In this application, the requirements on the optimization algorithm are: (1) small computational costs so that the criterion minimization can take place within a sampling period, and (2) robustness and reliability so that, even in the presence of perturbations, at least a lower value of the error criterion can be achieved. On the other hand, accuracy in estimating the controller parameters is not determinant. With this view, the minimization of the receding horizon criterion (15) is performed using 1SPSA-GS, with a maximum of 100 iterations per sampling interval. Figure 8 illustrates the excellent tracking capabilities of the NN PID controller.

However, in the face of disturbances in the acid feed temperature, the prediction of the NN process emulator deteriorates, which results in large tracking errors. Figure 9 shows the effect of step disturbances of -4 K in $t = 15$ h and $+8$ K in $t = 42$ h. The effect of these non-measured disturbances can be compensated using (16) (figure 10). The effect of measurement noise with a standard deviation of 0.75 K is also shown in figure 11. In all these cases, the predictive NN PID control scheme displays very satisfactory performance.

6. Conclusion

The SP approach devised by Spall (1992, 2000) is a very powerful technique that allows an approximation of the gradient and/or the Hessian of a noisy performance criterion to be computed by effecting simultaneous random perturbations in all the parameters. This numerical procedure, which contrasts with standard finite difference approximations in which the evaluation of the criterion gradient is achieved by varying the parameters one at a time, appears a priori attractive to address the parameter estimation problem in NNs, which is characterized by large sets of unknown parameters. In this connection, this paper discusses several test results in nonlinear system identification and control. Especially, a variation of the first-order SP algorithm, including adaptive gain sequences, gradient smoothing and a step rejection procedure, is described and evaluated with a realistic numerical application. In addition, a predictive NN PID control scheme is developed that shows very satisfactory performance. Remarkable properties of the SP algorithms are their simplicity of implementation and ease of use.

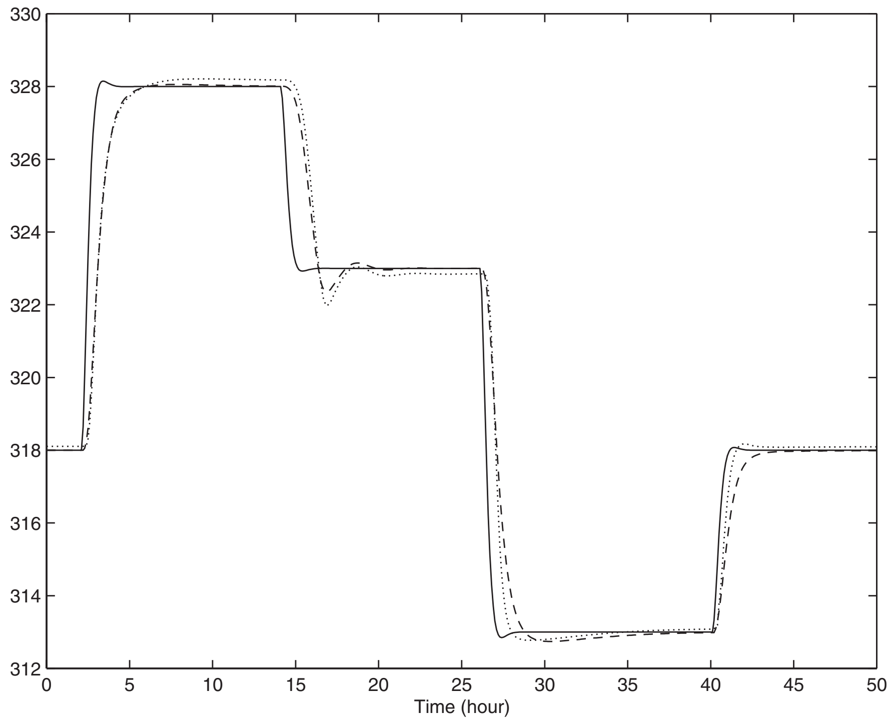

Figure 8. Two-tank system: tracking capabilities of an NN PID for the acid outlet temperature. Solid line, model reference; dotted line, NN process emulator; dashed line, process output.

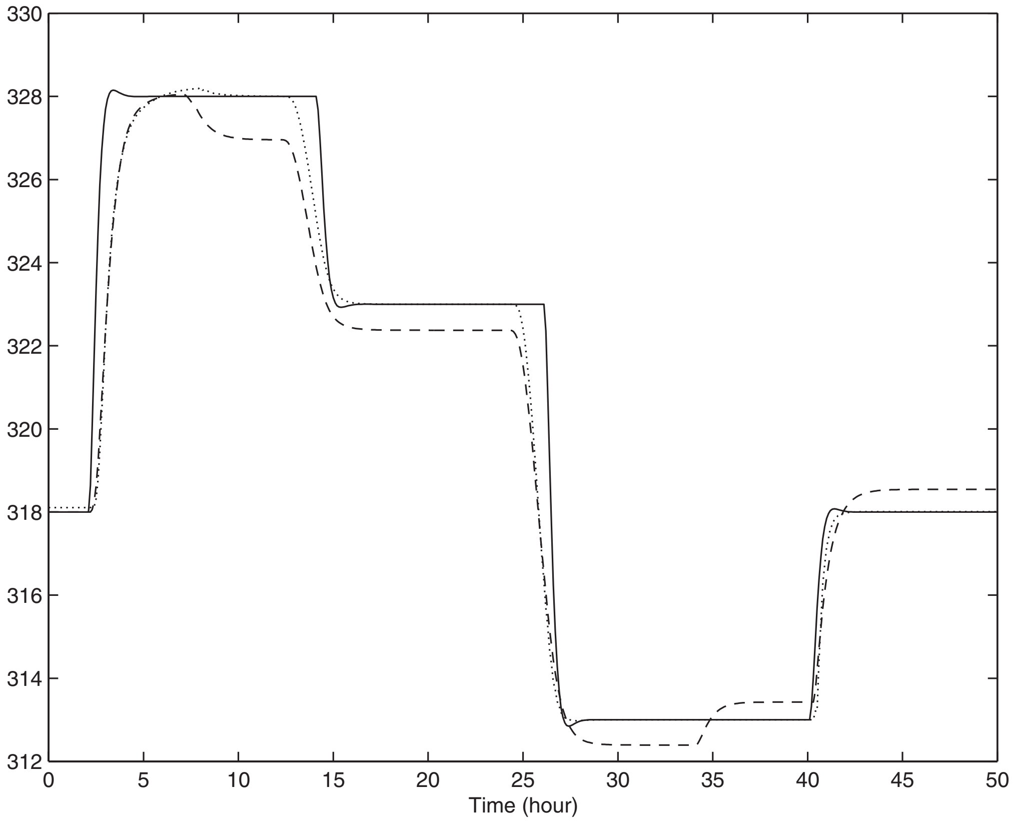

Figure 9. Two-tank system: effect of non-measurable disturbances in the acid feed temperature on the acid outlet temperature (key as in figure 8).

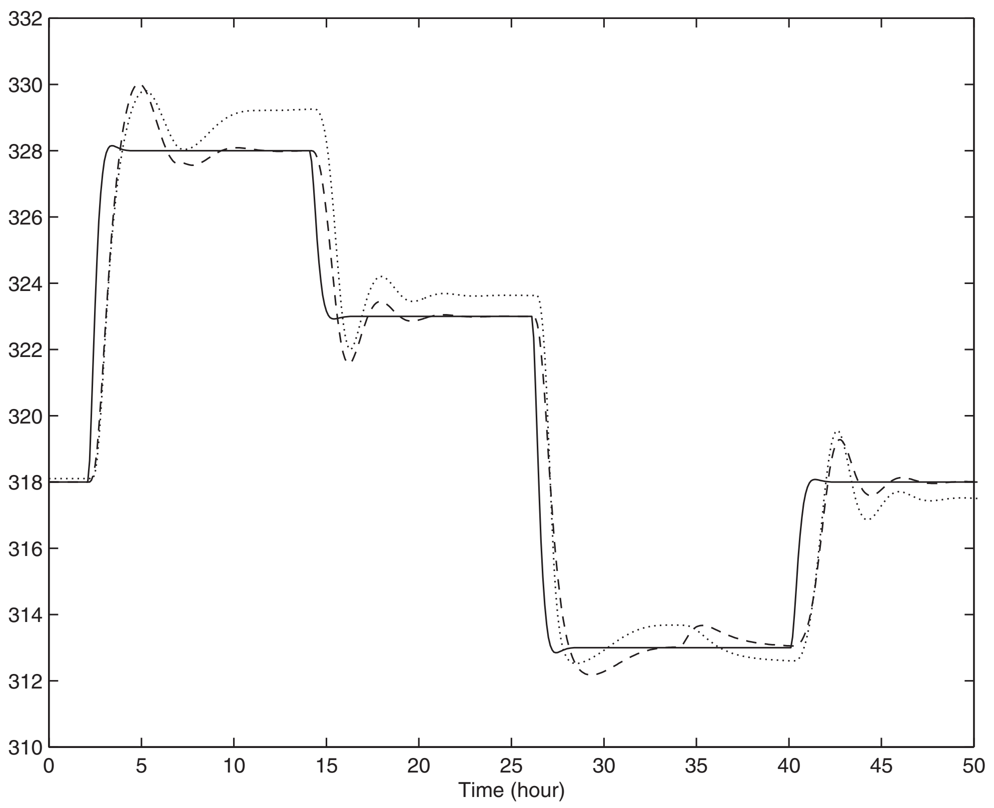

Figure 10. Two-tank system: modelling error compensation (key as in figure 8).

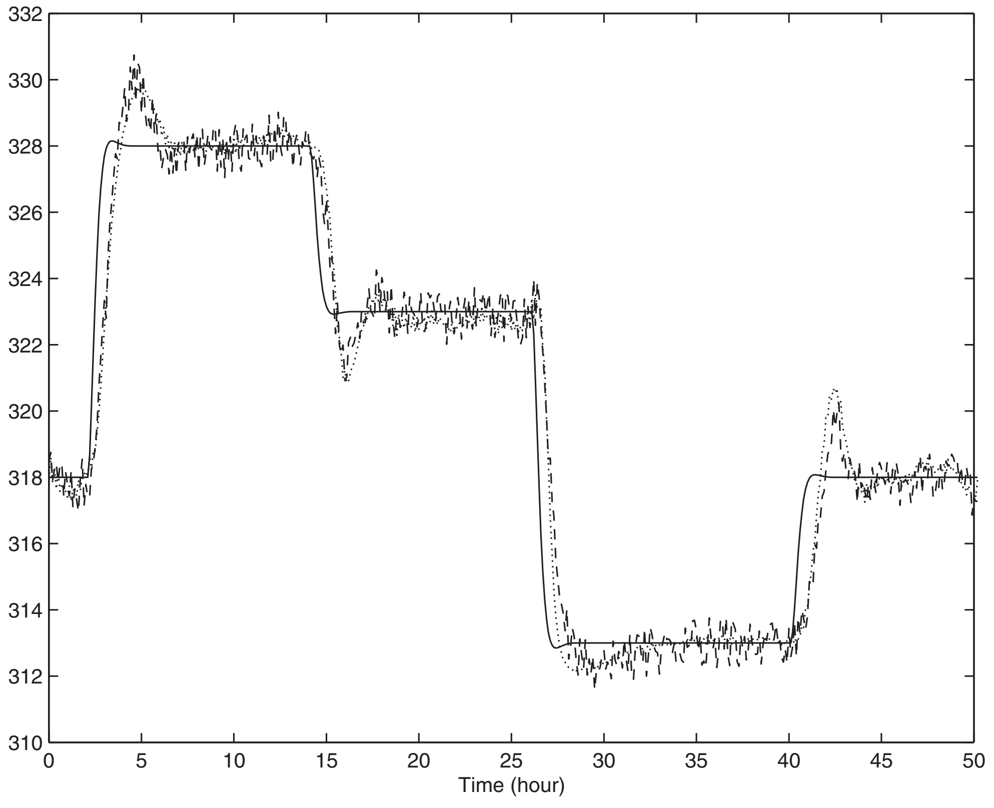

Figure 11. Two-tank system: effect of measurement noise (key as in figure 8).

Acknowledgements

The authors are very grateful to Dr James Spall for insightful comments and suggestions.

References

Ayoubi, M., 1996, *Nonlinear System Identification Based on Neural Networks with Locally Distributed Dynamics and Application to Technical Processes* (Dusseldorf: VDI).

Hunt, K. J., Sbardaro, D., Zbikowski, R., and Gawthrop, P. J., 1992, Neural network for control systems—a survey. *Automatica*, **28,** 1083–1112.

Jenson, V. G., and Jeffreys, G. V., 1977, *Mathematical Methods in Chemical Engineering* (New York: Academic Press).

Ljung, L., 1987, *System Identification*, *Theory for the User* (Eaglewood Cliffs: Prentice Hall).

Narendra, K. S., and Parthasarathy, K., 1990, Identification and control of dynamical systems using neural networks. *IEEE Transactions on Neural Networks*, **1,** 4–27.

Narendra, K. S., and Parthasarathy, K., 1991, Gradient methods for the optimization of dynamical systems containing neural networks. *IEEE Transactions on Neural Networks*, **2,** 252–262.

Patwardhan, A. A., Rawlings, J. B., and Edgar, T. F., 1990, Nonlinear model predictive control. *Chemical Engineering Communications*, **87,** 123–141.

Reklaitis, G. V., Ravindran, A., and Ragsdell, K. M., 1983, *Engineering Optimization—Methods and Applications* (Chichester: Wiley).

Rumelhart, D. E., Hinton, G. E., and Williams, R. J., 1986, Learning representations by back-propagating errors. *Nature*, **323,** 533–536.

Spall, J. C., 1992, Multivariate stochastic approximation using a simultaneous perturbation gradient approximation. *IEEE Transactions on Automatic Control*, **37,** 332–341.

Spall, J. C., 2000, Adaptive stochastic approximation by the simultaneous perturbation method. *IEEE Transactions on Automatic Control*, **45,** 1839–1853.

Spall, J. C., and Cristion, J. A., 1994, Nonlinear adaptive control using neural networks: estimation with smoothed form of simultaneous perturbation gradient approximation. *Statistica Sinica*, **4,** 1–27.

Suykens, J., Vandewalle, J., and De Moor, B., 1996, *Artificial Neural Networks for Modelling and Control of Non-Linear Systems* (Dordrecht: Kluwer).

Tan, Y., 1993, An architecture for adaptive neural control. *Journal A*, **34,** 12–16.

Werbos, P. J., 1990, Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, **78,** 1550–1560.

Zhu, X., and Spall, J. C., 2002, A modified second order SPSA optimization algorithm for finite samples. *Int. J. of Adaptive Control and Signal Processing*, **16**, 397–409.