International Journal of Control

ISSN: 0020-7179 (Print) 1366-5820 (Online) Journal homepage: https://www.tandfonline.com/loi/tcon20

Non-linear system identification using neural networks

S. CHEN , S. A. BILLINGS & P. M. GRANT

To cite this article: S. CHEN , S. A. BILLINGS & P. M. GRANT (1990) Non-linear system identification using neural networks, International Journal of Control, 51:6, 1191-1214, DOI: 10.1080/00207179008934126

To link to this article: https://doi.org/10.1080/00207179008934126

Published online: 27 Mar 2007.

INT. J. CONTROL, 1990, VOL. 51, NO. 6, 1191-1214

Non-linear system identification using neural networks

S. CHEN†, S. A. BILLINGS‡ and P. M. GRANT†

Multi-layered neural networks offer an exciting alternative for modelling complex non-linear systems. This paper investigates the identification of discrete-time non-linear systems using neural networks with a single hidden layer. New parameter estimation algorithms are derived for the neural network model based on a prediction error formulation and the application to both simulated and real data is included to demonstrate the effectiveness of the neural network approach.

1. Introduction

Both the theory and practice of non-linear system modelling has advanced considerably in recent years. It is known that a wide class of discrete-time non-linear systems can be represented by the non-linear autoregressive moving average with exogenous inputs (NARMAX) model (Leontaritis and Billings 1985, Chen and Billings 1989 b). The NARMAX model provides a description of the system in terms of a non-linear functional expansion of lagged inputs, outputs and prediction errors. The mathematical function describing a real-world system can be very complex and its exact form is usually unknown so that in practice modelling of a real-world system must be based upon a chosen model set of known functions. A desirable property for this model set is the capability of approximating a system to within an arbitrary accuracy. Mathematically, this requires that the set be dense in the space of continuous functions. Polynomial functions are one choice that have such a completeness property. This provides the foundation for modelling non-linear systems using the polynomial NARMAX model and several identification procedures based upon this model have been developed (Leontaritis and Billings 1988, Chen and Billings 1989 a, Chen *et al.* 1989). Because the derivation of the NARMAX model was independent of the form of the non-linear functional, other choices of expansion can easily be investigated within this framework and neural networks are an obvious alternative. Neural networks can therefore be viewed as just another class of functional representations.

Feedforward multi-layered neural networks have been widely used in many areas of signal processing (see the *I.E.E.E. Transactions*, 1988). A common feature in these applications is that neural networks are employed to realize some complex non-linear decision functions. Recent theoretical works (Cybenko 1989, Funahashi 1989) have rigorously proved that, even with only one hidden layer, neural networks can uniformly approximate any continuous function. The theoretical basis for modelling non-linear systems by neural networks is therefore sound.

The present study develops an identification procedure for discrete-time non-linear systems based on neural networks with a single hidden layer. New batch and

Received 28 August 1989.

† Department of Electrical Engineering, University of Edinburgh, Mayfield Road, Edinburgh EH9 3JL, Scotland, U.K.

‡ Department of Control Engineering, University of Sheffield, Mappin Street, Sheffield S1 3JD, England, U.K.

recursive estimation algorithms are derived for the neural network model based on the prediction error principle. It is shown that the classical back propagation algorithm is a special case of the new prediction error routines, and model validity tests are introduced as a means of measuring the quality of fit. The results of applying the neural network model to both simulated and real data are included and a suggestion for further research is also given.

2. System representation

Under some mild assumptions, a discrete-time multivariable non-linear stochastic control system with m outputs and r inputs can be represented by the multivariable NARMAX model (Leontaritis and Billings 1985):

$$\begin{aligned} y(t) = f\,(y(t-1), \ldots, y(t-n_y), u(t-1), \ldots, u(t-n_u), \\ e(t-1), \ldots, e(t-n_e)) + e(t) \end{aligned} \tag{1}$$

where

$$y(t) = \begin{bmatrix} y_1(t) \\ \vdots \\ y_m(t) \end{bmatrix}, \quad u(t) = \begin{bmatrix} u_1(t) \\ \vdots \\ u_r(t) \end{bmatrix}, \quad e(t) = \begin{bmatrix} e_1(t) \\ \vdots \\ e_m(t) \end{bmatrix} \tag{2}$$

are the system output, input and noise vectors, respectively; n_y, n_u and n_e are the maximum lags in the output, input and noise respectively; $e(t)$ is a zero-mean independent sequence; and $f(\cdot)$ is some vector-valued non-linear function.

The input–output relationship (1) is dependent upon the non-linear function $f(\cdot)$. In reality, $f(\cdot)$ is generally very complex and knowledge of the form of this function is often not available. The solution is to approximate $f(\cdot)$ using some known simpler function, and in the present study we consider using neural networks to approximate non-linear systems governed by the model

$$y(t) = f(y(t-1), \ldots, y(t-n_y), u(t-1), \ldots, u(t-n_u)) + e(t) \tag{3}$$

Notice that (3) is a slightly simplified version of (1) because only additive uncorrelated noise is considered. Extension of the results to the more general model description (1) is discussed.

3. Modelling by neural networks

Neural networks employed for function approximation are feedforward type networks with one or more hidden layers between the inputs and outputs. Each later consists of some computing units known as nodes. Figure 1 shows the structure of a multi-layer neural network. Inputs to the network are passed to each node in the first layer. The outputs of the first layer nodes then become inputs to the second layer, and so on. The outputs of the network are therefore the outputs of the nodes lying in the final layer. Usually all the nodes in a layer are fully connected to the nodes in adjacent layers, but there is no connection between nodes within a layer and no connecting bridging layers. The input–output relationship of each hidden node is determined by the connection weights w_i, a threshold parameter μ and the node activation function $a(\cdot)$, as follows:

$$y = a(\textstyle\sum w_i x_i + \mu) \tag{4}$$

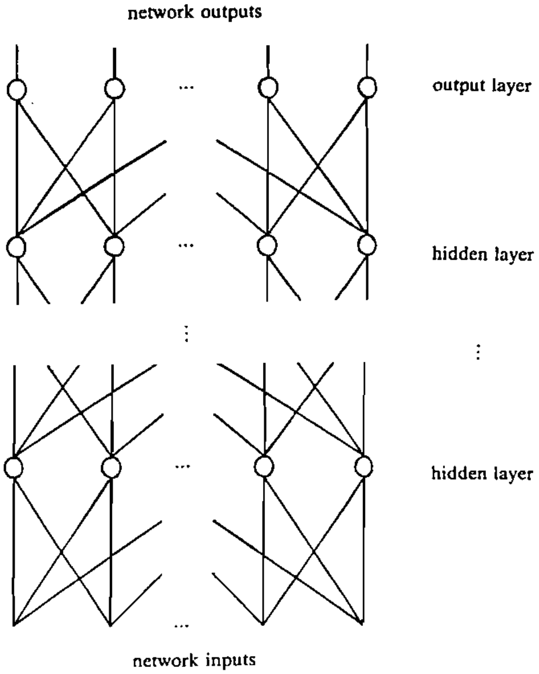


Figure 1. Multi-layer neural network.

where x_i are the node inputs and y is the node output. The activation function $a(\cdot)$ for each output node is specifically chosen to be linear, and the output node is the weighted sum of the inputs

$$y = \sum w_i x_i \tag{5}$$

The overall input–output relationship of an n-input m-output network with one or more hidden layers is described by a function $\hat{f}: \mathbb{R}^n \to \mathbb{R}^m$. Under very mild assumptions on the activation function $a(\cdot)$, it has been rigorously proved that any continuous function $f: D \subset \mathbb{R}^n \to \mathbb{R}^m$ can be uniformly approximated by an $\hat{f}$ on D, where D is a compact subset of $\mathbb{R}^n$ (Cybenko 1989, Funahashi 1989).

Our aim is to use neural networks with one hidden layer to model non-linear systems described by (3). Define $n = mn_y + rn_u$

$$\begin{aligned} x(t) &= [x_1(t) \quad \dots \quad x_n(t)]^{\mathrm{T}} \\ &= [y^{\mathrm{T}}(t-1) \quad \dots \quad y^{\mathrm{T}}(t-n_y) u^{\mathrm{T}}(t-1) \quad \dots \quad u^{\mathrm{T}}(t-n_u)]^{\mathrm{T}} \end{aligned} \tag{6}$$

and introduce the notation

n_h number of hidden nodes

$\mu_i^{(h)}$ threshold of ith hidden node

$w_{ij}^{(h)}$ connection weight from $x_j(t)$ to ith hidden node

$o_{hi}(t)$ output of ith hidden node

$w_{ki}^{(o)}$ connection weight from ith hidden node to kth output node

Let $\Theta = [\theta_1 \quad \dots \quad \theta_{n_\theta}]^{\mathrm{T}}$ be all the weights and thresholds of the network ordered in a chosen way. The network is then defined by the model

$$\hat{y}(t, \Theta) = \hat{f}(x(t); \Theta) = [\hat{f}_1(x(t); \Theta) \quad \dots \quad \hat{f}_m(x(t); \Theta)]^{\mathrm{T}} \tag{7}$$

with

$$\begin{aligned}\hat{y}_k(t, \Theta) = \hat{f}_k(x(t); \Theta) &= \sum_{i=1}^{n_h} w_{ki}^{(o)} o_{hi}(t) \\ &= \sum_{i=1}^{n_h} w_{ki}^{(o)} a\left(\sum_{j=1}^{n} w_{ij}^{(h)} x_j(t) + \mu_i^{(h)}\right), \quad 1 \leqslant k \leqslant m\end{aligned} \tag{8}$$

Without the loss of generality, the activation function $a(\cdot)$ will be chosen as

$$a(z) = \frac{1}{1 + \exp(-z)} \tag{9}$$

The network model (7) is therefore the one-step-ahead predictor for $y(t)$ and the prediction error or residual is given as usual by

$$\varepsilon(t, \Theta) = y(t) - \hat{y}(t, \Theta) \tag{10}$$

The first step in modelling non-linear systems using (3) is therefore to select values for n_y, n_u and n_h. The next is to determine values of all the weights and thresholds or to estimate Θ. The gradient of $\hat{y}(t, \Theta)$

$$\Psi(t, \Theta) = \left[\frac{d\hat{y}(t, \Theta)}{d\Theta}\right]^{\mathrm{T}} = \hat{g}(x(t); \Theta) \tag{11}$$

an $n_\theta \times m$ matrix, plays an important role in determining Θ. The combination of (7) and (11)

$$\begin{bmatrix} \hat{y}(t, \Theta) \\ \Psi(t, \Theta) \end{bmatrix} = \begin{bmatrix} \hat{f}(x(t); \Theta) \\ \hat{g}(x(t); \Theta) \end{bmatrix} \tag{12}$$

will be referred to as the extended network model. The stability of (12) is of vital importance in any implementation. The set of all Θ that each produce a stable extended network model is denoted as D_θ. Notice that, for the chosen activation function (9), D_θ is the whole n_θ-dimensional euclidean space and in this sense the corresponding extended network model is unconditionally stable. Furthermore, the elements of $\Psi(t, \Theta)$ for $1 \leqslant i \leqslant n_\theta$ and $1 \leqslant j \leqslant m$ are given by

$$\Psi_{ij}(t, \Theta) = \frac{d\hat{y}_j(t, \Theta)}{d\theta_i} = \begin{cases} o_{hk}(t) & \text{if } \theta_i = w_{jk}^{(o)},\ 1 \leqslant k \leqslant n_h \\ w_{jk}^{(o)} o_{hk}(t)(1 - o_{hk}(t)) & \text{if } \theta_i = \mu_k^{(h)},\ 1 \leqslant k \leqslant n_h \\ w_{jk}^{(o)} o_{hk}(t)(1 - o_{hk}(t)) x_l(t) & \text{if } \theta_i = w_{kl}^{(h)},\ 1 \leqslant k \leqslant n_h,\ 1 \leqslant l \leqslant n \\ 0 & \text{otherwise} \end{cases} \tag{13}$$

4. Identification algorithm

The network model (7) is non-linear in the parameters. This section applies the well-known prediction error estimation method to derive both the batch and recursive algorithms for estimating the parameter vector Θ in (7).

4.1. *Off-line prediction error algorithm*

A good measure of the closeness between $y(t)$ and $\hat{y}(t, \Theta)$ is the quadratic form

$$Q(\varepsilon(t, \Theta)) = \varepsilon^{\mathrm{T}}(t, \Theta)\Lambda^{-1}\varepsilon(t, \Theta) \tag{14}$$

where Λ is a given $m \times m$ symmetric positive definite matrix. Assume that a block of data $\{u(t), y(t)\}_{t=1}^{N}$ is available. The best Θ may then be selected by minimizing the loss function

$$J_1(\Theta) = \frac{1}{2N} \sum_{t=1}^{N} Q(\varepsilon(t, \Theta)) \tag{15}$$

over $\Theta \in D_\theta$. Such a method of obtaining Θ is known as the prediction error estimation method.

The minimization of criterion (15) can be performed efficiently using the following Gauss–Newton algorithm

$$\Theta^{(k)} = \Theta^{(k-1)} + s^{(k)}\eta(\Theta^{(k-1)}, \delta) \tag{16}$$

where

$$\eta(\Theta, \delta) = -[H_1(\Theta, \delta)]^{-1}\nabla J_1(\Theta) \tag{17}$$

is the optimizing direction vector, and

$$\nabla J_1(\Theta) = -\frac{1}{N} \sum_{t=1}^{N} \Psi(t, \Theta)\Lambda^{-1}\varepsilon(t, \Theta) \tag{18}$$

$$H_1(\Theta, \delta) = \frac{1}{N} \sum_{t=1}^{N} \Psi(t, \Theta)\Lambda^{-1}\Psi^{\mathrm{T}}(t, \Theta) + \delta I \tag{19}$$

are the gradient and the approximate hessian of $J_1(\Theta)$, respectively. δ is a non-negative small scalar and I is the identity matrix with appropriate dimension. The scalar $s^{(k)}$ is obtained by minimizing

$$J_1(\Theta^{(k-1)} + s\eta(\Theta^{(k-1)}, \delta)) \tag{20}$$

over $0 < s < 1$ using a linear search technique such as the golden section search. In practice, the direction vector $\eta(\Theta, \delta)$ is computed as follows. The square root decomposition method is first used to factorize the hessian as

$$H_1(\Theta, \delta) = U^{\mathrm{T}}U \tag{21}$$

where U is an upper triangular square matrix. $\eta(\Theta, \delta)$ is then solved from

$$U^{\mathrm{T}}(U\eta(\Theta, \delta)) = -\nabla J_1(\Theta) \tag{22}$$

by the forward and backward substitution algorithms (Bierman 1977).

The above Gauss–Newton algorithm is known to converge to at least a local minimum. Other loss functions can also be employed, and a different example to (15) is

$$J_2(\Theta) = \tfrac{1}{2}\log\det(C(\Theta)) \tag{23}$$

with

$$C(\Theta) = \frac{1}{N} \sum_{t=1}^{N} \varepsilon(t, \Theta)\varepsilon^{\mathrm{T}}(t, \Theta) \tag{24}$$

The gradient and the approximate hessian of $J_2(\Theta)$ are

$$\nabla J_2(\Theta) = -\frac{1}{N}\sum_{t=1}^{N} \Psi(t,\Theta)C^{-1}(\Theta)\varepsilon(t,\Theta) \tag{25}$$

$$H_2(\Theta,\delta) = \frac{1}{N}\sum_{t=1}^{N} \Psi(t,\Theta)C^{-1}(\Theta)\Psi^{\mathrm{T}}(t,\Theta) + \delta I \tag{26}$$

respectively. If Θ^* is a minimum of $J_1(\Theta)$, an optimal choice of Λ for the loss function $J_1(\Theta)$ is $C(\Theta^*)$. Choosing the criterion (23) is therefore equivalent to choosing the criterion (15) with Λ approaching an optimum. More detailed discussion of loss functions can be found in work by Goodwin and Payne (1977) and Ljung (1978).

4.2. *Recursive prediction error algorithm*

Many applications require recursive or adaptive updating of parameters. Ljung (1981) and Ljung and Söderström (1983) systematically studied recursive approximations of the prediction error method. Although they used the linear model in their studies, the principle is actually more general and can readily be applied to non-linear models as shown by Chen and Billings (1989 a). For the extended network model (12), the standard form of the recursive prediction error (RPE) algorithm based on the loss function (15) is

$$\begin{bmatrix} \hat{y}(t) \\ \Psi(t) \end{bmatrix} = \begin{bmatrix} \hat{f}(x(t);\hat{\theta}(t-1)) \\ \hat{g}(x(t);\hat{\Theta}(t-1)) \end{bmatrix} \tag{27}$$

$$\varepsilon(t) = y(t) - \hat{y}(t) \tag{28}$$

$$R(t) = R(t-1) + \gamma(t)[\Psi(t)\Lambda^{-1}\Psi^{\mathrm{T}}(t) + \delta I - R(t-1)] \tag{29}$$

$$\hat{\Theta}(t) = \hat{\Theta}(t-1) + \gamma(t)R^{-1}(t)\Psi(t)\Lambda^{-1}\varepsilon(t) \tag{30}$$

where $\hat{\Theta}(t)$ is the estimate of Θ at time t and $\gamma(t)$ is the gain at t. Notice that $\varepsilon(t)$, $\hat{y}(t)$ and $\Psi(t)$ depend upon all the old estimates $\hat{\Theta}(t-1)$ to $\hat{\Theta}(0)$. Thus (27) is time-varying.

$R(t)$ in (29) can clearly be viewed as a recursive form of (19). $\Psi(t)\Lambda^{-1}\varepsilon(t)$ corresponds to the gradient of $Q(\varepsilon(t))$ and is therefore a noisy or stochastic gradient. $R^{-1}(t)\Psi(t)\Lambda^{-1}\varepsilon(t)$ can thus be regarded as an approximation of the Gauss–Newton search direction (17). Equations (29) and (30) are mainly used for theoretical analysis. In practice, they are implemented in the equivalent form (with $\delta = 0$)

$$P(t) = \frac{1}{\lambda(t)}\{P(t-1) - P(t-1)\Psi(t) \\ \times [\lambda(t)\Lambda + \Psi^{\mathrm{T}}(t)P(t-1)\Psi(t)]^{-1}\Psi^{\mathrm{T}}(t)P(t-1)\} \tag{31}$$

$$\hat{\Theta}(t) = \hat{\Theta}(t-1) + P(t)\Psi(t)\Lambda^{-1}\varepsilon(t) \tag{32}$$

where

$$P(t) = \gamma(t)R^{-1}(t) \quad \text{and} \quad \lambda(t) = \gamma(t-1)(1-\gamma(t))/\gamma(t) \tag{33}$$

The simplest choice for Λ is I. A time-varying Λ:

$$\hat{\Lambda}(t) = \hat{\Lambda}(t-1) + \gamma(t)[\varepsilon(t)\varepsilon^{\mathrm{T}}(t) - \hat{\Lambda}(t-1)] \tag{34}$$

can however replace the constant Λ. The resulting RPE algorithm can be viewed as based on the criterion (23).

By applying a general method known as the differential equation method for the analysis of recursive parameter estimation algorithms developed by Ljung (1977), the convergence of the algorithm (27)–(30) can be proved. The underlying ideas of Ljung's method are as follows.

Assume that a projection is employed to keep $\hat{\Theta}(t)$ inside the stable region D_θ,

$$\lim_{t\to\infty} t\gamma(t) = \rho > 0 \tag{35}$$

$$R(t) \geqslant \delta I \quad \forall\, t \text{ and some } \delta > 0 \tag{36}$$

and some regularity conditions hold. Let $\Theta \in D_\theta$; then the time-invariant non-linear difference equation (12) is stable. The stability of the time-varying non-linear difference equation (27) will be guaranteed if $\hat{\Theta}(k)$ varies in a sufficiently small neighbourhood of Θ, and for sufficiently large t and some M, the influence of $\hat{\Theta}(k)$, $k = t - M - 1, \ldots, 0$, then becomes very small, i.e.

$$\begin{bmatrix} \hat{y}(t) \\ \Psi(t) \end{bmatrix} = \begin{bmatrix} \hat{y}(t, \hat{\Theta}(t-1), \ldots, \hat{\Theta}(0)) \\ \Psi(t, \hat{\Theta}(t-1), \ldots, \hat{\Theta}(0)) \end{bmatrix} \approx \begin{bmatrix} \hat{y}(t, \hat{\Theta}(t-1), \ldots, \hat{\Theta}(t-M)) \\ \Psi(t, \hat{\Theta}(t-1), \ldots, \hat{\Theta}(t-M)) \end{bmatrix} \tag{37}$$

Furthermore, assumption (35) implies $\gamma(t) \to 0$ as $t \to \infty$. For sufficiently large t, $\gamma(t)$ will be arbitrarily small, and it is seen that $\{\hat{\Theta}(t)\}$ will change more and more slowly, i.e.

$$\hat{\Theta}(t-1) \approx \ldots \approx \hat{\Theta}(t-M) \approx \tilde{\Theta} \tag{38}$$

As a consequence, the time-varying difference equation (27) behaves more and more like the time-invariant difference equation (12), and problems such as convergence with a probability of one, possible convergence points and asymptotic behaviour of the recursive algorithm can thus be studied in terms of an associated differential equation (for more details, see Ljung and Söderström 1983). The results show that the RPE algorithm has the same convergence properties as its corresponding off-line algorithm. One of these properties is that $\hat{\Theta}(t)$ converges with a probability of one to a local minimum of

$$\bar{J}_1(\Theta) = \lim_{N\to\infty} \frac{1}{2N} \sum_{t=1}^{N} E[Q(\varepsilon(t, \Theta))] \tag{39}$$

where $E[\,\cdot\,]$ is the expectation operator

For the neural network model (12), a projection to guarantee $\hat{\Theta}(t) \in D_\theta$ is not actually required because D_θ is the whole space $\mathbb{R}^{n_\theta}$. The above convergence results are obtained under assumption (35), which implies $\gamma(t) \to 0$ as $t \to \infty$ (or $\lambda(t) \to 1$ as $t \to \infty$). In order to track time-varying parameters, $\gamma(t)$ should not tend to zero. It is reasonable to believe that analysis under condition (35) will have relevance for the case where $\gamma(t)$ tends to some small non-zero value. As in any non-linear optimization problem, the initial conditions have an important influence on convergence and the speed of convergence. The performance surface (39) for a general network model is very complex and is known in general to contain many local minima. A study of this performance surface and the influence of $\hat{\Theta}(0)$ on the algorithm (27)–(30) is beyond the scope of this paper.

Strictly speaking, algorithm (30) or (32) is only a crude approximation of the off-line Gauss–Newton algorithm because $-\Psi(t)\Lambda^{-1}\varepsilon(t)$ is hardly a good approximation of the gradient (18). A modified RPE algorithm is proposed here by introducing a smoothed stochastic gradient

$$\Delta(t) = \Delta(t-1) + \gamma(t)[\Psi(t)\Lambda^{-1}\varepsilon(t) - \Delta(t-1)] \tag{40}$$

so that, in parameter updating,

$$\hat{\Theta}(t) = \hat{\Theta}(t-1) + \gamma(t)R^{-1}(t)\Delta(t) \quad \text{or} \quad \hat{\Theta}(t) = \hat{\Theta}(t-1) + P(t)\Delta(t) \tag{41}$$

The new algorithm is thus a truly recursive Gauss–Newton algorithm. Using a smoothed stochastic gradient usually improves the performance of the recursive algorithm at the cost of more computation in each recursion. Smoothed stochastic gradient algorithms can be directly obtained from this new algorithm by making some simplifications. One example is

$$\hat{\Theta}(t) = \hat{\Theta}(t-1) + \bar{\gamma}(t)\Delta(t) \tag{42}$$

The so-called back propagation algorithm (Rumelhart *et al.* 1986) is a simple version of this smoothed stochastic gradient algorithm with $\bar{\gamma}(t)$ in (42) and $\gamma(t)$ in (40) fixed to some constant values.

In adaptive identification, $0 < \lambda(t) < 1$. If the covariance matrix $P(t)$ is implemented in its basic form as given by (31), a phenomenon known as 'covariance wind-up' may occur. That is, $P(t)$ may become fairly large. When this occurs and in addition the gradient is dominated by noise, changes in $\hat{\Theta}(t)$ are unlikely in the direction of improving the model output and this can cause a problem known as 'parameter drift'. Two factors are likely to introduce covariance wind-up when applying the RPE algorithm to neural network models. Because of the complexity of the non-linear structure, it is possible that two different value of Θ can result in the same input–output relationship from (12). When the parametrization is not unique, covariance wind-up can occur (Janecki 1988). If the signal excitation is poor, covariance wind-up may happen (Sripada and Fisher 1987). For the recursive least squares algorithm, similar difficulties can arise and many numerical modifications have been developed to overcome these problems in the single-output ($m = 1$) case. A technique often used is the constant trace adjustment in which $P(t)$ is adjusted in such a way that its trace remains constant. A more sophisticated technique called exponential resetting and forgetting (Salgado *et al.* 1988) can also be employed.

5. Model validation

If modelling is adequate, $\varepsilon(t, \Theta)$ will be unpredictable from (uncorrelated with) all linear and non-linear combinations of past inputs and outputs. Model validity tests for other non-linear models (Billings and Voon 1986, Billings and Chen 1989, Billings *et al.* 1989, Leontaritis and Billings 1987) were developed based on this principle and can therefore be applied to the current neural network model. For simplicity, only single-input ($r = 1$) single-output model validity tests are briefly summarized.

If the identified model is adequate, the prediction errors should satisfy the following conditions (Billings and Voon 1986, Billings and Chen 1989)

$$\left.\begin{aligned} \Phi_{\varepsilon\varepsilon}(k) &= \text{an impulse function} \\ \Phi_{u\varepsilon}(k) &= 0 \quad \text{for all } k \\ \Phi_{\varepsilon(\varepsilon u)}(k) &= 0 \quad k \geqslant 0 \\ \Phi_{u^{2'}\varepsilon}(k) &= 0 \quad \text{for all } k \\ \Phi_{u^{2'}\varepsilon^2}(k) &= 0 \quad \text{for all } k \end{aligned}\right\} \tag{43}$$

where $\Phi_{xz}(k)$ indicates the cross-correlation function between $x(t)$ and $z(t)$, $\varepsilon u(t) = \varepsilon(t+1)u(t+1)$, $u^{2'}(t) = u^2(t) - u^2(t)$ and $u^2(t)$ represents the time average or mean value of $u^2(t)$. Therefore if these correlation functions are within the (95%) confidence intervals $\pm 1{\cdot}96/\sqrt{N}$, the model is regarded as adequate.

Alternatively a statistical test known as the chi-squared test (Bohlin 1978, Leontaritis and Billings 1987) can be employed to validate the identified model. Let $\Omega(t)$ be an s-dimensional vector-valued function of the past inputs, outputs and prediction errors, and

$$\Gamma^{\mathrm{T}}\Gamma = \frac{1}{N}\sum_{t=1}^{N} \Omega(t)\Omega^{\mathrm{T}}(t) \tag{44}$$

Then the chi-squared statistic is computed using the formula

$$\zeta = N\mu^{\mathrm{T}}(\Gamma^{\mathrm{T}}\Gamma)^{-1}\mu \tag{45}$$

where

$$\mu = \frac{1}{N}\sum_{t=1}^{N} \Omega(t)\varepsilon(t, \hat{\Theta})/\sigma_{\varepsilon} \tag{46}$$

$\hat{\Theta}$ is the estimate of Θ and σ_{ε}^2 is the variance of the residuals. Under the null hypothesis that the data are generated by the model, the statistic ζ is asymptotically chi-squared distributed with s degrees of freedom. A convenient choice for $\Omega(t)$ is

$$\Omega(t) = [\omega(t)\omega(t-1) \quad \dots \quad \omega(t-s+1)]^{\mathrm{T}} \tag{47}$$

where $\omega(t)$ is some chosen (non-linear) function of the past inputs, outputs and prediction errors. Thus if the values of ζ for several different choices of $\omega(t)$ are within the acceptance region (95%), that is

$$\zeta < \chi_s^2(\alpha) \tag{48}$$

the model can be regarded as adequate, where $\chi_s^2(\alpha)$ is the critical value of the chi-squared distribution with s degrees of freedom for the given significance level α ($0{\cdot}05$).

To sum up the discussion so far, the identification of a structure-unknown system described by (3) using a single hidden layer neural network involves the following procedure:

(*a*) choose values of n_y, n_u and n_h;

(*b*) estimate Θ;

(*c*) validate the estimated model. If the model is adequate, the procedure is terminated; otherwise go to step (*a*).

6. Simulation study

The parameter estimation algorithm used in this simulation study was the off-line prediction error algorithm and only single-input single-output examples are given.

Example 1

This is a simulated system. 500 points of data were generated by

$$\begin{aligned} y(t) = {} & (0{\cdot}8 - 0{\cdot}5\exp(-y^2(t-1)))y(t-1) - (0{\cdot}3 + 0{\cdot}9\exp(-y^2(t-1)))y(t-2) \\ & + u(t-1) + 0{\cdot}2u(t-2) + 0{\cdot}1u(t-1)u(t-2) + e(t) \end{aligned}$$

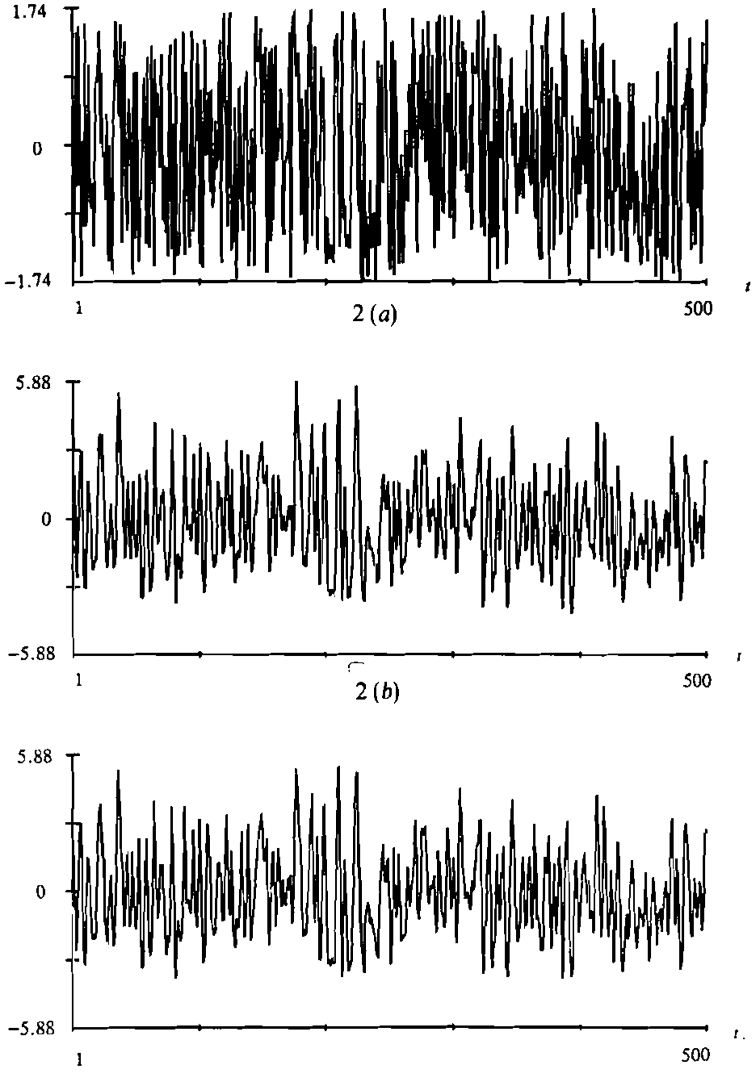
where the system noise $e(t)$ was a gaussian white sequence with mean zero and variance 0·04 and the system input $u(t)$ was an independent sequence of uniform distribution with mean zero and variance 1·0.

The input order of the network model was chosen as $n = n_y + n_u = 2 + 2$. When the number of hidden nodes was increased to $n_h = 5$ ($n_\theta = 30$) the model validity tests were satisfied. Figure 2 shows the system and model response where the model deterministic output $\hat{y}_d(t, \hat{\Theta})$ is defined by

$$\hat{y}_d(t, \hat{\Theta}) = \hat{f}(\hat{y}_d(t-1, \hat{\Theta}), \ldots, \hat{y}_d(t-n_y, \hat{\Theta}), u(t-1), \ldots, u(t-n_u); \hat{\Theta}) \tag{49}$$

and the deterministic error $\varepsilon_d(t, \hat{\Theta})$ is given as

$$\varepsilon_d(t, \hat{\Theta}) = y(t) - \hat{y}_d(t, \hat{\Theta}) \tag{50}$$



2 (*a*)

2 (*b*)

2 (*c*)

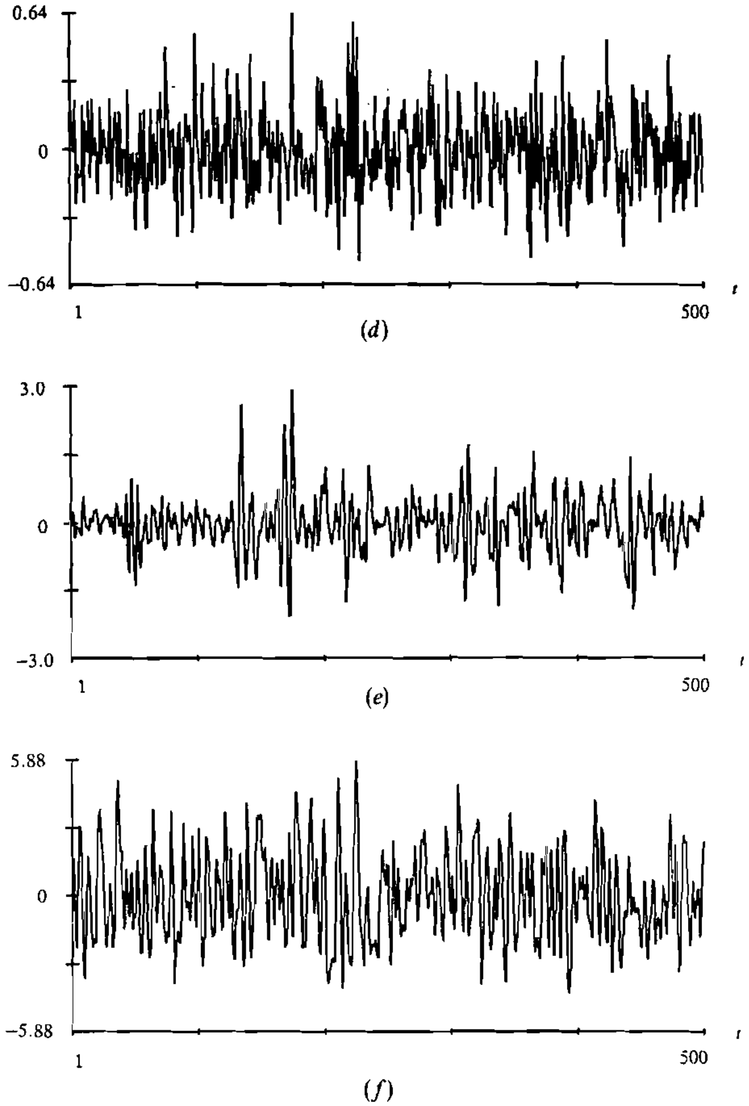


Figure 2. System and model response (Example 1): (*a*) $u(t)$; (*b*) $y(t)$; (*c*) $\hat{y}(t, \hat{\Theta})$; (*d*) $\varepsilon(t, \hat{\Theta})$; (*e*) $\varepsilon_d(t, \hat{\Theta})$; (*f*) $\hat{y}_d(t, \hat{\Theta})$.

Figures 3 and 4 display the correlation tests and some chi-squared tests for the estimated model.

It can easily be verified that the unforced response (that is $e(t) = 0$ and $u(t) = 0$) of this simulated system is a stable limit cycle as illustrated in Fig. 5. The unforced response from the estimated model with the same initial condition is plotted in Fig. 6, where it is seen that, although the shape is different from that in Fig. 5, the estimated model correctly predicts the existence of a limit cycle. The data shown in Fig. 5 were used to identify a network model with $n = n_y = 2$ and $n_h = 10$ ($n_\theta = 40$). The resulting model produces the limit cycle shown in Fig. 7, which is much closer to that produced by the unforced system.

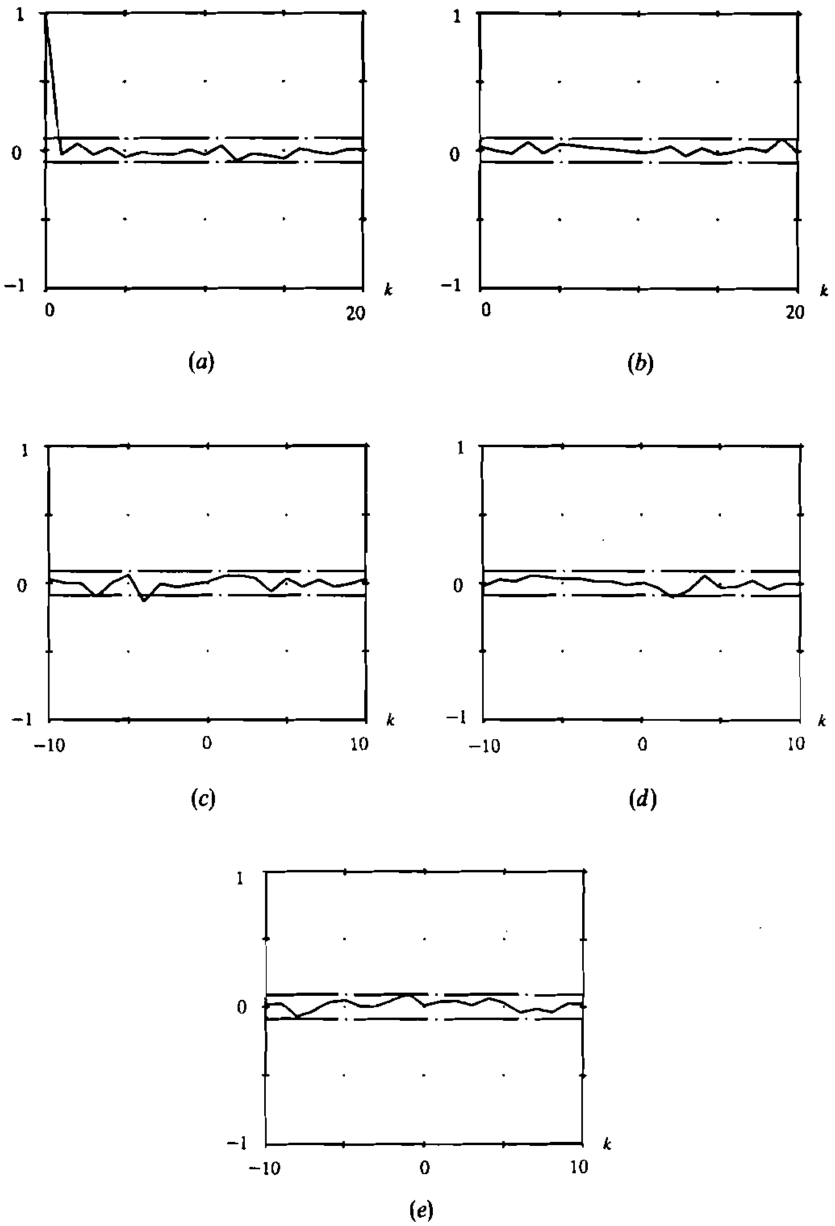


Figure 3. Correlation tests (Example 1): (*a*) $\Phi_{\varepsilon\varepsilon}(k)$; (*b*) $\Phi_{\varepsilon(\varepsilon u)}(k)$; (*c*) $\Phi_{u\varepsilon}(k)$; (*d*) $\Phi_{u^{2\prime}\varepsilon}(k)$; (*e*) $\Phi_{u^{2\prime}\varepsilon^{2}}(k)$. Dashed line: 95% confidence interval.

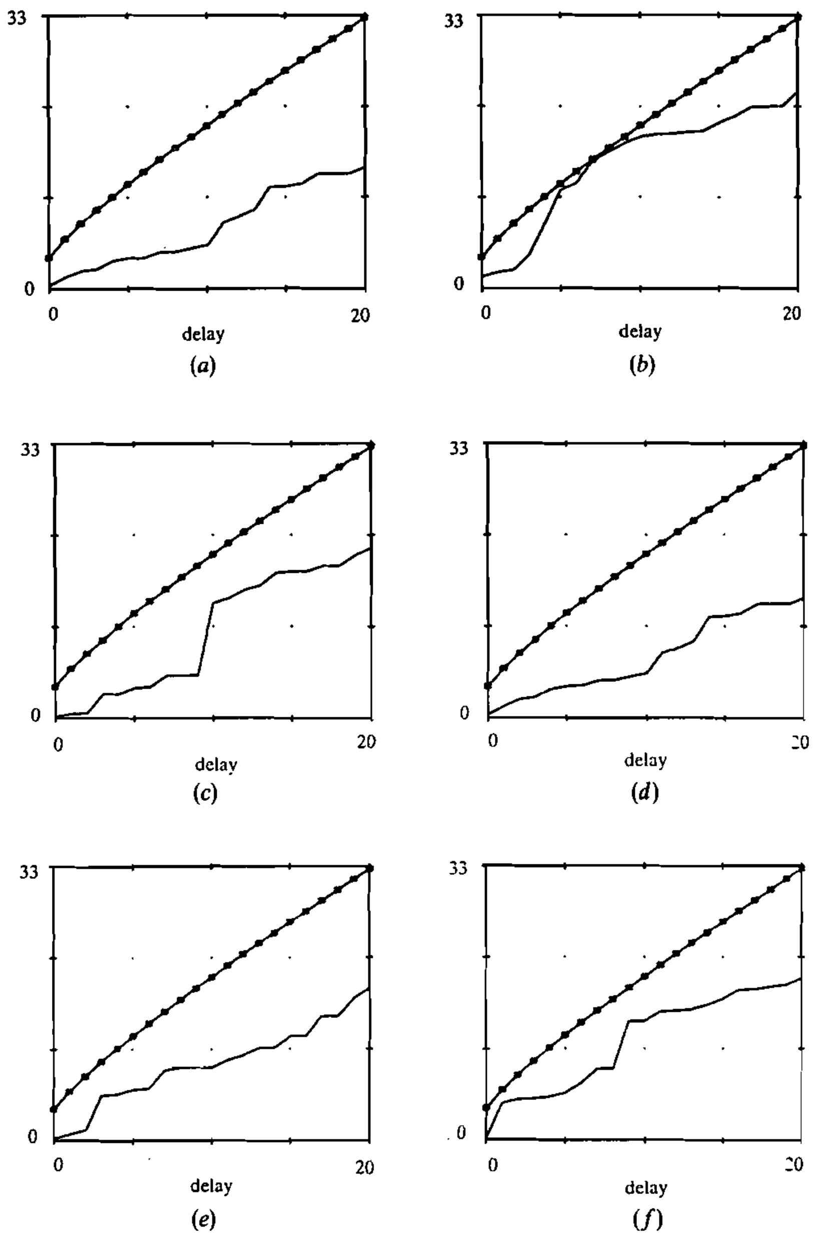


Figure 4. Chi-squared tests (Example 1): (*a*) $\omega(t)=\varepsilon(t-1,\hat{\Theta})$; (*b*) $\omega(t)=y(t-1)$; (*c*) $\omega(t)=\exp(u(t-1))$; (*d*) $\omega(t)=\tan h(\varepsilon(t-1,\hat{\Theta}))$; (*e*) $\omega(t)=y^2(t-1)\varepsilon^2(t-2,\hat{\Theta})$; (*f*) $\omega(t)=\exp(-u^2(t-2))\exp(-y^2(t-2))$. Dashed line: 95% confidence limit.

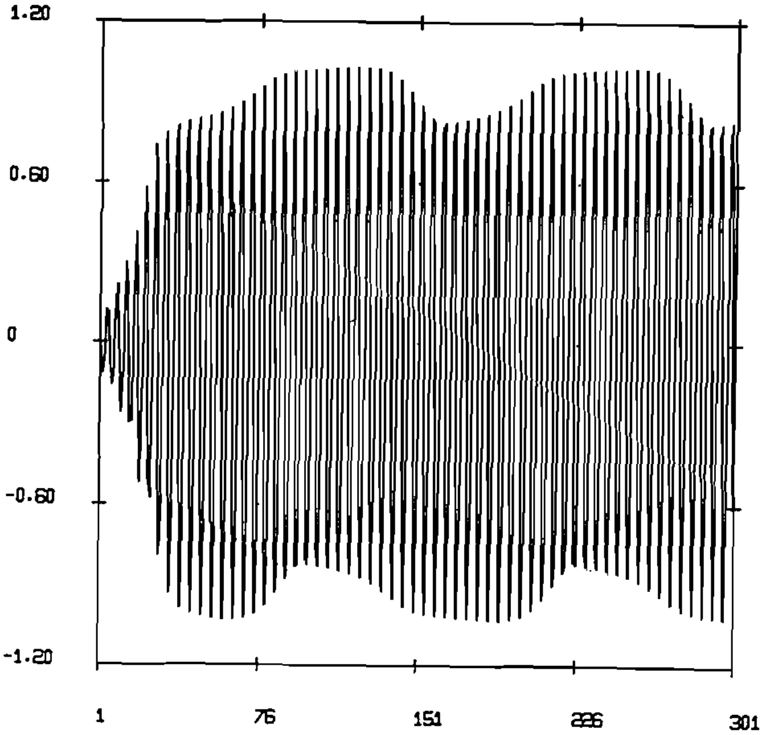


Figure 5. System unforced response (Example 1): initial condition: $y(-1) = 0{\cdot}01$, $y(0) = 0{\cdot}1$.

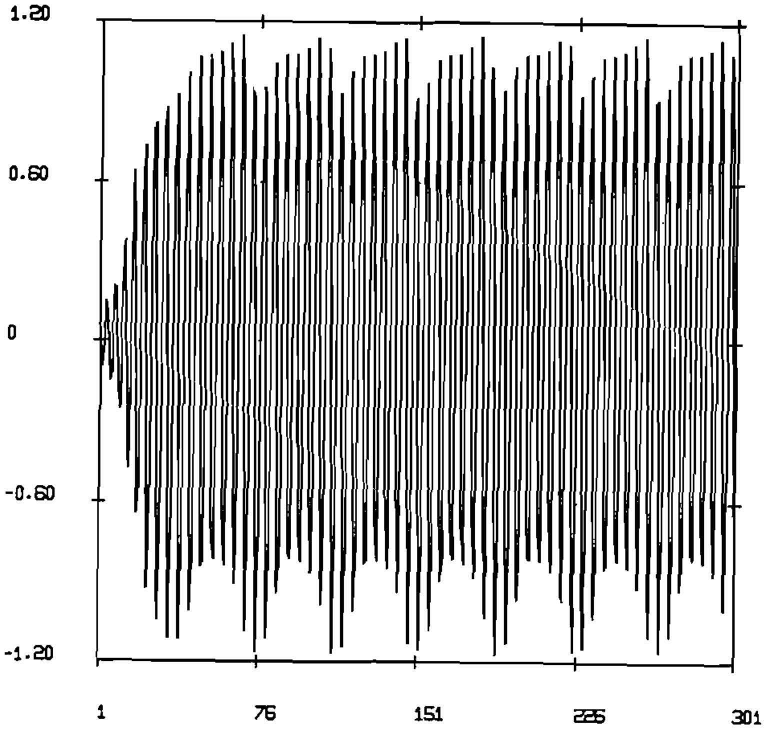


Figure 6. Control model unforced response (Example 1): initial condition: $y(-1) = 0{\cdot}01$, $y(0) = 0{\cdot}1$.

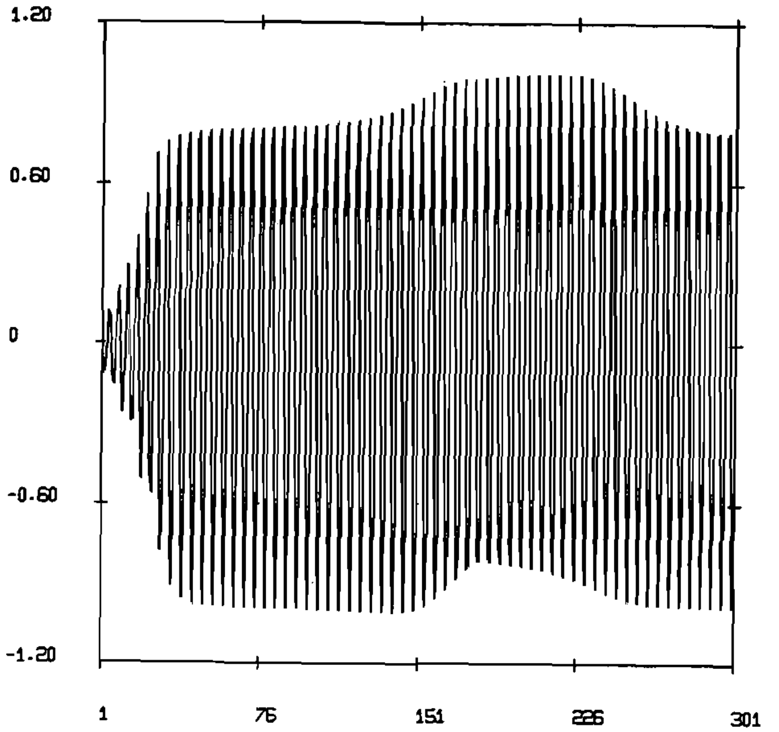


Figure 7. Time series model unforced response (Example 1): initial condition: $y(-1) = 0{\cdot}01$, $y(0) = 0{\cdot}1$.

Example 2

This is the time series of annual sunspot numbers. Observations for the years 1700 to 1979 can be found in a paper by Tong (1983, Appendix A. 1). The first 256 observations are plotted in Fig. 8 (*a*).

It has long been noticed that the record of sunspot numbers reveals an intriguing cyclical phenomenon of an approximate 11-year period. Chen and Billings (1989 c) fitted a subset polynomial model with $n_y = 9$ and polynomial-degree three to the first 221 observations. The unforced response of this subset polynomial model is a sustained oscillation with an approximate 11-year period as shown in Fig. 8 (*c*). In the current study a neural network model with $n = n_y = 9$ and $n_h = 5$ ($n_\theta = 55$) was fitted to the first 221 observations. The unforced response of this neural network model is illustrated in Fig. 8 (*b*) where it is seen that this time series model also produces a sustained oscillation with an approximate 11-year period.

Example 3

The data were generated from a heat exchanger and contains 996 points. A detailed description of this process and the experimental design can be found in work by Billings and Fadzil (1985). The first 500 points of the data, depicted in Fig. 9, were used as the identification set and the rest of the data as the test set.

A neural network model with $n_y = n_u = 5$ and $n_h = 3$ ($n_\theta = 36$) was fitted to the identification data set. Figures 10 and 11 show the correlation tests using the identi-

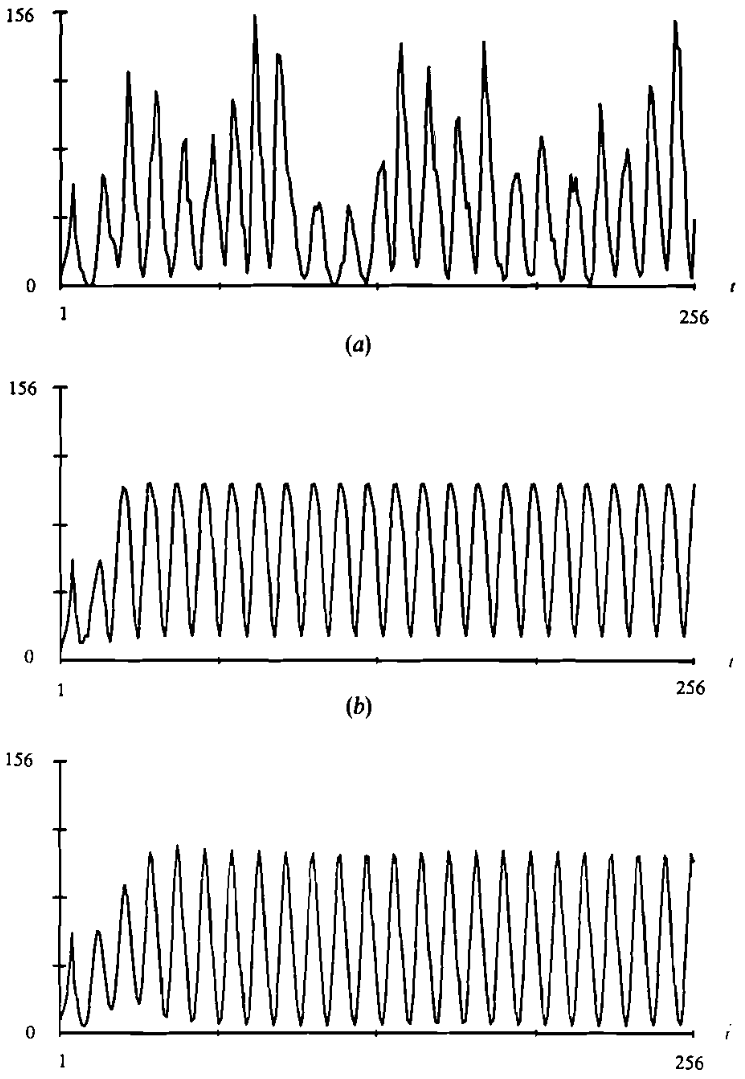


Figure 8: Observations and model unforced response (Example 2): (*a*) observations; (*b*) neural network model; (*c*) subset polynomial model; first nine observations used as initial condition in unforced response.

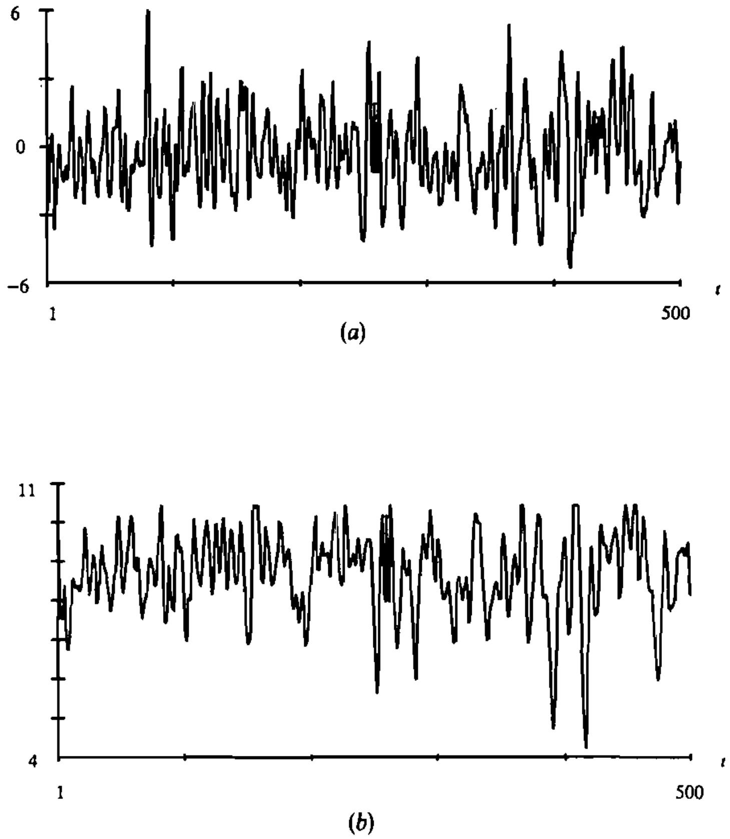


Figure 9. Identification data set (Example 3): (*a*) $u(t)$; (*b*) $y(t)$.

fication and test sets, respectively. The test set and model response for this set are given in Fig. 12. Further increasing the size of the network only slightly improved the quality of fit.

Previous identification results (Billings and Chen 1989) indicate that this non-linear process can be described better by using a model with the form of (1). The results obtained here are satisfactory considering that no noise model was fitted as part of the model estimation.

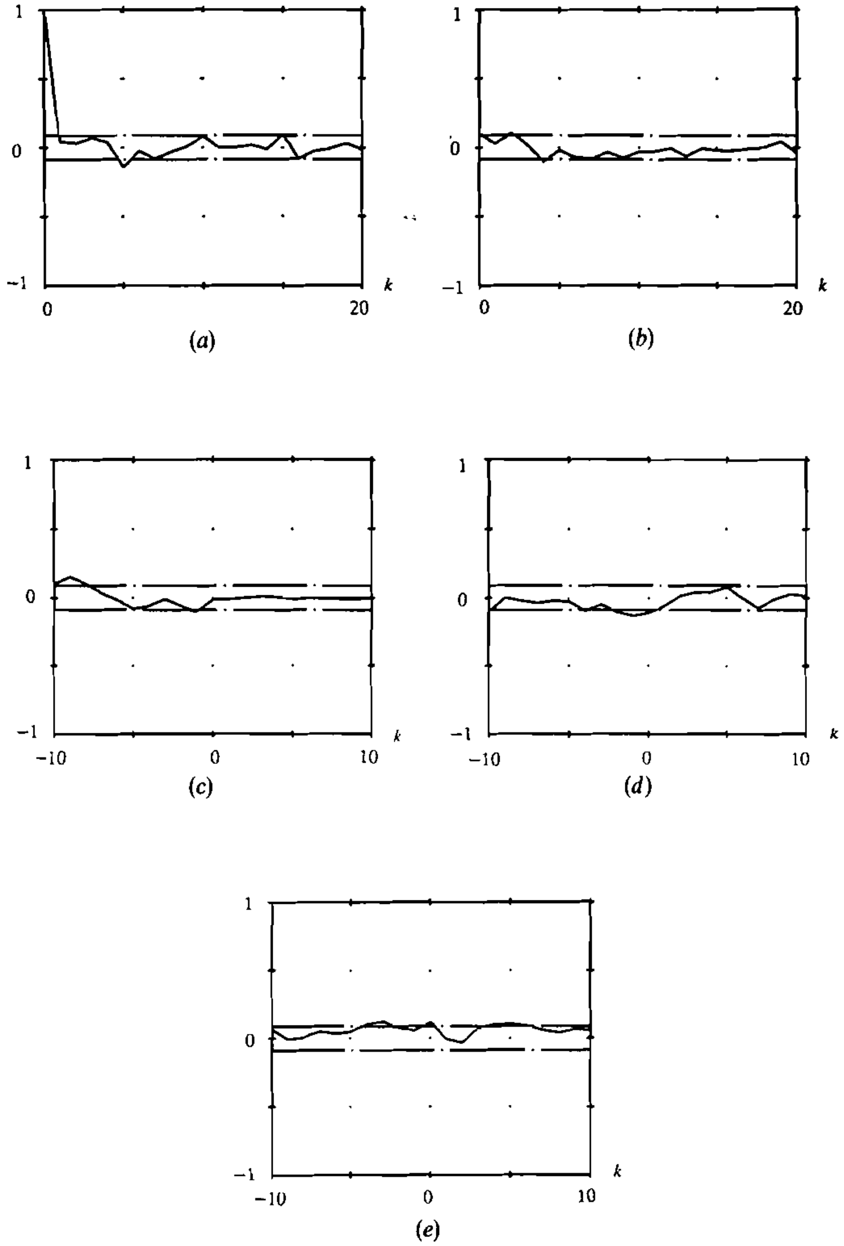


Figure 10. Correlation tests using identification set (Example 3): (*a*) $\Phi_{\varepsilon\varepsilon}(k)$; (*b*) $\Phi_{\varepsilon(\varepsilon u)}(k)$; (*c*) $\Phi_{u\varepsilon}(k)$; (*d*) $\Phi_{u^{2'}\varepsilon}(k)$; (*e*) $\Phi_{u^{2'}\varepsilon^{2}}(k)$. Dashed line: 95% confidence interval.

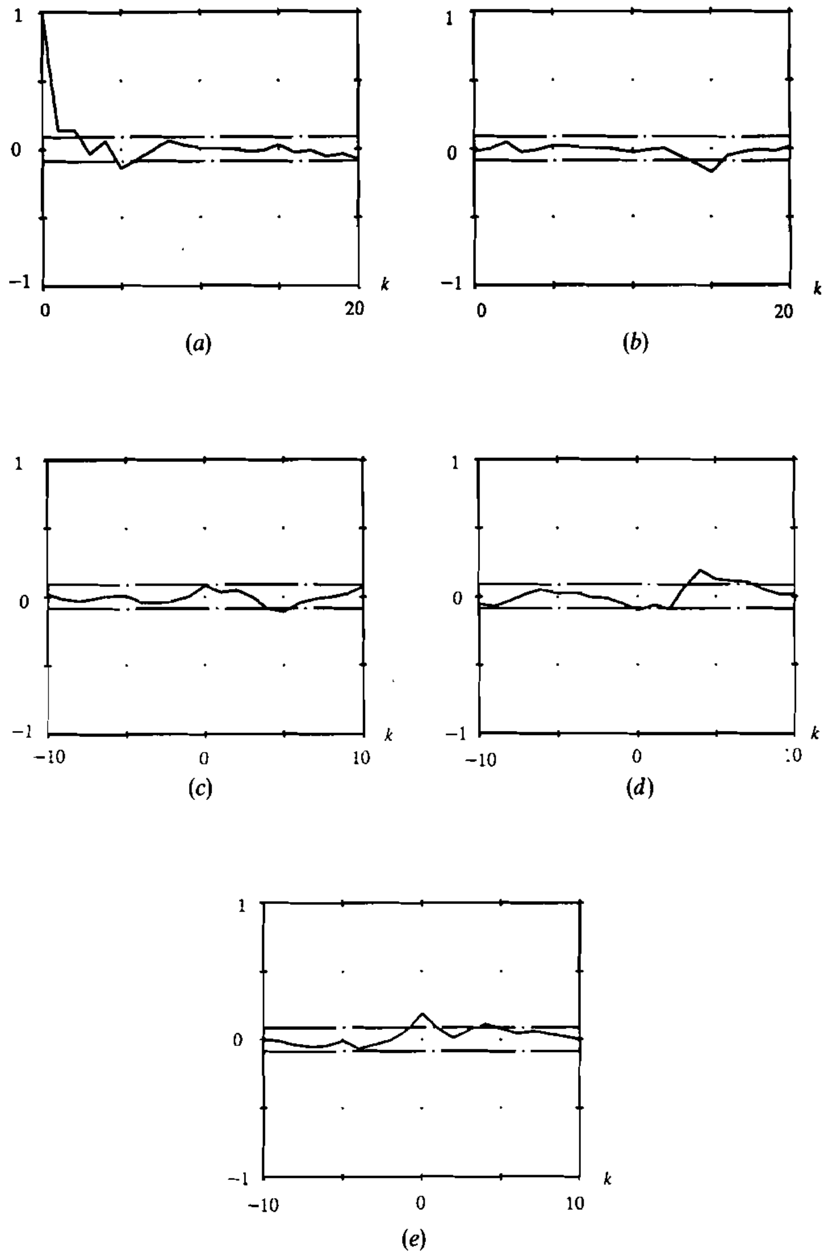
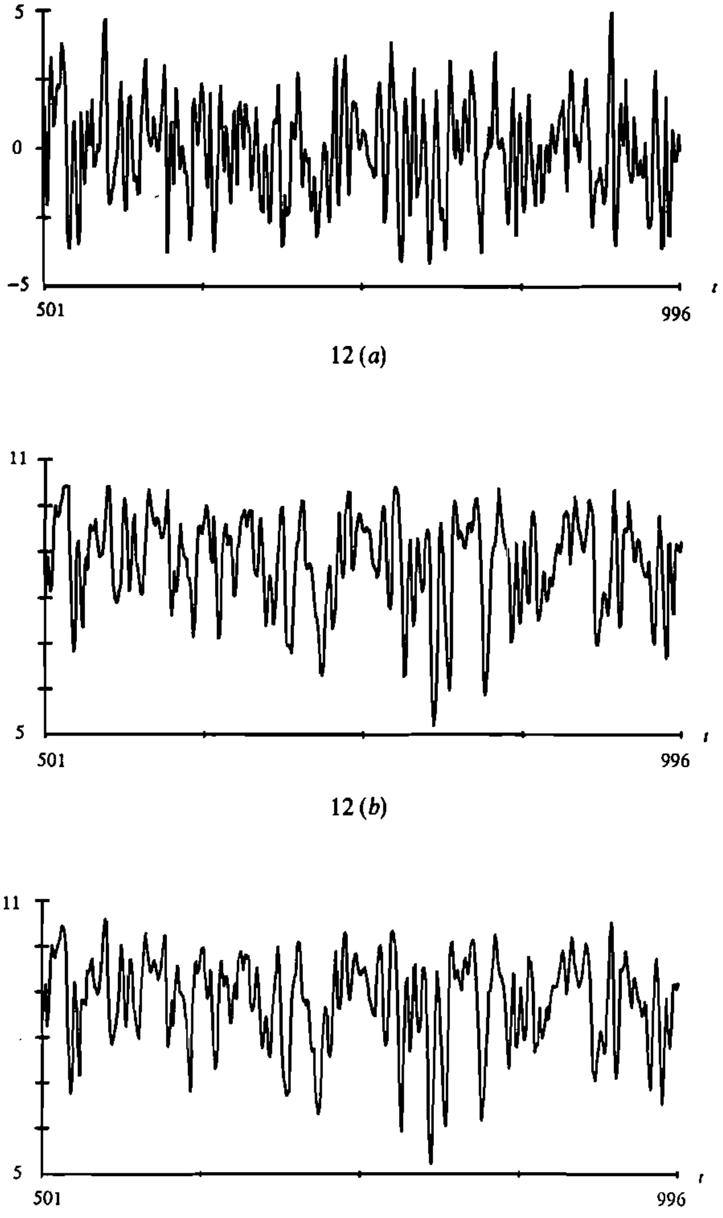


Figure 11. Correlation tests using test set (Example 3): (*a*) $\Phi_{\varepsilon\varepsilon}(k)$; (*b*) $\Phi_{\varepsilon(\varepsilon u)}(k)$; (*c*) $\Phi_{u\varepsilon}(k)$; (*d*) $\Phi_{u^{2'}\varepsilon}(k)$; (*e*) $\Phi_{u^{2'}\varepsilon^{2}}(k)$. Dashed line: 95% confidence interval.



12 (*a*)

12 (*b*)

12 (*c*)

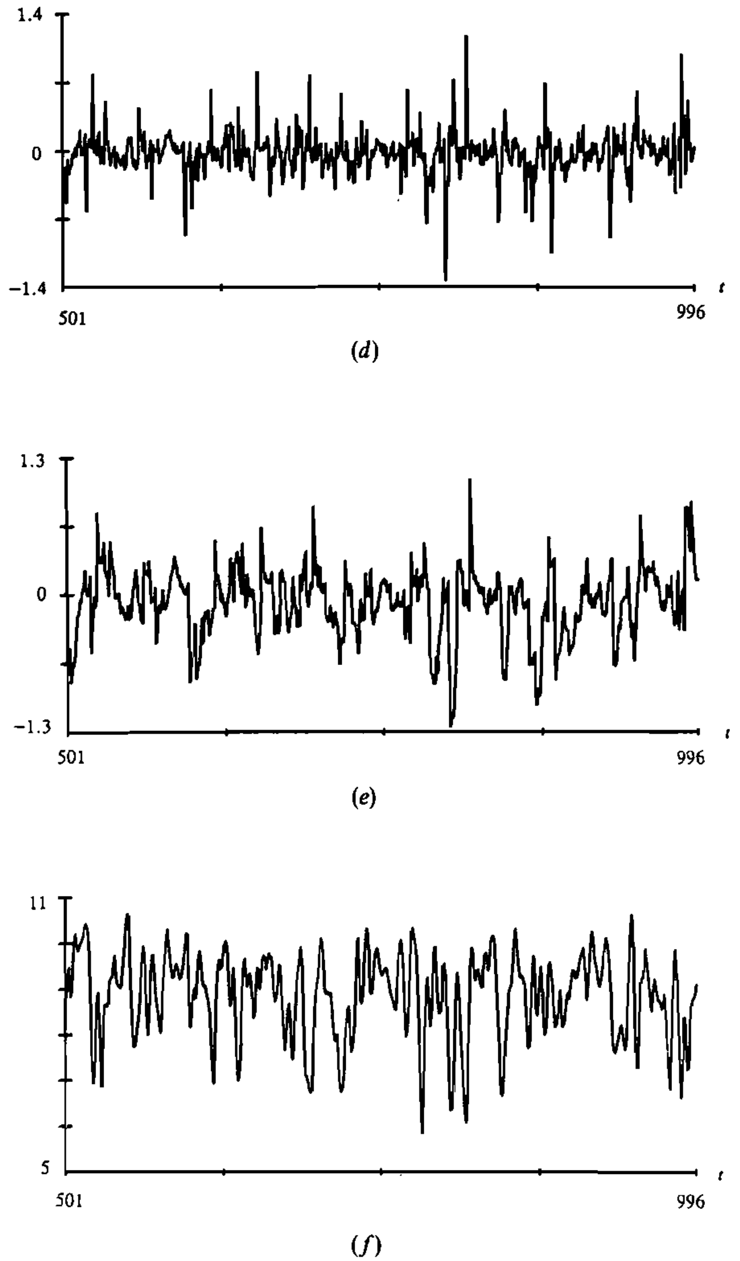


Figure 12. Test set and model response (Example 3): (*a*) $u(t)$; (*b*) $y(t)$; (*c*) $\hat{y}(t, \hat{\Theta})$; (*d*) $\varepsilon(t, \hat{\Theta})$; (*e*) $\varepsilon_d(t, \hat{\Theta})$; (*f*) $\hat{y}_d(t, \hat{\Theta})$.

7. Discussion for further research

As mentioned in § 2, (3) is only a special case of the general system (1). The approach developed in the present study can easily be extended to the general system (1) by augmenting $x(t)$ to

$$x(t) = [y^{\mathrm{T}}(t-1) \quad \dots \quad y^{\mathrm{T}}(t-n_y)u^{\mathrm{T}}(t-1) \quad \dots \quad u^{\mathrm{T}}(t-n_u)\varepsilon^{\mathrm{T}}(t-1,\Theta) \quad \dots \\ \varepsilon^{\mathrm{T}}(t-n_e,\Theta)]^{\mathrm{T}} \tag{51}$$

The extended network model becomes

$$\begin{bmatrix} \hat{y}(t,\Theta) \\ \Psi(t,\Theta) \end{bmatrix} = \begin{bmatrix} \hat{f}(x(t);\Theta) \\ \hat{g}(\Psi(t-1,\Theta),\dots,\Psi(t-n_e,\Theta),x(t);\Theta) \end{bmatrix} \tag{52}$$

The application of the parameter estimation algorithms of § 4 to this model is straightforward. The analysis of a non-linear model involving $\varepsilon(t-i,\Theta)$ within its arguments is however considerably more difficult. In particular, D_θ generally is no longer the whole n_θ-dimensional space and is dependent on system input–output statistics. In some extreme cases, D_θ may not even exist. In other words the issue of invertibility (Chen and Billings 1989 c), i.e. whether it is possible to compute $\varepsilon(t,\Theta)$ using the model and given system inputs and outputs, becomes critical. Unlike the polynomial model, which may be explosive, the network model with the activation function (9) can be non-explosive. The neural network approach may therefore be more suitable for modelling non-linear time series whose underlying processes are stable and non-explosive. There is scope for further investigation of this aspect.

A comprehensive study is required to compare the neural network model with other non-linear models. For the polynomial model, efficient procedures for selecting subset models have been developed (Chen *et al.* 1989, Leontaritis and Billings 1987). A parsimonious model has advantages in controller design, prediction and other practical applications. Selection of subset neural network models is worth invetigating. One possible approach is to develop a criterion like Akaike's information criterion (Leontaritis and Billings 1987) for removing insignificant connection weights.

It is a common belief that neural networks with several hidden layers can approximate a function more efficiently (with less nodes) for a given accuracy requirement than networks with a single hidden layer. More theoretical research is required to derive some quantitative results. There are other advantages of using highly layered networks, such as increasing integrity. The identification procedure for the network model with several hidden layers, however, will not be as simple as that given at the end of § 5 because more than one hidden layer will need to be specified.

For non-linear systems which exhibit a significant constant level independent of system input and noise, a threshold can be introduced to each output node. The activation function is not restricted to (9). The study of different activation functions to compare their performance is of practical interest. If a polynomial model is used to model the system (3), the loss function (39) has a single global minimum for a fixed dimension n_θ. It is well known that (39) contains many local minima if the neural network model is employed. Further investigation is required to analyse the effect of this on the outcome of the identification.

8 Conclusions

An identification procedure has been developed for disrete-time non-linear systems based on a neural network approach. Both batch and recursive prediction error estimation algorithms have been derived for a neural network model with a single hidden layer and model validation methods have been discussed. Application to some simulated and real systems has been demonstrated. The results obtained suggest that modelling non-linear systems by neural networks is an effective approach and further research in this field is worth pursuing.

Acknowledgments

This work is supported by the U.K. Science and Engineering Research Council. The authors are also grateful for information supplied by Dr G. J. Gibson.

References

Bierman, G. J., 1977, *Factorization Methods for Discrete Sequential Estimation* (New York: Academic Press).

Billings, S. A., and Chen, S., 1989, Identification of non-linear rational systems using a prediction-error estimation algorithm. *International Journal of Systems Science*, **20**, 467–494.

Billings, S. A., Chen, S., and Korenberg, M. J., 1989, Identification of MIMO non-linear systems using a forward-regression orthogonal estimator. *International Journal of Control*, **49**, 2157–2189.

Billings, S. A., and Fadzil, M. B., 1985, The practical identification of systems with nonlinearities. *Proc. of the 7th IFAC Symposium on Identification and System Parameter Estimation*, York, U.K., pp. 155—160.

Billings, S. A., and Voon, W. S. F., 1986, Correlation based model validity tests for nonlinear models. *International Journal of Control*, **44**, 235–244.

Bohlin, T., 1978, Maximum-power validation of models without higher-order fitting. *Automatica*, **4**, 137–146.

Chen, S., and Billings, S. A., 1989 a, Recursive prediction error parameter estimator for nonlinear models. *International Journal of Control*, **49**, 569–594; 1989 b, Representation of non-linear systems: the NARMAX model. *Ibid.*, **49**, 1013–1032; 1989 c, Modelling and analysis of non-linear time series. *Ibid.*, **50**, 2151–2171.

Chen, S., Billings, S. A., and Luo, W., 1989, Orthogonal least squares methods and their application to non-linear system identification. *International Journal of Control*, **50**, 1873–1896.

Cybenko, G., 1989, Approximations by superpositions of a sigmoidal function. *Mathematics of Control, Signals and Systems*, **2**, 303–314.

Funahashi, K., 1989, On the approximate realization of continuous mappings by neural networks. *Neural Networks*, **2**, 183–192.

Goodwin, G. C., and Payne, R. L., 1977, *Dynamic System Identification: Experiment Design and Data Analysis* (New York: Academic Press).

I.E.E.E., 1988, *I.E.E.E. Transactions on Acoustics, Speech and Signal Processing*, **36** (7).

Janecki, D., 1988, New recursive parameter estimation algorithms with varying but bounded gain matrix. *International Journal of Control*, **47**, 75–84.

Leontaritis, I. J., and Billings, S. A., 1985, Input-output parametric models for non-linear systems. Part 1: Deterministic non-linear systems; Part 2: Stochastic non-linear systems. *International Journal of Control*, **41**, 303–344; 1987, Model selection and validation methods for non-linear systems. *Ibid.*, **45**, 311–341; 1988, Prediction error estimator for non-linear stochastic systems. *International Journal of Systems Science*, **19**, 519–536.

Ljung, L., 1977, Analysis of recursive stochastic algorithms. *I.E.E.E. Transactions on Automatic Control*, **22**, 551–575; 1978, Convergence analysis of parametric identification methods. *Ibid.*, **23**, 770–783; 1981, Analysis of a general recursive prediction error identification algorithm. *Automatica*, **17**, 89–99.

Ljung, L., and Söderström, T., 1983, *Theory and Practice of Recursive Identification* (Cambridge, MA: MIT Press).

RUMELHART, D. E., HINTON, G. E., and WILLIAMS, R. J., 1986, Learning internal representations by error propagation. In *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, edited by Rumelhart, D. E., and McClelland, J. L., pp. 318–362 (Cambridge, MA: MIT Press).

SALGADO, M. E., GOODWIN, G. C., and MIDDLETON, R. H., 1988, Modified least squares algorithm incorporating exponential resetting and forgetting. *International Journal of Control*, **47**, 477–491.

SRIPADA, N. R., and FISHER, D. G., 1987, Improved least squares identification. *International Journal of Control*, **46**, 1889–1913.

TONG, H., 1983, *Threshold Models in Non-linear Time Series Analysis*. Lecture Notes in Statistics (New York: Springer-Verlag).