J. Indian Inst. Sci., Jan.–Feb. 1997, 76, 71–84.

Indirect adaptive control of periodic time-varying systems using neural networks

B. SRINIVASAN*, U. R. PRASAD**, AND N. J. RAO***
Department of Computer Science and Automation, Indian Institute of Science, Bangalore - 560012, India.
* presently at Institut d'Automatique, Swiss Federal Institute of Technology, CH-1015, Lausanne, Switzerland.
E-mail: srinivasan@ia.epfl.ch
** E-mail: urp@csa.iisc.ernet.in. *** E-mail: njrao@cedt.iisc.ernet.in

Abstract

Indirect adaptive control of a class of systems, which perform repetitive operations and which are not necessarily time-invariant is considered. Neural networks are used for this purpose, with the error committed in one cycle being used to correct the corresponding parameters in the next cycle. It is shown that the time variations of the plant parameters cannot be captured if every parameter is allowed to vary at the sampling rate. Hence, various schemes for unfolding the network partially are proposed. Identification and adaptive control algorithms for such schemes are presented. Simulation results on a simple robot are also provided to illustrate the application of the scheme.

Keywords: Indirect Adaptive Control, Periodic Systems, Time Varying Systems, Neural Networks

1. Introduction

Recently, neural networks have been successfully used in the areas of modeling and control of dynamic systems[1–5]. Various approaches for employing neural networks in control have been addressed, which include inverse control[6], feedback linearisation[7], stochastic methods[8], optimal control[3], and adaptive control methods[2,5].

An adaptive control scheme is one where the controller is automatically adjusted in response to variations in the plant and its environment so that the closed loop system behavior matches the specified behavior. One of the assumptions made in most of the adaptive control schemes proposed in the literature[9, 10, 2, 5] is that the variation of the plant parameters is 'relatively' slow compared to the plant dynamics. In other words, for all practical purposes the plant is considered time-invariant. However, when using neural networks, such an assumption is not always appropriate. Methods based on sensitivity models[2] strongly depend on the assumption of time-invariance, while back-propagation through time[11, 3], or equivalently back-propagation through the adjoints[4], does not require such an assumption as a pre-requisite for calculating the required sensitivities.

With time-invariance, the response to past actions can also be used to deduce the sensitivity of a parameter on the cost. But, without such an assumption, a signal input or a parameter change can be assessed only after a delay. The delay is quite logical since the presence of dynamics causes the result of an action to manifest only after a lapse of time.

However, in reality, it is impossible to go backwards in time to correct it. The problem of delay is clearly absent in an off-line scheme such as Nguyen and Widrow[3] and was overcome using prediction in Srinivasan *et al.*[5]. Here we take a different stand by considering the class of systems which perform repetitive operations. An error committed at some point of time in the present cycle is used to correct the input at a corresponding instant in the next cycle.

Many practical systems such as the assembly robots, numerically controlled machines and others which are motivated by the science of cybernetics fall under this category. Other processes such as the batch reactors can also be included in this group. Hence, it is logical to use a learning scheme which is also biologically motivated. Learning by humans and animals, especially basic capabilities, can be viewed as 'learning to do operations which are by-and-large repetitive'. In learning a repetitive operation, there is only marginal, almost negligible, learning during any one operation, while a substantial amount of learning takes place between two repetitions. This is due to the fact that the presence of dynamics causes the effect of an action to manifest only after a certain length of time. In biological systems the error or reinforcement is 'stored in memory' and is used for correction only when it is called upon to act the next time around. Some authors such as Barto[12] consider this as the basic difference between an adaptive controller and a learning controller.

All indirect adaptive control schemes rely on the certainty-equivalence assumption, as the identified plant is used for controller synthesis instead of the true one. For this assumption to be valid, the parameters of the plant have to converge to the global minimum[5]. For this to happen in the time-varying case, a necessary condition is that all the parameters should not be allowed to vary at the sampling rate. Hence, the network should be unfolded in time only partially, for which various schemes are proposed. Gradient calculation and update laws for each of the schemes are presented and discussed. Then, the adaptive control problem is formulated as a problem of identification and algorithms similar to those used for identification are used. Algorithms for the adaptation of controller parameters for different control structures, i.e., open loop, state feedback, etc., are illustrated as special cases of the output feedback structure developed till then.

The paper is organised as follows: In Section 2, the method of back-propagation through the adjoint is briefly reviewed. Section 3 deals with the definition of periodic systems and the identifiability problems faced while modeling them. Identification and adaptive control algorithms for various schemes are provided in Section 4. Simulation results on a simple robot are presented in Section 5 and Section 6 concludes the paper.

2. Parametric Sensitivities using Adjoints

Adaptive control schemes can be broadly classified into (i) indirect, and (ii) direct schemes, depending on whether or not an explicit plant model is used. The indirect adaptive control problem should always be attacked in two stages : (i) the identification stage - obtaining a model of the plant from the input-output data and (ii) the controller design

stage - obtaining a controller which achieves the desired control objectives, given the identified model of the plant.

In the neural net framework, both the identification and adaptive control problems are posed as optimisation problems as stated below.

The Identification Problem

$$\min_{W_{mf}(k),W_{mg}(k)} J_i = \frac{1}{2}\sum_{i=0}^{M}\left\|y_p(k-i)-y_m(k-i)\right\|^2 \tag{1}$$

subject to

$$x_m(k+1) = f_m\left(x_m(k),u(k),W_{mf}(k)\right) \tag{2}$$

$$y_m(k) = g_m\left(x_m(k),u(k),W_{mg}(k)\right) \tag{3}$$

The Adaptive Control Problem

$$\min_{W_{cf}(k),W_{cg}(k)} J_c = \frac{1}{2}\sum_{i=0}^{M}\left\|y_d(k-i)-y_p(k-i)\right\|^2 \tag{4}$$

subject to

$$x_m(k+1) = f_m\left(x_m(k),u(k),W_{mf}(k)\right) \tag{5}$$

$$y_m(k) = g_m\left(x_m(k),u(k),W_{mg}(k)\right) \tag{6}$$

$$x_c(k+1) = f_c\left(x_c(k),y_m(k),y_r(k),W_{cf}(k)\right) \tag{7}$$

$$u(k) = y_c(k) = g_c\left(x_c(k),W_{cg}(k)\right) \tag{8}$$

where J_i is the identification fit criterion and J_c the control fit criterion defined over the M samples of the batch under consideration, ||·|| the standard Euclidean norm, $y_p(k)$ and $y_m(k)$ are the respective outputs of the actual plant and the plant model (see Fig. 1), for the same applied input $u(k) = y_c(k)$, viz., the output of the controller (see Fig. 2), $y_d(k)$ the desired output trajectory, $y_r(k)$ the reference input applied to the system, $x_m(k)$ and $x_c(k)$ the states of the model and the controller respectively, f_m and g_m the respective input-state and the output mappings of the plant model with f_c, g_c being the corresponding mappings for the controller, and k the discretized time. The nonlinear mappings f_m, g_m, f_c, and g_c are approximated by multi-layer perceptrons whose weights are W_{mf}, W_{mg}, W_{cf} and W_{cg} respectively.

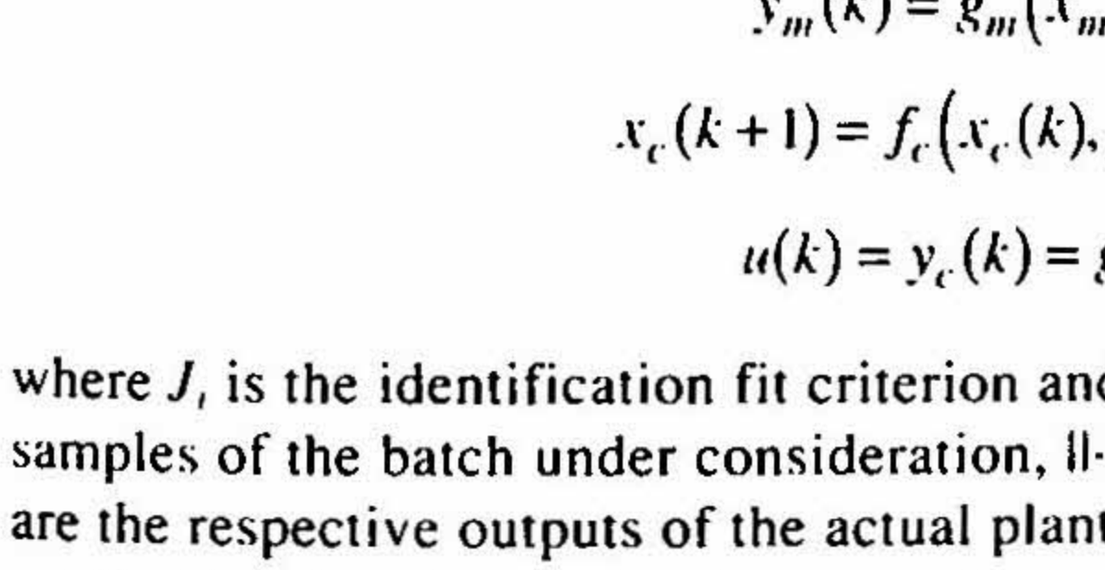

FIG. 1. Model of the plant.

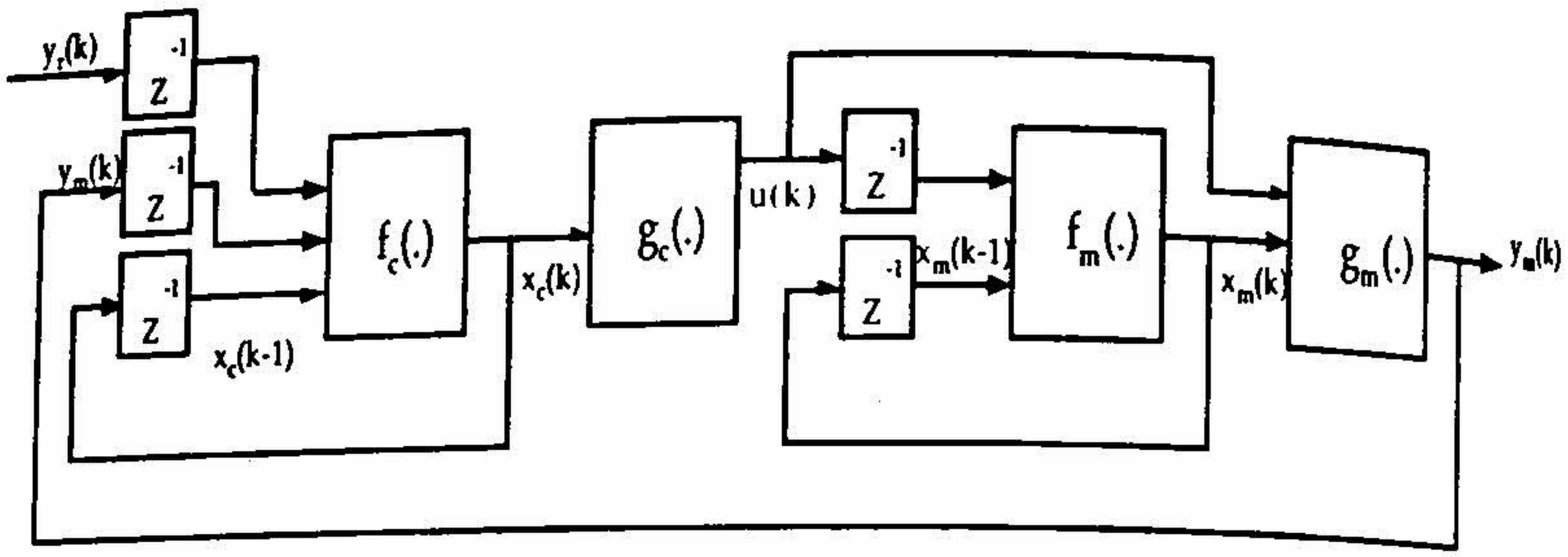

FIG. 2. The basic control loop.

In general, the identification problem is treated as a problem with only algebraic constraints, though the constraints are inherently of a dynamic nature. Since, the adaptive control problem has to handle dynamics in any case, the algorithms used for these two problems are, in general, quite different. Yet, if the dynamic constraints are handled using adjoints at the identification stage itself, then similar algorithms can be used for both identification and adaptive control. Such a strategy will be followed in the rest of the paper.

The adjoint of the given system, through which the error is back propagated, is constructed by the three steps: (i) gradient linearisation, (ii) reversing the directions of both time and signal flows, and (iii) by changing the nodes to summing junctions and vice versa. For an interconnected system, the error is back propagated through the adjoint of the interconnection, in which each of the system blocks is replaced by its corresponding adjoint. Since we have two cost criteria, we have two adjoints corresponding to the two criteria which are given below (refer Figs. 3 and 4). The first adjoint corresponds to the plant model alone while the second corresponds to the controller-plant combination.

$$\lambda_{im}(0) = 0$$

$$\lambda_{im}(\kappa) = \mathfrak{I}^T_{x_m(M-\kappa)} f_m \lambda_{im}(\kappa-1) + \mathfrak{I}^T_{x_m(M-\kappa)} g_m \nabla_{ym(M-\kappa)} J_i \tag{9}$$

$$\lambda_{cm}(0) = 0,\ \lambda_{cc}(0) = 0,$$

$$\lambda_{cm}(\kappa) = \mathfrak{I}^T_{x_m(M-\kappa)} f_m \lambda_{cm}(\kappa-1) + \mathfrak{I}^T_{x_m(M-\kappa)} g_m \mathfrak{I}^T_{y_p(M-\kappa)} f_c \lambda_{cc}(\kappa-1) + \mathfrak{I}^T_{x_m(M-\kappa)} g_m \nabla_{y_p(M-\kappa)} J_c \tag{10}$$

$$\lambda_{cc}(\kappa) = \mathfrak{I}^T_{x_c(M-\kappa)} f_c \lambda_{cc}(\kappa-1) + \mathfrak{I}^T_{x_c(M-\kappa)} g_c \mathfrak{I}^T_{u(M-\kappa)} g_m \lambda_{cm}(\kappa-1) + \mathfrak{I}^T_{x_c(M-\kappa)} g_c \mathfrak{I}^T_{u(M-\kappa)} g_m \nabla_{y_p(M-\kappa)} J_c \tag{11}$$

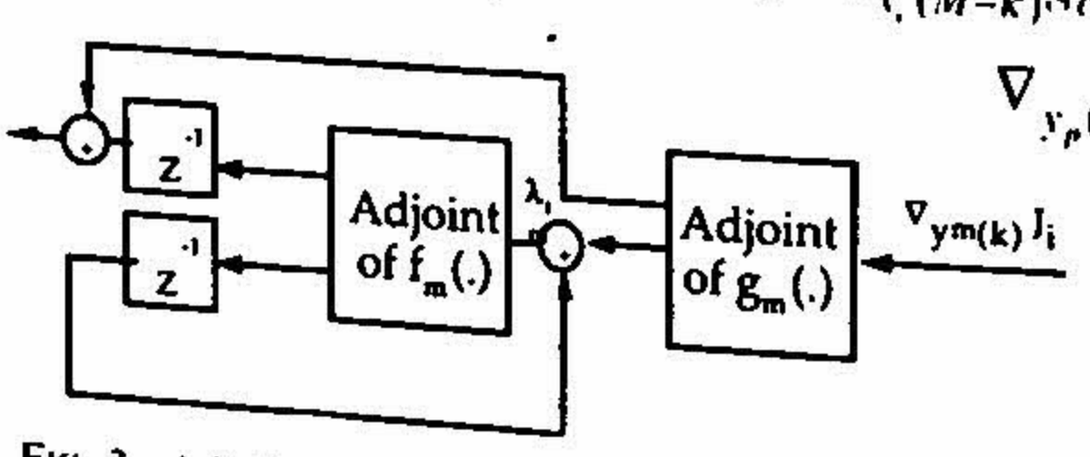

FIG. 3. Adjoint of the plant model - for identification purposes.

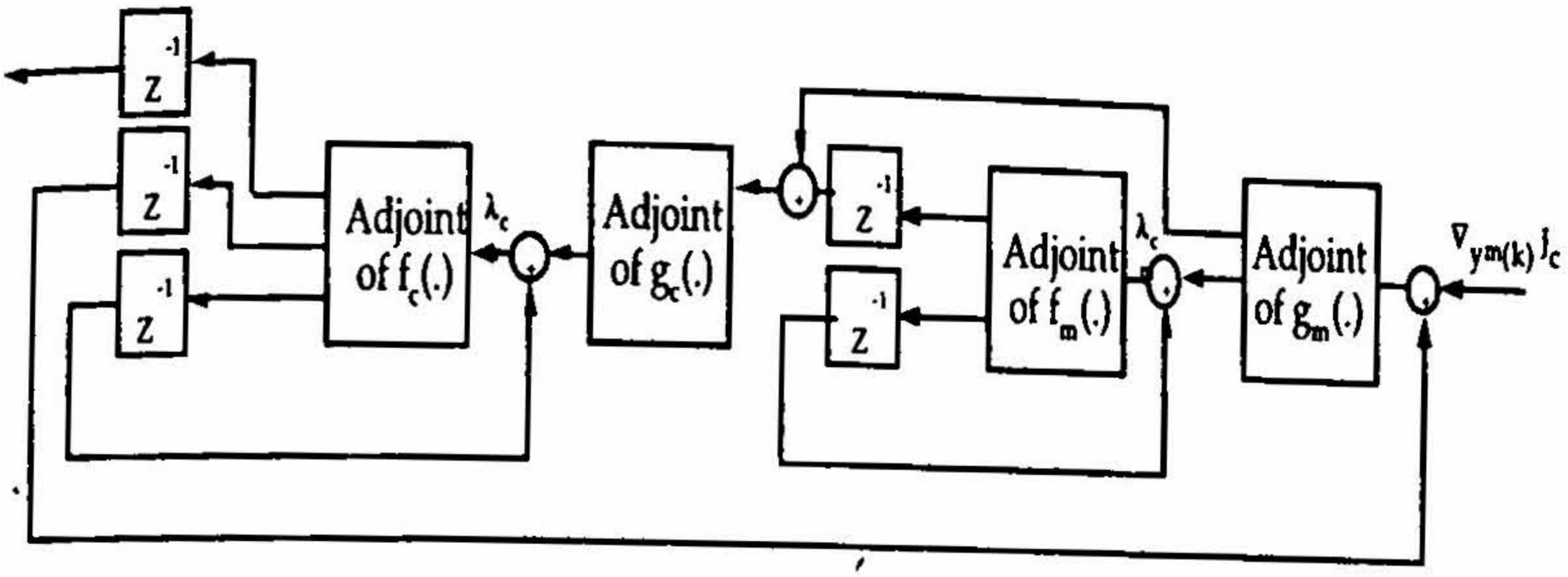

FIG. 4. Adjoint of the controller-plant combination - for control purposes.

where $\Im$ is used to denote the Jacobian matrix at the specified time instant, ∇ the gradient, λ_{im} the adjoint of the plant model constructed for identification purposes, λ_{cm} and λ_{cc} the adjoints of the plant model and the controller constructed for control purposes and κ the retrograde time.

Srinivasan *et al.*[4] pointed out that the sensitivity of a parameter at time instant 'k' can be obtained from the adjoint by multiplying (i) the signal that enters the weight in the forward (*i.e.*, system) run at time instant 'k' and (ii) the signal which enters the weight after $M-k$ retrograde time steps in the adjoint run. So,

$$\nabla_{W_{mf}(k)} J_i = \Im^T_{W_{mf}(k)} f_m \lambda_{im}(M-k-1) \tag{12}$$

$$\nabla_{W_{mg}(k)} J_i = \Im^T_{W_{mg}(k)} g_m \nabla_{ym(k)} J_i \tag{13}$$

$$\nabla_{W_{cf}(k)} J_c = \Im^T_{W_{cf}(k)} f_c \lambda_{cc}(M-k-1) \tag{14}$$

$$\nabla_{W_{cg}(k)} J_c = \Im^T_{W_{cg}(k)} g_c \Im^T_{u(k)} f_m(k) \lambda_{cm}(M-k-1) \tag{15}$$

It is important to note that the adjoint method gives the sensitivity of a particular time-tagged weight on any of the criterion functions. Hence it is immaterial whether or not plant parameters are time-varying. In an off-line scheme, we can go backwards in time after the batch is over to independently correct each of these weights spread over time. Consider such situation where the plant is known and the objective is to choose the controller parameters so as to minimise the control fit criterion J_c. Then the gradient descent parameter update will look like,

$$W^{i+1}_{cf}(k) = W^i_{cf}(k) - \mu \nabla_{W^i_{cf}(k)} J_c \;\&\; W^{i+1}_{cg}(k) = W^i_{cg}(k) - \mu \nabla_{W^i_{cg}(k)} J_c \tag{16}$$

where the superscript indicates the iteration number.

However, if the plant is unknown, one has to necessarily run the plant for the sake of identification. In such a case, on-line schemes have the advantage of testing possible controllers directly on the plant. The price paid, however, is that a weight cannot be corrected

before the batch is complete and it is not possible to go backwards in time to correct the weight later. This is due to the fact that in a dynamic system we cannot assess the result of an action without examining the response over a non-zero interval of time. To overcome this problem, prediction was resorted to in Srinivasan *et al.*[5]. The concept is similar to model predictive control, where the response is predicted over a finite horizon, and only the initial part of the optimal input is actually given to the system. Here, only the weight corresponding to the present time instant is corrected. Correction of inputs and controller states is also found necessary for the sake of convergence[9].

In this paper, we consider only the class of systems which perform repetitive operations. The advantage with this class is that the error committed in the present run/cycle can be used to correct the parameters of the next cycle. This way, no prediction is necessary and update laws similar to off-line iterations can be used on-line. Since time-invariance or the knowledge of time variations is a pre-requesite only for the sake of prediction, even that assumption drops out.

3. Periodic Systems and their Identifiability

A periodic system is one in which a particular batch operation is repeated over and again. Let us assume that the process runs over M sampling instants and that this number is fixed. This type of operation is somewhat similar to the iterative solution of an optimal control problem with a fixed terminal time. Now, a periodic system is one in which,

$$y_d(k+M) = y_d(k);\ \forall k = 0, 1, \ldots, M-1 \tag{17}$$

$$W_p(k+M) = W_p(k);\ \forall k = 0, 1, \ldots, M-1, \tag{18}$$

and in an ideal case

$$y_p(k) = y_p(k+M);\ \forall k = 0, 1, \ldots, M-1 \tag{19}$$

where $W_p(k)$ are some representative parameters of the plant.

From such a definition of a periodic system, it is clear that the sensitivity can be used to correct $W_{(\cdot)}(k+M)$. The update laws are given by,

$$W_{mf}(k+M) = W_{mf}(k) - \mu\nabla_{W_{mf}(k)}J_i \;\&\; W_{mg}(k+M) = W_{mg}(k) - \mu\nabla_{W_{mg}(k)}J_i \tag{20}$$

$$W_{cf}(k+M) = W_{cf}(k) - \mu\nabla_{W_{cf}(k)}J_c \;\&\; W_{cg}(k+M) = W_{cg}(k) - \mu\nabla_{W_{cg}(k)}J_c \tag{21}$$

This clearly means that we should have M different copies of each of the networks and update them independently. The plant is thus completely unfolded in time and there need not be any relationship between the two networks. Though, this gives an advantage that the parameters can be time varying in an arbitrary manner at the sampling rate, a major question that arises is whether we can use the model obtained after identification for control purposes at all.

All indirect adaptive control methods inherently assume certainty-equivalence, which means that the identified model is assumed to be a true replica of the actual plant. In a

linear time-invariant case, the identified model asymptotically matches the true one if the input is persistently exciting[9]. However, for a nonlinear plant, the problem of getting trapped in a local minimum arises; when so trapped, it cannot be proved that the assumption of certainty-equivalence is valid. The validity can be established when the parameters of the plant model converge to the global minimum (which means either an algorithm which performs global minimisation should be used or the initial conditions should be in the attraction region of the global minimum)[9]. In a time-varying case, an additional aspect is that the convergence to the global minimum should be assured for the plant at every unfolded time instant (*i.e.*, all the M sets of networks should converge to the global minimum).

This brings us to the basic problem of identifiability[13]. The parameter estimation can be unique only if the total number of conditions (C say) which the parameters have to satisfy is greater than or equal to the total number of parameters (P say), *i.e.*, $C \geq P$. Note that this is a necessary condition and does not guarantee a unique determination of the parameters. This is a very basic condition in addition to which we need to assume persistency of excitation, initial conditions being in the attraction region of the global minimum etc., to obtain parametric convergence. With complete unfolding, this identifiability condition is violated even for some of the simple cases, which indicates that no parameter convergence can be established.

Proposition 1: Let m be the number of outputs and p the total number of parameters corresponding to the network at a single time instant. If the plant is completely unfolded in time, the identifiability condition

$$C \geq P \Rightarrow p \leq m \tag{22}$$

Proof: If p is the number of network parameters at any time instant and M the number of stages, then the total number of parameters $P = pM$. Each of the m outputs corresponds to a condition which the parameters have to satisfy. With all the outputs at the M sampling instants available, the total number of conditions the parameters have to satisfy $C = mM$. So the identifiability condition $C \geq P \Rightarrow p \leq m$.

Remark 1: In a single output system, Proposition 1 implies that the number of parameters should be less than or equal to 1, which is unrealistic.

Looking back, such an unrealistic scenario stems from the fact that we have assumed that plant parameters vary at the sampling rate which is at least twice the maximum frequency encountered by the signals in the system. If the parameters vary, they do so at a much slower rate and hence unfolding the network completely does not make much sense. Hence, various partial unfolding schemes are discussed below.

Option 1:

The first natural partial unfolding scheme is to consider that the parameters of the plant/model change only once in r time instants. By the definition of the periodic system, this means

$$W_p(k+i) = W_p(k),\ \forall i = 1, 2, \ldots, r-1 \text{ and } k = 0, r, 2r, \ldots, M-r \quad . \tag{23}$$

Proposition 2: If the plant is partially unfolded as in Option 1, then the identifiability condition

$$C \geq P \Rightarrow p \leq mr \tag{24}$$

Proof: Since the parameters change only once in r time instants, the number of copies of the networks maintained is (M/r). So the total number of parameters $P = (pM/r)$. The total number of conditions the parameters have to satisfy still remains $C = mM$. So the identifiability condition $C \geq P \Rightarrow p \leq mr$. □

Option 2:

In Option 1 all the parameters change at the same rate, though in actuality it need not be the case. If a single rate of parameter change is chosen, r should be small enough to capture the fastest time variation and be big enough to accommodate all the parameters of the model. Instead, one could use to advantage the fact that all the parameters do not change at the same rate while unfolding the plant.

Let p_j be the number of parameters in the model which change every 'j' time instants, with 'j' allowed to vary from 1 to M. The number of time invariant parameters is p_M. Option 1 is a special case of Option 2 where $p_j = 0$; $\forall j \neq r$ and $p_r = p$. If the weight vector W_p is partitioned into $W_{p1}, W_{p2},\ldots, W_{pM}$, corresponding to their rates of change, then the variation of one representative group W_{p_i} is given by

$$W_{p_i}(k+i) = W_{p_i}(k); \forall i = 1,2,\ldots, j-1 \text{ and } k = 0, j, 2j,\ldots, M-j \text{ for } j = 1,2,\ldots, M \tag{25}$$

Proposition 3: If the plant is partially unfolded as in Option 2, the identifiability condition yields,

$$\sum_{j=1}^{M}\left(p_j / j\right) \leq m \text{ and } \sum_{j=1}^{M} p_j = p \tag{26}$$

Proof: Since the parameter p_j changes only once in j time instants, the number of copies of this parameter will be (M/j). Considering the summation over j, the total number of parameters $P = \sum_{j=1}^{M}(p_j M/j)$. As $C = mM$, the proposition follows.

Remark 2: In a single output case, if $p > 1$, which is normally the case, $p_1 \geq 1$ violates Proposition 3. So, p_1 should be zero. This indicates that not even one parameter can be allowed to vary at every time instant.

Option 3

To avoid having M different copies of the network, it is possible to approximate the time variation of the parameters by another neural network for which the time instant or the stage variable acts as the input, *i.e.*, $W(\cdot)(k) = f_{aux}\left(W_{aux_{(\cdot)}}, k\right)$. The auxiliary parameters, $W_{aux_{(\cdot)}}$, are chosen suitably to approximate the time variation and hence the number of such parameters required depends on the smoothness of the time variation, or on the

length of the cycle 'M', or on both. If the number of auxiliary parameters plus the number of parameters of the original network is less than those needed for Option 2, then the present scheme becomes more cost effective.

Instead of having one network to capture the dynamics of the signal and another for the parameters, we can have a single big augmented network where the time or the stage variable enters explicitly as one of the arguments. Then, the time varying system can be represented by, $x(k+1) = f_{aug}(x(k), u(k), W_{f_{aug}}, k)$, where the augmented parameters $W_{f_{aug}}$ are time invariant. With the introduction of the stage variable as an additional input, Option 3 can be considered as a special case of Option 2 for adaptation purposes with $p_M = p$.

In Option 2, we consider the time variation just as a sequence of parameters, with no relationship amongst them. Hence, we need to determine every element of it. But, in Option 3, we assume some relationship among the members of the sequence. The entropy of the sequence is thus reduced and hence fewer 'parameters' can be used to define the sequence. For example, if we know that the torque variation while turning a square rod is a square wave with unknown amplitude and frequency, the time variation of the torque can be captured with just two parameters. Certain other disturbances of specific types have been captured with smaller number of parameters (output of an unforced linear system), *e.g.*, Mukhopadhyay and Narendra[14].

4. Identification and adaptive control algorithms

Having discussed various methods of partial unfolding, we now develop update rules for the adaptation of identifier and controller parameters. As far as adaptation laws are concerned, Option 2 is the most general one and the other two options can be considered as special cases. So, update laws are developed below only for Option 2.

Let $\mathcal{J}$ be a general cost function and $W(k)$ be the representative set of parameters. Then the change in cost function due to changes in W(k) is given by,

$$\Delta\mathcal{J} = \sum_{k=0}^{M-1} \Delta^T W(k) \nabla_{W(k)} \mathcal{J} \tag{27}$$

If we have the condition, $W(k) = W(k+1) = \ldots = W(k+j-1)$ for $k = 0, j, 2j, \ldots, M-j$, then we have $\Delta W(k) = \Delta W(k+1) = \ldots = \Delta W(k+j-1)$. Hence (27) can be written as,

$$\Delta\mathcal{J} = \sum_{k=0,j,2j,\ldots,M-j} \left[\Delta^T W(k) \sum_{i=0}^{j-1} \nabla_{W(k+i)} \mathcal{J} \right] \tag{28}$$

Effectively, the sensitivity of a change in $W(k)$ on the cost is given by,

$$\Delta^{+}_{W(k)} \mathcal{J} = \sum_{i=0}^{j-1} \nabla_{W(k+i)} \mathcal{J} \tag{29}$$

where the superscript (+) is used to denote the effective gradient.

As before, consider the weight vectors W_{mf}, W_{mg}, W_{cf} and W_{cg} to be partitioned into $W_{(\cdot)_1}, W_{(\cdot)_2}, \ldots, W_{(\cdot)_M}$, corresponding to their rates of change. Then the adaptation laws for some 'j' are given by,

$$W_{mf_j}(k+M) = W_{mf_j}(k) - \mu \sum_{i=0}^{j-1} \nabla_{W_{mf_j(k+i)}} J_i \tag{30}$$

$$W_{mg_j}(k+M) = W_{mg_j}(k) - \mu \sum_{i=0}^{j-1} \nabla_{W_{mg_j(k+i)}} J_i \tag{31}$$

$$W_{cf_j}(k+M) = W_{cf_j}(k) - \mu \sum_{i=0}^{j-1} \nabla_{W_{cf_j(k+i)}} J_c \tag{32}$$

$$W_{cg_j}(k+M) = W_{cg_j}(k) - \mu \sum_{i=0}^{j-1} \nabla_{W_{cg_j(k+i)}} J_c \tag{33}$$

Various other control structures can also be handled under the mechanism of the output feedback structure assumed so far. For open loop control, the state mapping for the controller, *i.e.*, (7), has to be dispensed with and the output mapping of the controller should be replaced by $u(k) = y_c(k) = g(W_{cg}(k))$. For state feedback, again (7) has to be discarded from the model and the output mapping will look like $u(k) = y_c(k) = g_c(x_m(k), x_r(k), W_{cg}(k))$, where x_r is the reference state. However, no change is required as far as the sensitivity calculation or the update laws are concerned.

In many cases, the periodic variation is with respect to some abstract variable, $s = 0, 1, \ldots, S-1$ (say) rather than time. For example, in pick and place operation, the load carried by the gripper varies depending on whether it is moving in one direction or the other. In such cases, unfolding of the network has to be done with respect to the variable, s, over which the parameter varies, rather than over time. This complicates the problem since the adjoint can apportion the error only in time but not with respect to networks separated by some other abstract variable. However, adjoints do give all the necessary information which needs only to be used properly in such cases.

Let c be the running index for the cycle and we are looking for an update law which takes the parameters from $W(s, c)$ to $W(s, c+1)$. During that cycle, let $f_s(k)$ be a function representing the variation of this abstract variable, s, over the discretized time k. Then (27) can be rewritten as,

$$\Delta \mathcal{J} = \sum_{s=0}^{S-1} \left[\Delta^T W(s,c) \sum_{\{k/s=f_s(k)\}} \nabla_{W(k)} \mathcal{J} \right] \tag{34}$$

Proceeding in a similar manner, the update law, will look like,

$$W_{(\cdot)}(s,c+1) = W_{(\cdot)}(s,c) - \mu \sum_{\{k/s=f_s(k)\}} \nabla_{W(k)} \mathcal{J} \tag{35}$$

5. Simulation Results

In this section, we present the outputs of a simulation study which illustrate some of the theoretical developments presented above. A simple two-dimensional robot used for pick and place operation is considered. For simplicity, a semi-circular trajectory is taken where the robot picks an object at one end of the trajectory and drops it at another. Considerable time is spent at either of the ends so that the dynamic effects die out. The robot consists of two revolute joints each of length 1.25. The position of the objects, the robot and the trajectory are given in Fig. 5. The problem is characterised by a variable inertia (nonlinearity) primarily due to the construction of the machine and additionally due to the load picked (time variation). Note that the inertia change due to picking up of the load also changes with position.

In Fig.6, control of such a robot (under no load) with a simple P controller is shown. The response is quite good, but it should be noted that the machine moves slower towards the end of the trajectory due to the larger inertia faced at that position. Fig. 7 shows the deviation from the desired trajectory when the load is picked up. The return trajectory remains the same.

A standard mechanical load was chosen with the inertia In_i of the first motor being a function of the motor positions (ρ_1, ρ_2). The controllers were proportional controllers with the gain of the first motor K_1 being a function of the motor positions to compensate for the inertia changes.

$$\rho_1(s) = I_1(s)\frac{K_{T_1}}{s\left(In_1(\rho_1,\rho_2)s + B\right)} \tag{36}$$

$$\rho_2(s) = I_2(s)\frac{K_{T_2}}{s\left(In_2 s + B\right)} \tag{36}$$

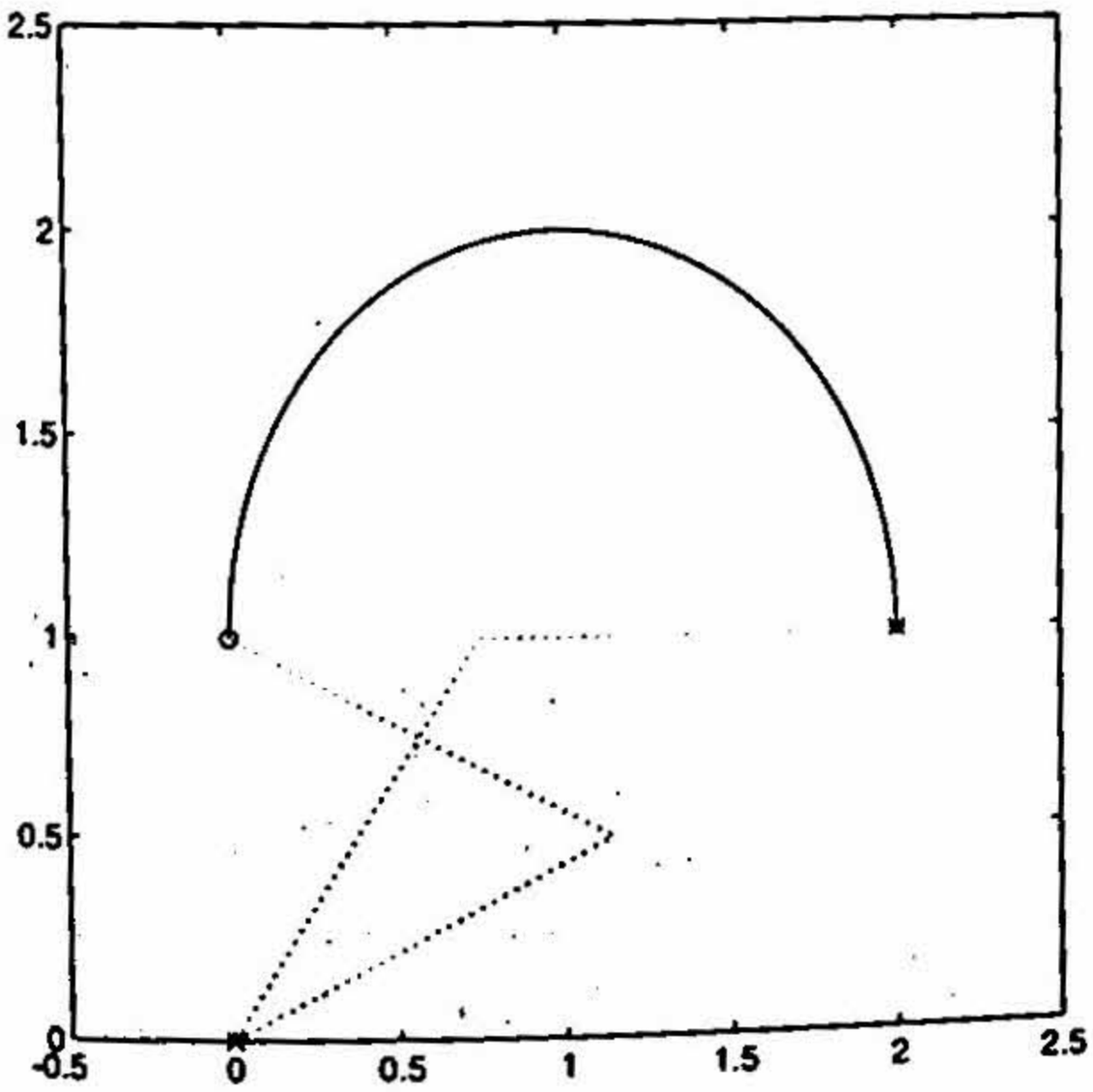

FIG. 5. The structure of the robot.

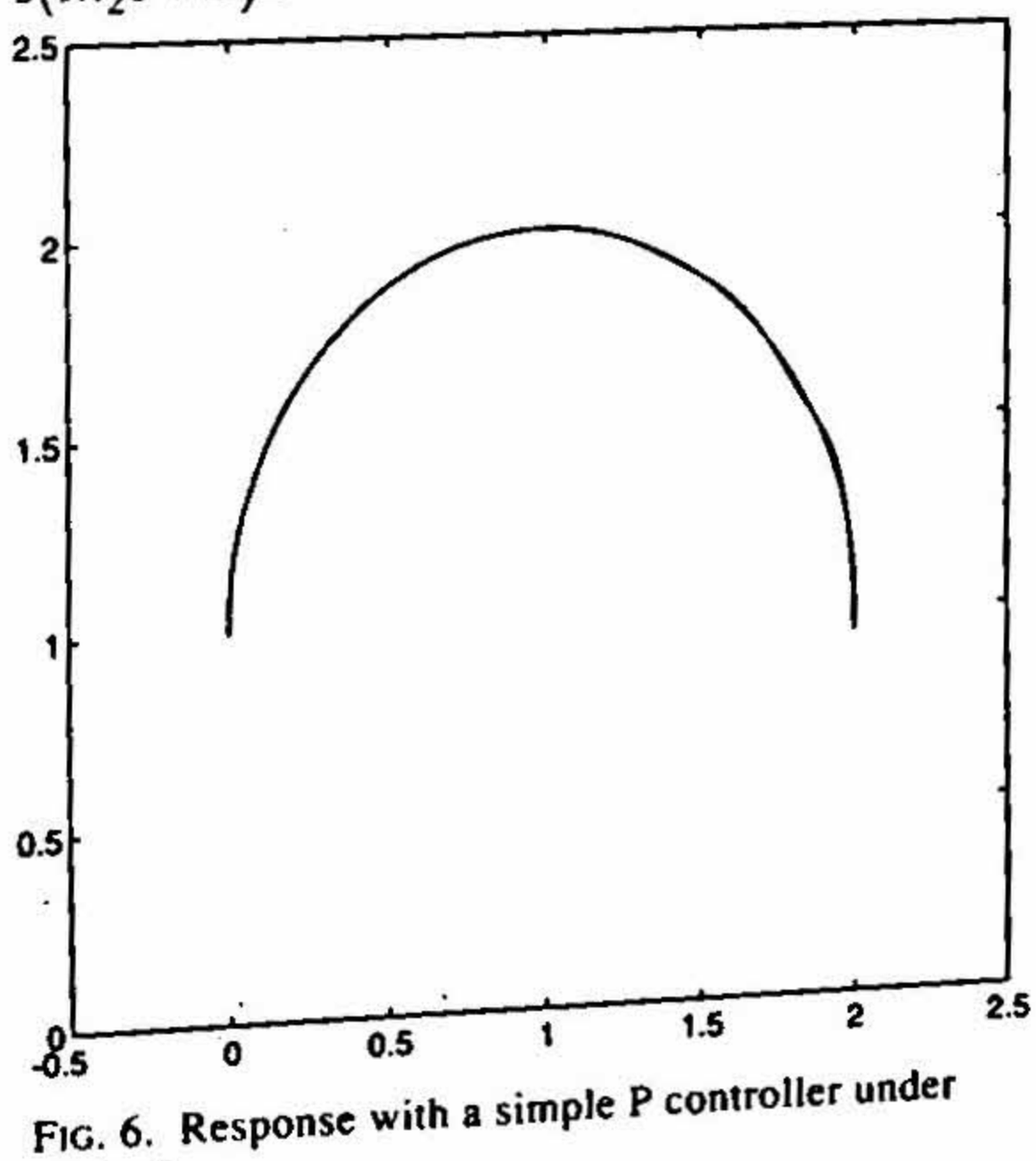

FIG. 6. Response with a simple P controller under no load.

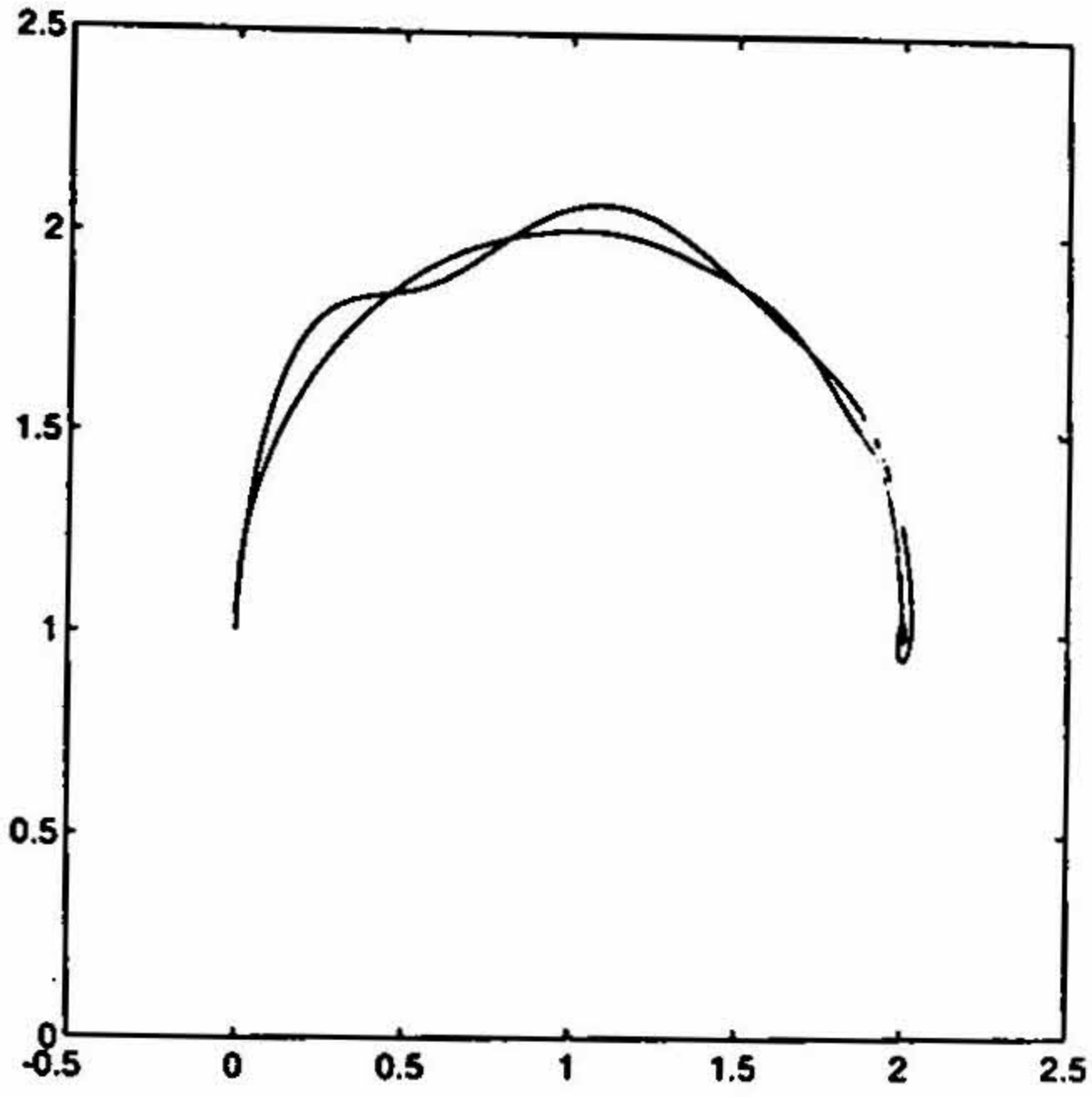

FIG. 7. Response with a P controller loaded in one direction.

$$I_1(s) = K_1(\rho_1, \rho_2)\left(\rho_{1_{ref}} - \rho_1\right) \tag{38}$$

$$I_2(s) = K_2\left(\rho_{2_{ref}} - \rho_2\right) \tag{38}$$

where B_i is the friction encountered, I_i the current supplied, $\rho_{i_{ref}}$ the reference position and K_{T_i} the torque constant of the ith motor, and s the Laplace operator.

The variations of the inertia and the proportional gain due to change in position was captured by neural networks. They were two input, single output networks with 2 hidden layers of 3 nodes each. Initially, no unfolding was done. After 200 runs of learning, this lead to fairly acceptable trajectories, except for the beginning of the pick operation and the end of the place operation. The simulation result is shown in Fig. 8. Then both $J_1(\cdot)$ and $K_1(\cdot)$ were partially unfolded. Two copies were held corresponding to the forward and backward runs (initially both copies were identical as was obtained from earlier experiment). Then, near-perfect trajectory tracking was seen after another 100 runs as shown in Fig. 9.

6. Conclusion

The sensitivities calculated by back-propagating the error through the adjoint neural model were used on-line without prediction in the class of systems which perform repetitive operations. The methods were directly applicable to time varying systems also. The error committed in any cycle was used to correct the parameters in the next cycle. It is shown that the condition for identifiability is violated if every parameter varies at the sampling rate. Hence, three schemes were proposed for partial unfolding of the network. Identification and adaptive control algorithms, for such partially unfolded networks were

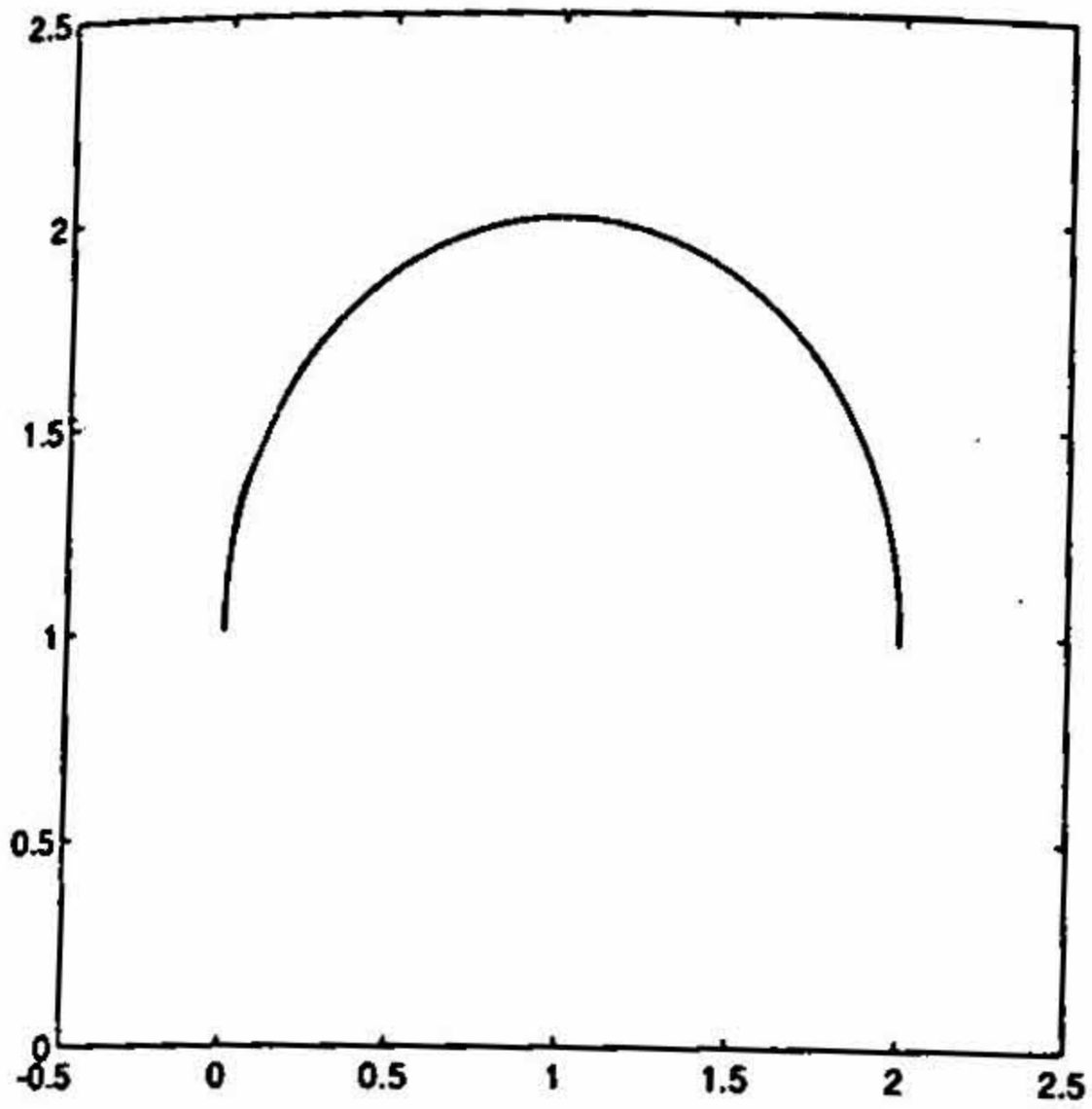

FIG. 8. Response with a neural network controller-with no unfolding in time.

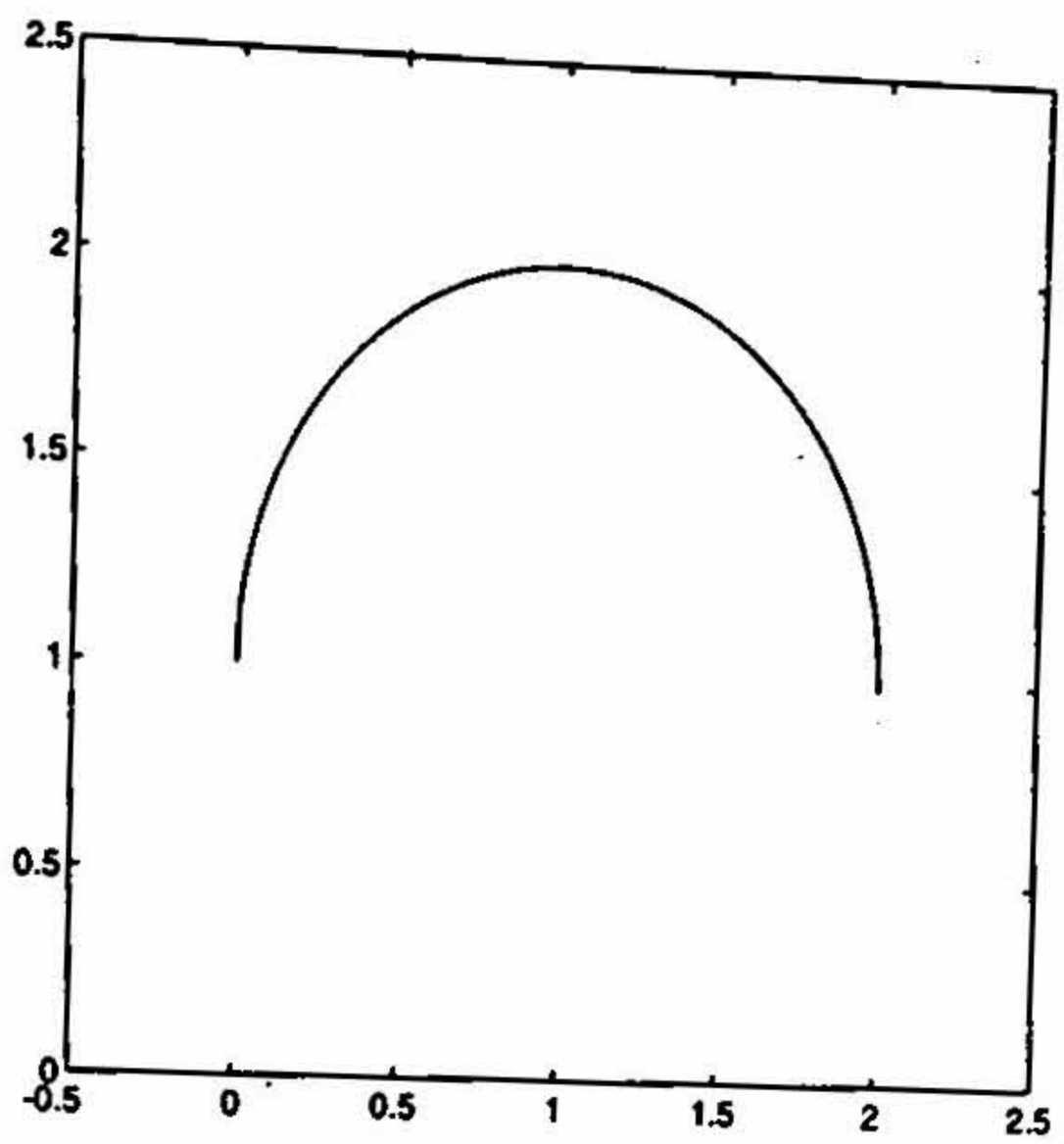

FIG. 9. Response with a neural network controller-with unfolding in time.

presented. The algorithms were tested by simulation of a two dimensional robot performing pick and place operation.

References

1. HUNT, R. J., SBARBARO, D., ZBIKOWSKI, R., AND GAWTHROP, P.J., — Neural networks for control systems - A survey, Automatica, 1992, **28**, 1083–1112.
2. NARENDRA, K. S. AND PARTHASARATHY, K. — Gradient methods for the optimization of dynamical systems containing neural networks, *IEEE Trans. on Neural Networks*, 1991, **2**, 252–262.
3. NGUYEN, D. AND WIDROW, B. — Neural networks for self-learning control systems, *IEEE Control Systems Magazine*, 1990, **10**, 18–23.
4. SRINIVASAN, B., PRASAD, U. R. AND Rao, N.J., — Back propagation through adjoints for the identification of non-linear dynamical systems using recurrent neural networks, *IEEE Trans. on Neural Networks*, 1994, **5**, 213–28.
5. SRINIVASAN, B., PRASAD, U. R. AND Rao, N. J, — Adjoint back propagation for indirect model reference adaptive control of nonlinear systems using neural networks, Submitted to *IEEE Trans. on Neural Networks*, 1995.
6. KAWATO, M., UNO, Y., ISOBE, M. AND SUZUKI, R., — Hierarchical neural network model for voluntary movement with application to robotics, *IEEE Control Systems Magazine*, 1988, **8**, 8–16.
7. LEVIN, A. U. AND NARENDRA, K. S. — Control of nonlinear dynamical systems using neural networks: controllability and stabilisation, *IEEE Trans. on Neural Networks*, 1993, **4**, 192–206.
8. BARTO, A. G., SUTTON, R. S., AND ANDERSON, C. W. — Neuron-like adaptive elements that can solve difficult learning control problems, *IEEE Trans. on Systems, Man, and Cybernetics*, 1983, **13**, 834–846.

9. SASTRY, S. S. AND BODSON, M. Adaptive control: Stability, convergence, robustness, 1989, Prentice Hall.

10. ASTROM, K. J. AND WITTENMARK, B. Adaptive control, 1989, Addsion-Wesley.

11. RUMELHART, D. E., HINTON, G. E. AND WILLIAMS, R. J. Learning internal representations by error propagation, Parallel distributed processing: Explorations in the microstructure of cognition, Ch. 8, vol. 1: Foundations, 1986, MIT Press.

12. BARTO, A. G. Connectionist learning for control: An overview, In Neural networks for control, (W. T. Miller, R. S. Sutton and P. J. Werbos eds), 1990, MIT press.

13. WALTER, E. Identifiability of Parametric Models, 1987, Pergamon Press.

14. MUKHOPADHYAY, S. AND NARENDRA, K. S. Disturbance rejection in nonlinear systems using neural networks", IEEE Trans. on Neural Networks, 1993, 4, 63–72.