Proceedings of the 38th
Conference on Decision & Control
Phoenix, Arizona USA • December 1999

Modelling of Nonlinear Stochastic Dynamical Systems using Neurofuzzy Networks

W.C. Chan, C.W. Chan, K.C. Cheung, Y. Wang
Department of Mechanical Engineering,
University of Hong Kong, Pokfulam Road, Hong Kong
Email: wcchan@hkumea.hku.hk

Abstract

Though nonlinear stochastic dynamical system can be approximated by feedforward neural networks, the dimension of the input space of the network may be too large, making it to be of little practical importance. The Nonlinear Autoregressive Moving Average model with eXogenous input (NARMAX) is shown to be able to represent nonlinear stochastic dynamical system under certain conditions. As the dimension of the input space is finite, it can be readily applied in practical application. It is well known that the training of recurrent networks using gradient method has a slow convergence rate. In this paper, a fast training algorithm based on the Newton-Raphson method for recurrent neurofuzzy network with NARMAX structure is presented. The convergence and the uniqueness of the proposed training algorithm are established. A simulation example involving a nonlinear dynamical system corrupted with the correlated noise and a sinusoidal disturbance is used to illustrate the performance of the proposed training algorithm.

Keywords: stochastic dynamical system, NARMAX model, neurofuzzy network, Newton-Raphson method.

1. Introduction

Neurofuzzy Networks are neural networks, where the output of the network can be considered to be derived from a set of linguistic rules. Hence, neurofuzzy network has the transparency of fuzzy systems [8,9]. Further, as neurofuzzy networks have the ability to approximate nonlinear function with arbitrary accuracy, they are widely used for modelling nonlinear dynamical systems [8,9,12]. To model stochastic dynamical systems using feedforward networks, it would require a network with a large number of inputs. The Nonlinear Autoregressive Moving Average model with eXogenous input (NARMAX) [5,13], represents stochastic dynamical system with finite order and can be readily implemented by recurrent networks.

In this paper, recurrent neurofuzzy networks [11] with NARMAX structure are presented. The recurrent network consists of two subnetworks: one representing the nonlinear dynamical system, and the other the contaminated correlated noise. The advantage of using the NARMAX structure is model parsimony, since the input dimension of NARMAX model is much smaller than that of the Nonlinear Autoregressive model with eXogenous input (NARX). Though the dynamic back propagation proposed by Narendra and Parthasarathy is suitable for training the recurrent network [11], fast convergence rate cannot always be obtained. A new training algorithm with a faster convergence rate is derived based on the Newton-Raphson method [9,10]. In this algorithm, the concept of the sensitivity model in [11] is used to establish the relationship between the partial derivatives of the output error and the basis functions of the recurrent neurofuzzy network.

In §2, the nonlinear stochastic dynamical system and the NARMAX model are introduced. Then, the recurrent neurofuzzy network with NARMAX structure is presented in §3. In training the network, the new training algorithm is devised in §4. The convergence analysis is given in §5. In §6, the fast convergence rate of the new training algorithm and the unbiased estimates are illustrated by examples of nonlinear dynamical systems corrupted with the correlated noise and a sinusoidal disturbance.

2. Nonlinear Stochastic Dynamical Systems

Consider a nonlinear stochastic dynamical system [5,13] given by,

$$y(k) = f_S\left(k, y^{k-1}, u^{k-1}\right) + \varepsilon(k) \tag{1}$$

where y(k), u(k), and $\varepsilon(k) \sim N(0,\sigma^2)$ are respectively the output, the input and the white noise, and $f_S(.)$ is a nonlinear function. Let

$$y^k = [y(k),..., y(0)]^T, \quad u^k = [u(k),..., u(0)]^T$$

Since y(k) is a function of $\varepsilon(k)$, a random variable and hence y^k is also a random variable. The conditional probability density function of y^k given u^k, $g(y^k|u^{k-1})$, characterizes completely the nonlinear stochastic dynamical system. By Bayes's rule, the conditional probability density function of the output y(k) is: $g\left(y(k)|y^{k-1}, u^{k-1}\right)$, and the conditional mean of y(k) given u^{k-1} and y^{k-1} is

$$\hat{y}(k) = E[y(k)|y^{k-1}, u^{k-1}] = f_M(k, y^{k-1}, u^{k-1}) \tag{2}$$

In prediction-error form, y(k) is given by,

$$y(k) = \hat{y}(k) + e(k) \tag{3}$$

where e(k) is the prediction error. Let

$$e^k = [e(k),..., e(0)]^T$$

Then e^{k-1} can be evaluated iteratively from y^{k-1} and u^{k-1} using (3). Similarly, y^{k-1} can be obtained from e^{k-1} and u^{k-1}. Therefore the information contained in (y^{k-1},u^{k-1}) is equivalent to that in (e^{k-1},u^{k-1}), hence

0-7803-5250-5/99/$10.00 © 1999 IEEE

$$g(y(k)|y^{k-1}, u^{k-1}) = g(y(k)|e^{k-1}, u^{k-1}) \quad (4)$$

Alternatively, the prediction of y(t) is obtained from,

$$\hat{y}(k) = E[y(k)|e^{k-1}, u^{k-1}] = f_M^*(k, e^{k-1}, u^{k-1}) \quad (5)$$

Indeed, the dimension of f_M and f_M^* increases as k increases, and thus the computation cost and memory storage will also increase. Based on (2) and (5), the system can be represented by a model with finite input dimension,

$$y(k) = F(y(k-1),..., y(k-m), u(k-1),..., u(k-p), e(k-1),..., e(k-n)) + e(k) \quad (6)$$

The model given by (6) is known as Nonlinear Autoregressive Moving Average model with eXogenous input (NARMAX). It is shown in [5,13] that a stochastic dynamical system can always be represented by the NARMAX model in the region around an equilibrium point subject to two sufficient conditions:

M1: The deterministic function $f_M : [U^{k-1}\ Y^{k-1}] \rightarrow Y$ of the system is finitely realizable (i.e. it has a state-space realization) where U^k and Y^k are respectively the sets of all sequence (u(1),u(2),...,u(k)), (y(1),y(2),...,y(k)) and Y is the output set.

M2: A linearized model exists if the system is operating close to the specified equilibrium point.

3. Recurrent Neurofuzzy Networks

B-spline neurofuzzy networks [8] are often used to estimate the output of nonlinear systems for given input and output measurement. Let x(k) be the input of the network. It is shown in [8,9] that under certain conditions, the output of a single hidden layer B-spline neural network is a linear function of the weights of the hidden layer $\{w_i, i=1,...,q\}$, and the transformed inputs $\{s_i(x(k)), i=1,...,q\}$,

$$\hat{y}(k) = \sum_{i=1}^{q} w_i s_i(x(k)) \quad (7)$$

The transformed input $s_i(x)$ is obtained by tensor product from the following k^{th} order B-spline basis function [7],

$$s_k^j(x) = \left(\frac{x - x_{j-k}}{x_{j-1} - x_{j-k}}\right) s_{k-1}^{j-1}(x) + \left(\frac{x_j - x}{x_j - x_{j-k+1}}\right) s_{k-1}^{j}(x) \quad (8)$$

$$s_1^j(x) = \begin{cases} 1 & \text{if } x \in I_j \\ 0 & \text{otherwise} \end{cases}$$

where x_j is the j^{th} knot defined on X, and $I_j \subset X$, is the j^{th} interval, $[x_{j-1}, x_j)$. Let X(k) and Z(k) be defined as

$$X(k) = [y(k-1),...,y(k-m),u(k-1),...,u(k-p)]^T$$
$$Z(k) = [e(k-1),...,e(k-n)]^T \quad (9)$$

Assuming that the system is corrupted with the correlated noise, the NARMAX model can be rewritten by the sum of two neurofuzzy networks,

$$y(k) = \sum_{i=1}^{m'} \alpha_i a_i(X(k)) + \sum_{i=1}^{n'} \beta_i b_i(Z(k)) + e(k) \quad (10)$$

where m' and n' are the numbers of weights; $a_i(X(k))$ and $b_i(Z(k))$ are the tensor products obtained from (8); α_i and β_i are the corresponding parameters. Note that X(k), Z(k), m' and n' should be chosen correctly because they affect the networks' approximation capability. Hence, the prediction of y(k) can be represented by the output of the following recurrent neurofuzzy network,

$$\hat{y}(k) = \sum_{i=1}^{m'} \alpha_i a_i(X(k)) + \sum_{i=1}^{n'} \beta_i b_i(Z(k)) = r^T(k)\hat{\theta}$$
$$Z(k) = \begin{bmatrix} y(k-1) \\ \vdots \\ y(k-n) \end{bmatrix} - \begin{bmatrix} \hat{y}(k-1) \\ \vdots \\ \hat{y}(k-n) \end{bmatrix} \quad (11)$$

where $r(k) = [a_1(k)...a_{m'}(k)\ \ b_1(k)...b_{n'}(k)]^T$

$$\hat{\theta} = [\alpha_1...\alpha_{m'}\ \ \beta_1...\beta_{n'}]^T$$

The training of the network involves estimating the weights $\hat{\theta}$, as discussed in the next section.

4. Training Algorithm for Neurofuzzy Networks

Following [1], it is assumed that the system given by (1) satisfies the following assumptions.

S1: Let $\varepsilon^k = [\varepsilon(k),...,\varepsilon(0)]^T$ and Ω^k be the σ–algebra generated by (ε^k, u^k). Then $E[\varepsilon(k) | \Omega^{k-1}] = 0$.

S2: $E[\varepsilon^k \cdot \varepsilon^{kT} | \Omega^{k-1}] \geq \delta I, \quad \delta > 0.$

Let θ be a parameter vector, and D_M, a compact set. The model set M is denoted by

$$M = \{M(\theta) | \theta \in D_M\} \quad (12)$$

The proposed training algorithm is aimed at finding a least mean square estimator from the models defined in M. Since $\varepsilon(k)$ is zero mean, the neurofuzzy network can be trained using N measured input and output data, {y(1),...,y(N), u(1),...,u(N)} by minimizing the following cost function,

$$J(\hat{\theta}) = \frac{1}{N} \sum_{k=1}^{N} L(\hat{\theta}, e(k)) \quad (13)$$

where L(.) is a scalar function of e(k) defined by

$$L(\hat{\theta}, e(k)) = \frac{1}{2} e^2(k) \quad (14)$$

As the neurofuzzy network is a recurrent network, the estimate of its weights $\hat{\theta}$ can only be obtained numerically by minimizing (13) using hill-climbing methods. Though gradient method [8,9,11] is popular, it is not too suitable here as the weights of the neurofuzzy networks given by (11) are from two networks whose inputs can be quite different in magnitude, leading to slow convergence rate. Instead, Newton-Raphson method is used here, as it has a faster convergence rate when the estimate is close to the optimum solution [10]. For convenience, let $\hat{\theta}$ and r(k) in (11) be denoted by

$$\hat{\theta} = [\theta_1 \quad \theta_2 \quad ... \quad \theta_{m'+n'}]^T \quad (15)$$

$$r(k) = [r_1(k) \quad r_2(k) \quad ... \quad r_{m'+n'}(k)]^T \quad (16)$$

The inverse model of (10) is given by,

$$e(k) = y(k) - r^T(k)\hat{\theta} \quad (17)$$

The estimate of θ using Newton-Raphson method is computed as follows,

$$\theta(t+1) = \theta(t) - H^{-1}G \quad (18)$$

where G and H are respectively the gradient vector and the Hessian matrix of the cost function $J(\theta)$ as given below.

$$G_i(\theta)=\frac{\partial J(\theta)}{\partial \theta_i},\quad H_{i,j}(\theta)=\frac{\partial^2 J(\theta)}{\partial \theta_i \partial \theta_j} \tag{19}$$

Differentiating the cost function given by (13), G and H can be written as,

$$G=\frac{1}{N}\Phi^T E,\quad H=\frac{1}{N}\Phi^T\Phi+\frac{1}{N}S \tag{20}$$

where Φ and S are,

$$\Phi_{i,j}=\frac{\partial e(i)}{\partial \theta_j},\quad S_{i,j}=\sum_{k=1}^{N} e(k)\frac{\partial^2 e(k)}{\partial \theta_i \partial \theta_j} \tag{21}$$

Hence, (18) becomes,

$$\theta(t+1)=\theta(t)-(\Phi^T\Phi+S)^{-1}\Phi^T E \tag{22}$$

where $E(t)=[e(1),e(2),...,e(N)]^T$. In the feedforward neurofuzzy networks, the derivatives of the output error e(t) with respect to the weights are simply the corresponding basis functions. However, the calculation of these derivatives in the recurrent networks is not so straightforward. Here, the derivatives are derived based on the dynamic back propagation introduced in [11]. Define

$$d(k,j)=\begin{cases}\dfrac{\partial r^T(k)\theta}{\partial e(j)}, & k-1\ge j\ge k-n\\ 1, & k=j\\ 0, & \text{otherwise}\end{cases} \tag{23}$$

Calculating d(k,j) involves the derivative of B-spline function. Assuming that the initial conditions of the first and second partial derivatives are zero, differentiating (17) with respect to θ_i, yields,

$$\frac{\partial e(k)}{\partial \theta_i}=-r_i(k)-\sum_{j=k-1}^{k-n}\frac{\partial r^T(k)\theta}{\partial e(j)}\bullet\frac{\partial e(j)}{\partial \theta_i} \tag{24}$$

The first partial derivative $\frac{\partial e(k)}{\partial \theta_i}$ is regarded as the output of a linear dynamic model, called sensitivity model [11]. From (23), (24) can be rearranged as,

$$\sum_{j=1}^{N} d(k,j)\frac{\partial e(j)}{\partial \theta_i}=-r_i(k) \tag{25}$$

The Jacobian matrix Φ is the solution of (25) in matrix form.

$$D\Phi=-R \tag{26}$$

where D and R are given by

$$D=\begin{bmatrix} d(1,1) & \cdots & d(1,N)\\ \vdots & \ddots & \\ d(N,1) & & d(N,N)\end{bmatrix},\quad R=\begin{bmatrix} r_1(1) & \cdots & r_{m'+n'}(1)\\ \vdots & \ddots & \\ r_1(N) & & r_{m'+n'}(N)\end{bmatrix}$$

From the definition of d(k,j), it is easy to observe that D is a lower triangular square matrix with unity diagonal element. For large N, Φ can be obtained by direct substitution instead of inverting D, since D is triangular. Define

$$c_i(k,j)=\begin{cases}\dfrac{\partial r_i(k)}{\partial e(j)}, & i=m'+1,...,m'+n' \text{ and } k-1\ge j\ge k-n\\ 0, & \text{otherwise}\end{cases}$$

Differentiating (25) with respect to θ_i, a sensitivity model of the second partial derivative is obtained.

$$\sum_{j=1}^{N} d(k,j)\frac{\partial^2 e(j)}{\partial \theta_h \partial \theta_i}=-\sum_{j=1}^{N} c_i(k,j)\cdot\frac{\partial e(j)}{\partial \theta_h} \tag{27}$$

then

$$DV(i)=-C(i)\Phi \tag{28}$$

where V(i) and C(i) are given by,

$$V_{k,h}(i)=\frac{\partial^2 e(k)}{\partial \theta_h \partial \theta_i},\quad C(i)=\begin{bmatrix} c_i(1,1) & \cdots & c_i(1,N)\\ \vdots & \ddots & \\ c_i(N,1) & & c_i(N,N)\end{bmatrix}$$

Matrix C(i) is a lower triangular square matrix with zero diagonal elements. Direct substitutions can be applied again to solve (28). Hence, matrix S can be obtained from,

$$S=[V^T(1)E\ \cdots\ V^T(m'+n')E]$$

5. Convergence Analysis

The convergence analysis of the estimate θ of using the algorithm derived in section 4 follows closely to that given in [1].

Lemma 1. The function $L(\hat{\theta},e(k))$ given by (14) satisfies the following two regularity conditions [1].

C1: $\left|\frac{\partial}{\partial e}L(\theta,e)\right|\le C|e|$; $\theta\in D_M$

C2: $\left|\frac{\partial}{\partial \theta}L(\theta,e)\right|\le C|e|^2$; $\theta\in D_M$

Proof: Differentiating the function $L(\theta,e(k))$ with respect to e(k), gives,

$$\frac{\partial}{\partial e(k)}L(\theta,e(k))=e(k) \tag{29}$$

Clearly, condition C1 is satisfied for some $C\ge 1$. The first derivative of $L(\theta,e(k))$ with respect to θ is the product of e(k) and the first derivative of e(k) with respect to θ, i.e.,

$$\left|\frac{\partial}{\partial \theta}L(\theta,e(k))\right|=|e(k)|\left|\frac{\partial e(k)}{\partial \theta}\right| \tag{30}$$

As θ is defined as the weight vector in D_M, the prediction error is exponentially stable. Since the derivative, $\frac{\partial e(k)}{\partial \theta}$, is assumed to be bounded by

$$\left|\frac{\partial e(k)}{\partial \theta}\right|\le C|e(k)| \tag{31}$$

then condition C2 is also satisfied. □

Since the input and output are random variables, $J(\hat{\theta})$ and $\hat{\theta}$ are also random variables. It is assumed that the expected value of $J(\hat{\theta})$ exists and is denoted by $\bar{J}(\hat{\theta})$.

Theorem 1. The estimate of the cost function, $J(\hat{\theta})$ given by (13), has the following property,

$$\sup_{\theta\in D_M}\left|J(\hat{\theta})-\bar{J}(\hat{\theta})\right|\to 0 \text{ w.p.1 as } N\to\infty \tag{32}$$

Proof: Consider the supremum of the sum over small open spheres given by

$$B(\theta^*,\rho)=\left\{\theta\,\middle|\,|\theta-\theta^*|<\rho\right\} \tag{33}$$

Let D be the open neighborhood of D_M. Choose $\theta^* \in D_M$, let $B = B(\theta^*,\rho) \cap D$ and consider

$$\sup_{\theta\in B}\frac{1}{N}\sum_1^N\left[L(\theta,e(k))-EL(\theta,e(k))\right]\le\frac{1}{N}\sum_1^N\eta(k) \quad (34)$$

where

$$\eta(k)=\sup_{\theta\in B}L(\theta,e(k))-EL(\theta,e(k))$$

Let $\{x(k)\}$ be a sequence of random variables with zero-mean values and

$$\left|Ex(k)x(h)\right|\le C\frac{k^p+h^p}{1+\left|k-h\right|^q} \qquad 0\le 2p<q<1 \quad (35)$$

Then $\frac{1}{N}\sum_{k=1}^N x(k)\to 0$ w.p.1 as $N\to\infty$.

Let k>h, and

$$e_h(k,\theta)=y_h(k)-f_M(\theta;k,y_h^{k-1},u_h^{k-1})$$

where

$$y_h^{k-1}=(y_h(k-1),...,y_h(h+1),0,...,0)$$

$$u_h^{k-1}=(u_h(k-1),...,u_h(h+1),0,...,0)$$

Note that $y_h(j)$ and $u_h(j)$ are random variables belonging to Ω^j. For convenience, let $y_h(j)=0$ for $j\le h$. Let

$$\eta_h(k)=\sup_{\theta\in B}L(\theta,e(k))-EL(\theta,e(k))$$

As $\eta(k)$ is independent of $\eta(h)$ for $h\neq k$, the covariance of $\eta(k)$ and $\eta(h)$ is,

$$\begin{aligned}Cov[\eta(k),\eta(h)]&=Cov[\eta(k)-\eta_h(k),\eta(h)]\\&\le\left[E\eta^2(h)E(\eta(k)-\eta_h(k))^2\right]^{1/2}\end{aligned} \quad (36)$$

Condition C1 given in Lemma 1 implies that

$$\begin{aligned}\left|\eta(k)-\eta_h(k)\right|&\le\sup_{\theta\in B}\left|L(\theta,e(k))-L(\theta,e_h(k))\right|\\&\le C\sup_{\theta\in B}\left\{\left|e(k)\right|+\left|e_h(k)\right|\right\}\sup_{\theta\in B}\left|e(k)-e_h(k)\right|\\&\le C\left[\sum_{j=0}^k\lambda^{k-j}\left\{\left|y(j)\right|+\left|y_h(j)\right|+\left|u(j)\right|+\left|u_h(j)\right|\right\}\right]\\&\quad\bullet\left[\sum_{j=0}^k\lambda^{k-j}\left\{\left|y(j)-y_h(j)\right|+\left|u(j)-u_h(j)\right|\right\}\right]\end{aligned} \quad (37)$$

From the Schwarz' inequality and assumption S1,

$$E\left|\eta(k)-\eta_h(k)\right|^2\le C\lambda^{k-h} \quad (38)$$

As $\eta(k)$ satisfies (35), it implies that

$$\frac{1}{N}\sum_1^N(\eta(k)-E\eta(k))\to 0 \quad \text{w.p.1 as } N\to\infty \quad (39)$$

The expected value of $\eta(k)$ is bounded and is given by

$$\begin{aligned}E[\eta(k)]&=E\sup_{\theta\in B}L(\theta,e(k))-EL(\theta,e(k))\\&\le E\sup_{\theta\in B}L(\theta,e(k))-L(\theta^*,e(k))\\&\quad+\sup_{\theta\in B}E[L(\theta^*,e(k))-L(\theta,e(k))]\\&\le\left[E\sup_{\theta\in B}\left|\frac{d}{d\theta}L(\theta,e(k))\right|+\sup_{\theta\in B}E\left|\frac{d}{d\theta}L(\theta,e(k))\right|\right]\bullet\left[\theta-\theta^*\right]\\&\le C^*\rho\end{aligned} \quad (40)$$

where C^* is a constant according to the scalar function L(.) and ρ is radius of the open sphere B. Substituting this inequality into (34) gives

$$\begin{aligned}&\sup_{\theta\in B}\frac{1}{N}\sum_{k=1}^N\left[L(\theta,e(k))-EL(\theta,e(k))\right]\\&\le\frac{1}{N}\sum_{k=1}^N(\eta(k)-E\eta(k))+C^*\rho\end{aligned} \quad (41)$$

From (39), the first term of (41) vanishes as N tends to infinity. Hence,

$$\sup_{\theta\in B}\frac{1}{N}\sum_{k=1}^N\left[L(\theta,e(k))-EL(\theta,e(k))\right]<\delta(N) \quad (42)$$

where $\delta(N)>0$ and it takes arbitrarily small value for large N. Equation (32) implies that $J(\hat{\theta})$ is arbitrary close to the minimum of $\bar{J}(\hat{\theta})$ as N increases, giving the convergence property of the estimate, $\hat{\theta}$. □

Let $\bar{\theta}$ be the true weight vector. It is shown below that the unique minimum of $\bar{J}(\hat{\theta})$ is $\bar{J}(\bar{\theta})$.

Theorem 2. $\bar{J}(\hat{\theta})$ satisfies the following property,

$$\bar{J}(\hat{\theta})>\bar{J}(\bar{\theta})$$

Proof: Let $\bar{\theta}$ be the true weights of the B-spline neural network [8,12], i.e.,

$$\bar{y}(k)=f(X(k))+g(Z(k))\equiv r^T(k)\bar{\theta} \qquad \bar{\theta}\in D_M \quad (43)$$

The difference between the cost functions with the true weight vector and another weight vector in D_M can be written as,

$$\begin{aligned}\bar{J}(\hat{\theta})-\bar{J}(\bar{\theta})&=\frac{1}{2N}\sum_{k=1}^N E\left[e^2(k)-\varepsilon^2(k)\right]\\&=\frac{1}{2N}\sum\left\{E\left[2\varepsilon(k)(e(k)-\varepsilon(k))\right]+E\left[(e(k)-\varepsilon(k))^2\right]\right\}\\&=\frac{1}{2N}\sum\left\{E\left[2\varepsilon(k)(\bar{y}(k)-\hat{y}(k))\right]+E\left[(\bar{y}(k)-\hat{y}(k))^2\right]\right\}\end{aligned} \quad (44)$$

As $\varepsilon(k)$ is a white noise sequence independent of the past inputs and outputs, the first term on the right hand side of (44) is zero and (44) becomes,

$$\bar{J}(\hat{\theta})-\bar{J}(\bar{\theta})=\frac{1}{2N}\sum_1^N E\left[(\bar{y}(k)-\hat{y}(k))^2\right]>0,\hat{\theta}\neq\bar{\theta} \quad (45)$$

giving $\bar{J}(\hat{\theta})>\bar{J}(\bar{\theta})$. Hence $\bar{\theta}$ is the unique minimum. □

6. Training Procedure

The training procedure of the recurrent neurofuzzy network given by (11) is presented below.

(i) Select suitable input variables X(k) and Z(k), m' and n', and the initial values of θ, $\theta(0)$.

(ii) Compute $E=[e(1),...,e(N)]^T$ using the inverse model given by (17), for k=1,...,N.

(iii) Compute Φ and S in (21) using the respective sensitivity models.

(iv) Update the parameter θ using (22).

(v) Repeat (ii) to (iv). Terminate the iteration process if

$$\left| \frac{\Delta J(\hat{\theta})}{J(\hat{\theta})} \right| < \varepsilon \text{, say } \varepsilon = 10^{-3}.$$

7. Example

Consider the following nonlinear system [8],

$$\begin{aligned} y(k) = &\left(0.8 - 0.5e^{-y^2(k-1)}\right) y(k-1) \\ &- \left(0.3 + 0.9e^{-y^2(k-1)}\right) y(k-2) \\ &+ 0.1\sin(\pi y(k-1)) + \eta(k) \end{aligned} \tag{46}$$

where $\eta(k)$ is a disturbance. For $\eta(k) = 0$, and initial conditions of y(k-1) and y(k-2) are [0.1,0.1], y(k) spirals out from an unstable equilibrium at the origin towards a periodic attractor, as shown in Fig. 1(a).

(i) Correlated noise

Let $\eta(t)$ be a correlated noise given by

$$\eta(k) = (1 - \lambda z^{-1})\varepsilon(k) \tag{47}$$

where $\varepsilon(k) \sim N(0, 0.1^2)$ and $|\lambda| \leq 1$. From (46) and (47), m=2 and n=1. The range of y(k) is ±1.5. Following [12], two second order basis functions are used for y(k-2), and four third order basis functions for y(k-1). Two second order basis functions is used for e(k). Three values of λ are selected: 0.5, 0.8 and 1. 1000 training data are generated using (46). The estimate of the weights converges in about 10 iterations. The estimated outputs from the networks are shown in Figs. 1(b), 1(c) and 1(d), showing good estimates of the output.

(ii) Sinusoidal disturbance

Consider next $\eta(k)$ given by [12],

$$\eta(k) = 0.5\sin(0.1k) \tag{48}$$

As $\eta(k)$ is a second order system, it can be modelled by

$$\eta(k) = g(\eta(k-1), \eta(k-2)) + g'(e(k-1)) \tag{49}$$

The system given by (46) is now modelled by the following recurrent network.

$$\begin{aligned} \hat{y}(k) &= f(y(k-1), y(k-2)) + \eta(k) \\ &= f(y(k-1), y(k-2)) + g(\eta(k-1), \eta(k-2)) \\ &\quad + g'(e(k-1)) \end{aligned} \tag{50}$$

Second order basis functions are used in g(.) and g'(.), whilst that for f(.) are the same as used previously. $\eta(k-1)$ and $\eta(k-2)$ are obtained from equation (49) and $e(k-1) = y(k-1) - \hat{y}(k-1)$. In training the networks, the same initial values are used for f(.). However, care must be exerted in choosing the initial values of the weights of the network g(.) for $\eta(k)$. If the initial values are too far from the actual value, the training may stop at a local minimum, as the network may be estimating the harmonics of the sinusoidal disturbance instead. In this example, the initial values of the weights of g(.) are chosen to be: {-1.4747, -4.4747, 4.4747, 1.4747}, generating signal whose frequency is 30% higher than the actual frequency. The results, shown in Fig. 2, are comparable to that obtained in [12], though it is assumed in [12] that the frequency of the sinusoidal disturbance is known.

8. Conclusion

Recurrent neurofuzzy network with NARMAX structure for modelling nonlinear stochastic dynamical systems is presented. To improve the convergence rate of the estimated weights of the network, a training algorithm based on Newton-Raphson method is derived. The convergence of the proposed training algorithm is established and its performance is illustrated by a simulation example involving a nonlinear stochastic dynamical system. From the simulation example, it is shown that the proposed training algorithm converges in a small number of iterations.

Acknowledgement

The work was supported by the Research Grants Council of Hong Kong.

References

[1] Ljung, L., "Convergence Analysis of Parametric Identification Methods", *Trans. IEEE on Automatic Control*, AC-23, 5, 1978.

[2] Ljung, L., "Analysis of a General Recursive Prediction Error Identification Algorithm", *Automatica*, 17, 89-99, 1981.

[3] Ljung, L., *System Identification: Theory for the User*, Prentice Hall, Englewood Cliffs, 1987.

[4] Goodwin, G. C., and K. S. Sin, *Adaptive Filtering Prediction and Control*, Prentice Hall, Englewood Cliffs, 1984.

[5] Chen, S., and S. A. Billings, "Representations of non-linear systems: the NARMAX model", *Int. J. Control*, 49, 1013-1032, 1989.

[6] Chen, S., and S. A. Billings, "Neural networks for nonlinear dynamic system modelling and identification", *Int. J. Control*, 56, 319-346,1992.

[7] Piegl, L., and W. Tiller, *The NURBS Book*, Springer-Verlag, Berlin Heidelberg, 1995.

[8] Brown, M., and C. J. Harris, *Neurofuzzy Adaptive Modelling and Control*, Prentice Hall, 1994.

[9] Jang, J. -S. R., C. –T. Sun, and E. Mizutani, *Neuro-fuzzy and Soft Computing: A Computational Approach to Learning and Machine Intelligence*, Prentice Hall, 1997.

[10] Åström, K. J., and T. Bohlin, "Numerical identification of linear dynamic systems from normal operating records", *IFAC Symp. Theory of Self-Adaptive Control Systems, Teddington, Engl. In Theory of Self-Adaptive Control Systems* (Ed. P. H. Hammond), Plenum Press, New York, 1966.

[11] Narendra, K. S., and K. Parthasarathy, "Gradient methods for the optimization of dynamical systems containing neural networks", *Trans. IEEE on Neural Networks*, 2, 252-262, 1991.

[12] Wang, H., M. Brown, and C. J. Harris, "Neural network based modelling of unknown systems subject to immeasurable disturbances", *Proc. IEE*, Pt. D, 141, 216-222, 1994.

[13] Leontaritis, I. J., and S. A. Billings, "Input-output parametric models for nonlinear systems Part II: stochastic nonlinear systems", *Int. J. Control*, 41, 329-344, 1985.

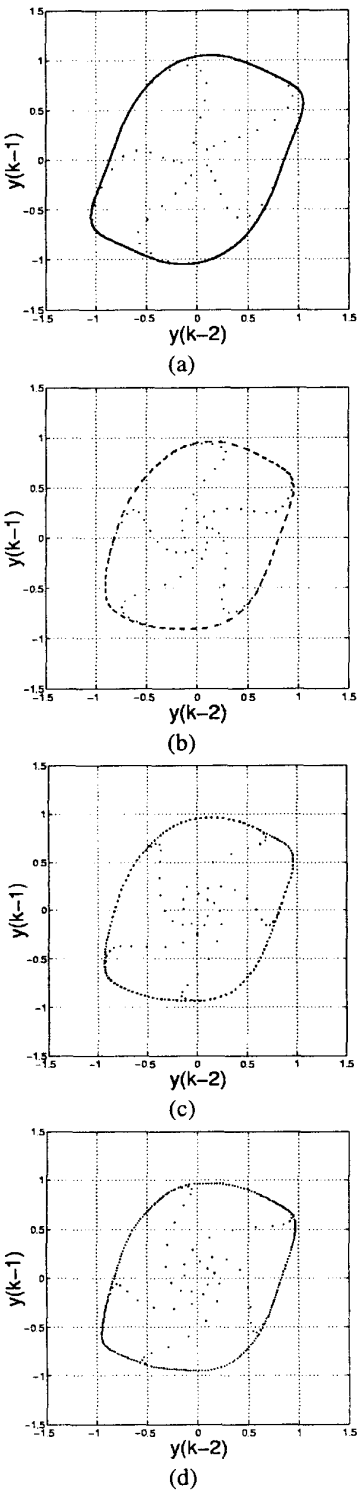

Fig. 1 (a) Output of nonlinear system for $\eta(k)=0$; (b), (c) and (d) Estimated outputs for $\lambda=0.5,0.8,1$ respectively.

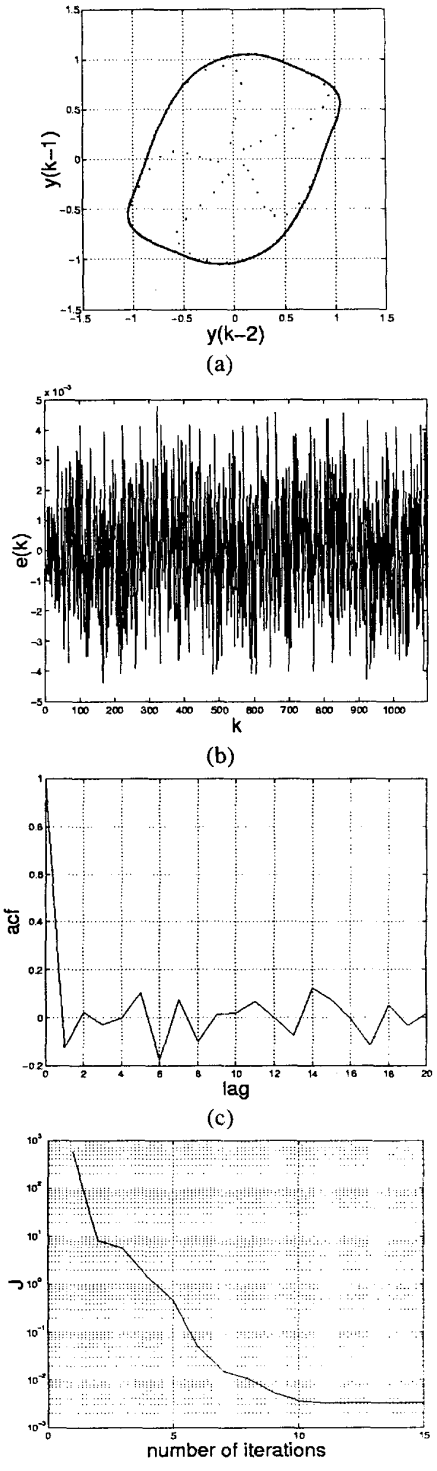

Fig. 2 (a) Estimated output; (b) Prediction error; (c) Auto-correlation function of e(k); and (d) Cost function.