International Journal of Systems Science, 1996, volume 27, number 2, pages 221–226

Training of Elman networks and dynamic system modelling

D. T. PHAM† and X. LIU†

A dynamic backpropagation (DBP) algorithm is presented to train the Elman network to model dynamic systems. The relationship between the Elman network trained by the DBP algorithm and the modified Elman network previously proposed by the authors is clarified. The paper shows that the modified Elman network is an approximate realization of the Elman network trained by the DBP algorithm. It is the self-feedback links of the context units of the modified Elman network which provide a dynamic trace of the gradients in the parameter space and enable the network to model dynamic systems of orders higher than one. The paper first gives the results of modelling a second-order linear plant and a third-order linear plant. Neither plant could be modelled using Elman networks trained by the standard backpropagation algorithm, but both were successfully modelled by DBP-trained Elman networks as they had been in previous studies by modified Elman networks. Finally, the paper reports on the application of the DBP-trained Elman net to model a benchmark nonlinear process.

1. Introduction

The Elman network (Elman 1990) is a type of recurrent network. There has been much research interest in this network (Pham and Liu 1992, Hertz *et al.* 1991, Kuan 1989, Kuan *et al.* 1989) and it has been built into the MATLAB software (The Math Works Inc. 1989) and applied to dynamic system identification (Pham and Liu 1992) and financial prediction (Kamijo and Tanigawa 1990). A modified Elman network has been proposed by Pham and Liu (1992) because it was found that the basic Elman network trained by the standard backpropagation (BP) algorithm (Rumelhart and McClelland 1986) was able to model only first-order dynamic systems. This paper describes the dynamic BP (DBP) algorithm which is proper for training the basic Elman network and shows that the modified Elman network is an approximation of the Elman network trained by DBP. This further clarifies why the modified Elman network can model higher-order dynamic systems.

2. The Elman network

Elman (1990) introduced a simple recurrent neural network, as shown in Fig. 1 (the network is assumed to have only one input unit and one output unit in this case). This network is very similar to the network proposed by Robinson and Fallside (1987). The latter possesses a structure whose connections are defined in the same way as that employed by Werbos (1990). By forcing some weights to zero, the networks given by Elman (1990) and Robinson and Fallside (1987) have the same structure. The work of this paper follows the feedforward layered structure defined by Elman (1990).

From Fig. 1 it can be seen that in a basic Elman network, in addition to the input unit, the hidden units

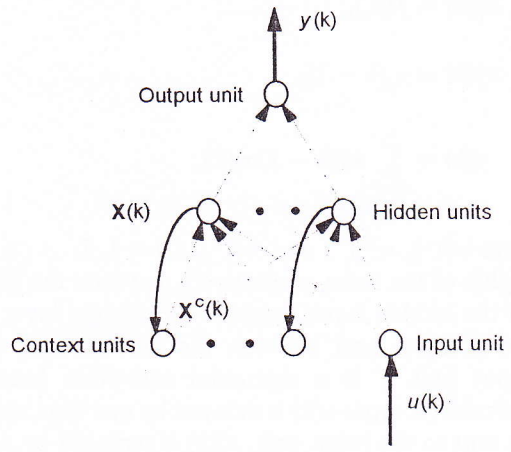

Figure 1. Basic Elman network.

Received 16 August 1995. Accepted 22 August 1995.

† Intelligent Systems Research Laboratory, School of Engineering, University of Wales, Cardiff, U.K.

0020-7721/96 $12.00

and the output unit, there are also context units. The input and output units interact with the outside environment, whereas the hidden and context units do not. The input unit is only a buffer unit which passes the signals without changing them. The output unit is a linear unit which sums the signals fed to it. The hidden units can have linear or nonlinear activation functions. The context units are used only to memorise the previous activations of the hidden units and can be considered to function as one-step time delays. The feedforward connections (dotted lines) are modifiable; the recurrent connections (solid lines) are fixed. Because the recurrent connections are fixed, the Elman network is sometimes called a partially recurrent network.

At a specific time k, the previous activations of the hidden units (at time $k-1$) and the current input (at time k) are used as inputs to the network. At this stage, the network acts as a feedforward network and propagates these inputs forward to produce the output. The standard back-propagation learning rule (Rumelhart and McClelland 1986) can then be employed to train the network. After this training step the activations of the hidden units at time k are sent back through the recurrent links to the context units and saved there for the next training step (time $k+1$). At the beginning of the training process, the activations of the hidden units are unknown. Usually, they are set to one-half of their maximum range. For a sigmoidal activation function the initial values can be set to 0·5. For a hyperbolic tangent activation function they can be equated to 0·0.

In Fig. 1 the external input to the network is represented by $u(k)$ and the network output by $y(k)$. The total input to the ith hidden unit is denoted as $v_i(k)$. The output of the ith hidden unit is denoted as $x_i(k)$. The output of the jth context unit is $x_j^c(k)$. The following equations hold:

$$v_i(k) = \sum_{j=1}^{n} w_{i,j}^x(k-1)x_j^c(k) + w_i^u(k-1)u(k), \quad (1a)$$

$$x_i(k) = f(v_i), \quad (1b)$$

$$x_j^c(k) = x_j(k-1), \quad (1c)$$

$$y(k) = \sum_{i=1}^{n} w_i^y(k-1)x_i(k), \quad (1d)$$

where $w_i^u(\cdot)$, $w_{i,j}^x(\cdot)$ and $w_i^y(\cdot)$, $i, j = 1, 2, \ldots, n$, are the weights of the links, respectively, between the input unit and the hidden layer, between the context layer and the hidden layer, and between the hidden layer and the output unit. f is a sigmoidal activation function. In particular, if input $u(k)$ is delayed by one time step before it is sent to the input unit, $x_j^c(k)$ is replaced by $x_j(k-1)$, and the hidden units are assumed to be linear, the above equations become

$$v_i(k) = \sum_{j=1}^{n} w_{i,j}^x(k-1)x_j(k-1) + w_i^u(k-1)u(k-1), \quad (2a)$$

$$x_i(k) = v_i(k), \quad (2b)$$

$$y(k) = \sum_{i=1}^{n} w_i^y(k-1)x_i(k). \quad (2c)$$

Equations (2 *a*)–(2 *c*) are the state-space description of an nth-order linear dynamic system, where n is the dimension of $\boldsymbol{x}(k) = \{x_i(k)\}$, that is the number of hidden/context units. The order of the model depends on the number of states, which is also the number of hidden units. Equations (2 *a*)–(2 *c*) can be expanded into the following:

$$y(k) = A_1y(k-1) + A_2y(k-2) + \cdots + A_ny(k-n) + B_1u(k-1) + B_2u(k-2) + \cdots + B_nu(k-n). \quad (3)$$

Therefore, theoretically an Elman network is able to model an nth-order dynamic system if it can be trained to do so. To model a system represented by (3) using input–output data, $2n$ input units would be needed if a feedforward neural network is used. For an Elman network the input unit number is one or $n+1$ if the context units are regarded as input units. Thus an Elman network will be significantly smaller in structure than a feedforward network when n is large.

3. Training of the Elman network

The feedback vector $\boldsymbol{x}^c(k) = \{x_i^c(k)\}$ is $\boldsymbol{x}(k-1)$ which is a function of $\boldsymbol{w}^x(k-2)\boldsymbol{x}(k-2) + \boldsymbol{w}^u(k-2)u(k-2)$. Therefore, $\boldsymbol{x}^c(k)$ depends on the weights of previous time instants. When the BP method is applied, the dependence of $\boldsymbol{x}^c(k)$ on the weights should also be taken into account. A BP algorithm that takes care of this dependence is the dynamic BP algorithm (Kuan 1989).

To obtain the dynamic BP algorithm, consider (1 *a*)–(1 *d*). Let the training data set be $(u(k), y_d(k))$, $k = 1, 2, \ldots, N$, where $y_d(k)$ is the desired output of the network. When an input–output data pair is presented to the network at time k, the squared error at the network output is defined as

$$E_k = \tfrac{1}{2}(y_d(k) - y(k))^2. \quad (4)$$

When pattern-based learning is adopted, the weights are modified at each time step k. For $w_i^y(k-1)$, the error gradient is

$$\frac{\partial E_k}{\partial w_i^y(k-1)} = -(y_d(k) - y(k))\frac{\partial y(k)}{\partial w_i^y(k-1)} = -(y_d(k) - y(k))x_i(k). \quad (5)$$

For $w_i^u(k-1)$ and $w_{i,j}^x(k-1)$

$$\frac{\partial E_k}{\partial w_i^u(k-1)} = -\frac{\partial E_k}{\partial y(k)}\frac{\partial y(k)}{\partial x_i(k)}\frac{\partial x_i(k)}{\partial v_i(k)}\frac{\partial v_i(k)}{\partial w_i^u(k-1)}$$
$$= -(y_d(k) - y(k))w_i^y(k-1)f_{v_i}u(k). \quad (6)$$

$$\frac{\partial E_k}{\partial w_{i,j}^x(k-1)} = -\frac{\partial E_k}{\partial y(k)}\frac{\partial y(k)}{\partial x_i(k)}\frac{\partial x_i(k)}{\partial w_{i,j}^x(k-1)}$$
$$= -(y_d(k) - y(k))w_i^y(k-1) \times \frac{\partial x_i(k)}{\partial w_{i,j}^x(k-1)}. \quad (7)$$

In (6) f_{v_i} denotes the derivative of f with respect to v_i.

As discussed above, the internal feedback $\boldsymbol{x}(k-1)$ is dependent on $\boldsymbol{w}^x(k-2)$. Therefore, from (1 *a*) and (1 *b*)

$$\frac{\partial x_i(k)}{\partial w_{i,j}^x(k-1)} = \frac{\partial x_i(k)}{\partial v_i(k)}\frac{\partial v_i(k)}{\partial w_{i,j}^x(k-1)}$$
$$= f_{v_i}\frac{\partial v_i(k)}{\partial w_{i,j}^x(k-1)}$$
$$= f_{v_i}\left\{x_j(k-1) + \sum_{l=1}^{n} w_{i,l}^x(k-1)\frac{\partial x_l(k-1)}{\partial w_{i,j}^x(k-1)}\right\}. \quad (8)$$

If the weight changes are assumed to be small in each iteration, then (8) can be approximately written in a recursive form as

$$\frac{\partial x_i(k)}{\partial w_{i,j}^x(k-1)} = f_{v_i}\left\{x_j(k-1) + \sum_{l=1}^{n} w_{i,l}^x(k-1)\frac{\partial x_l(k-1)}{\partial w_{i,j}^x(k-2)}\right\}. \quad (9)$$

Equation (9) keeps a dynamic trace of the gradient. This is equivalent to applying BP through time (Werbos 1990).

The general weight modification in the gradient descent method is

$$\Delta w = -\eta\frac{\partial E_k}{\partial w}. \quad (10)$$

The dynamic BP algorithm for training an Elman network can thus be summarized as follows.

3.1. *Network*

$$v_i(k) = \sum_{j=1}^{n} w_{i,j}^x(k-1)x_j(k-1) + w_i^u(k-1)u(k),$$
$$x_i(k) = f(v_i),$$
$$y(k) = \sum_{i=1}^{n} w_i^y(k-1)x_i(k).$$

3.2. *Algorithm*

$$\Delta w_i^y(k) = \eta(y_d(k) - y(k))x_i(k), \quad (11\,a)$$

$$\Delta w_i^u(k) = \eta(y_d(k) - y(k))w_i^y(k-1)f_{v_i}u(k), \quad (11\,b)$$

$$\Delta w_{i,j}^x(k) = \eta(y_d(k) - y(k))w_i^y(k-1)\frac{\partial x_i(k)}{\partial w_{i,j}^x(k-1)}, \quad (11\,c)$$

$$\frac{\partial x_i(k)}{\partial w_{i,j}^x(k-1)} = f_{v_i}\left\{x_j(k-1) + \sum_{l=1}^{n} w_{i,l}^x(k-1)\frac{\partial x_l(k-1)}{\partial w_{i,j}^x(k-2)}\right\}, \quad (11\,d)$$

where

$$f_{v_i} = \frac{\partial f}{\partial v_i} = (1 - f^2(v_i)),$$

$$\frac{\partial x_l(k-1)}{\partial w_{i,j}^x(k-2)} = f_{v_i}\sum_{m=1}^{n} w_{j,m}^x(k-2)\frac{\partial x_m(k-2)}{\partial w_{i,j}^x(k-3)}, \quad l \neq i.$$

If the dependence of $\boldsymbol{x}(k-1)$ on $\boldsymbol{w}^x(k-2)$ is ignored, the above algorithm degenerates to the standard BP algorithm, and (11 *d*) and (11 *c*) become

$$\frac{\partial x_i(k)}{\partial w_{i,j}^x(k-1)} = f_{v_i}x_j(k-1), \quad (11\,e)$$

$$\Delta w_{i,j}^x(k) = \eta(y_d(k) - y(k))w_i^y(k-1)f_{v_i}x_j(k-1). \quad (11\,f)$$

4. Relation with the modified Elman network

It has been discovered through simulations that a linear Elman network trained by the standard BP algorithm can only model first-order linear systems (Pham and Liu 1992). It can be seen from (11 *e*) that the gradient only traces back one time step in standard BP. However, the gradient in (11 *d*) traces back indefinitely. Following the theory of Robinson and Fallside (1987) it can be deduced that an Elman network trained by the standard BP algorithm can only represent a first-order finite impulse response (hence first-order dynamic systems). However, 11(*a*)–11(*d*) can train an Elman network to model an infinite impulse response (thus higher-order dynamic systems).

Pham and Liu (1992) introduced self-feedback links with fixed gains to the context units to enable the Elman network to represent higher-order systems (see Fig. 2). The idea of employing self-feedback links was borrowed from the Jordan network (Jordan 1986). The modified Elman network of Pham and Liu (1992) can be described by the following equations.

4.1. *Network*

$$v_i(k) = \sum_{j=1}^{n} w^x_{i,j}(k-1)x^c_j(k) + w^u_i(k-1)u(k), \quad (12a)$$

$$x_i(k) = f(v_i), \quad (12b)$$

$$x^c_i(k) = x_i(k-1) + \alpha x^c_i(k-1), \quad (12c)$$

$$y(k) = \sum_{i=1}^{n} w^y_i(k-1)x_i(k). \quad (12d)$$

4.2. *Algorithm (standard BP)*

$$\Delta w^y_i(k) = \eta(y_d(k) - y(k))x_i(k), \quad (13a)$$

$$\Delta w^u_i(k) = \eta(y_d(k) - y(k))w^y_i(k-1)f_{v_i}u(k), \quad (13b)$$

$$\Delta w^x_{i,j}(k) = \eta(y_d(k) - y(k))w^y_i(k-1)\frac{\partial x_i(k)}{\partial w^x_{i,j}(k-1)}, \quad (13c)$$

$$\frac{\partial x_i(k)}{\partial w^x_{i,j}(k-1)} = f_{v_i}x^c_j(k). \quad (13d)$$

Substituting (13 *d*) into (12 *c*) gives

$$\frac{\partial x_i(k)}{\partial w^x_{i,j}(k-1)} = f_{v_i}x_j(k-1) + \alpha\frac{\partial x_i(k-1)}{\partial w^x_{i,j}(k-2)}. \quad (14)$$

It can be seen that (14) is similar in structure to (9). Although (14) does not provide exactly the same search direction as (9) (note that $\alpha(\partial x_i(k-1)/\partial w^x_{i,j}(k-2))$ appears in (14), but (9) has

$$f_{v_i}\sum_{l=1}^{n} w^x_{i,l}(k-1)\,\partial x_l(k-1)/\partial w^x_{i,j}(k-2)$$

in it), it can provide an infinite impulse response. This is the reason why the modified Elman network of Pham and Liu (1992) was able to model higher-order dynamic systems, even when trained with the standard BP algorithm.

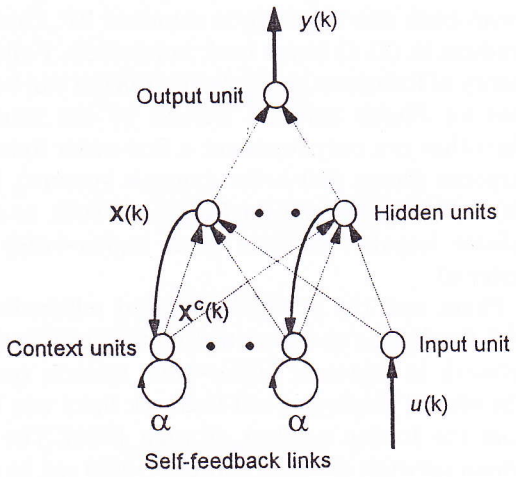

Figure 2. Modified Elman network.

5. Simulation results

5.1. *Linear systems identification*

The dynamic backpropagation algorithm has first been used to identify a second-order linear system and a third-order linear system. For those linear systems, Elman networks with linear hidden units were employed. The second-order linear system model is

$$G(s) = \frac{\omega}{(s+a)^2 + \omega^2}. \quad (15)$$

Its discrete form is

$$y(k) = A_1y(k-1) + A_2y(k-2) + B_1u(k-1) + B_2u(k-2). \quad (16)$$

With the sampling period $T = 0{\cdot}1$ s and the parameters $a = 1{\cdot}0$ and $\omega = 2\pi/2{\cdot}5$, the coefficients of (16) are $A_1 = 1{\cdot}752\,821$, $A_2 = -0{\cdot}818\,731$, $B_1 = 0{\cdot}011\,689$ and $B_2 = 0{\cdot}010\,942$.

A linear network with one input unit, six hidden/context units and one output unit was used to identify the system represented by (16) (the number of hidden/context units, which should at least be equal to the order of the system to be modelled, was taken as six to enable the network to model most practical systems). A training set of 100 data points was produced by sending a uniformly random bounded sequence $|u(k)| \leq (a^2 + \omega^2)/\omega(=2{\cdot}911\,160)$ to the system model with zero initial conditions and recording the output data. After training, the network was tested using a step input signal $u(k) = (a^2 + \omega^2)/\omega$.

The system described by (16) could not be identified by the original Elman network trained by the standard BP algorithm (Pham and Liu 1995). For comparison, the results of Pham and Liu (1995) are shown in Fig. 3. Using the dynamic backpropagation algorithm described above, the responses of the system and the trained

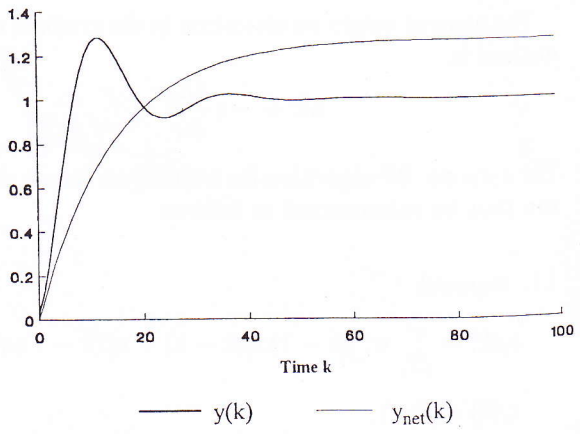

Figure 3. Response of Elman network trained by standard BP (second-order linear system).

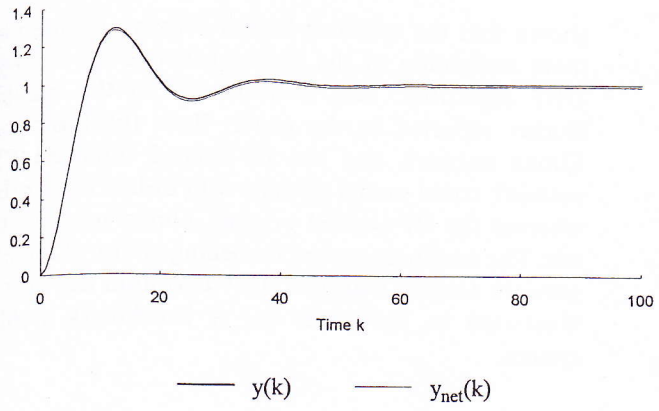

Figure 4. Response of Elman network trained by dynamic BP (second-order linear system).

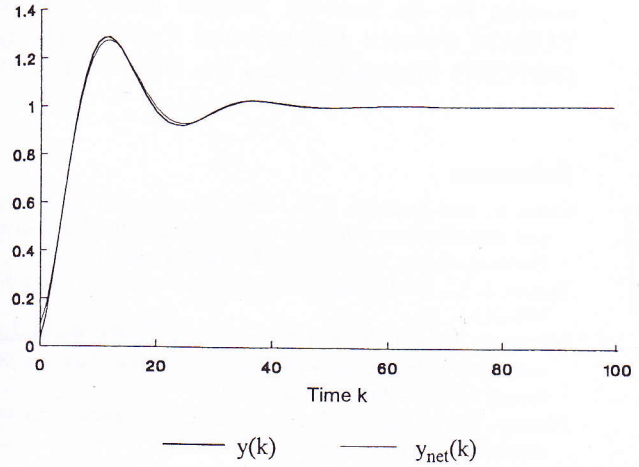

Figure 5. Response of modified Elman network with $\alpha = 0{\cdot}6$ (second-order linear system).

network are shown in Fig. 4. The response of a modified Elman network with the same number of hidden and context units trained by standard BP is plotted in Fig. 5.

The third-order system employed in the simulations has one real pole and two complex poles:

$$G(s) = \frac{1}{(s+b)[(s+a)^2+\omega^2]}. \tag{17}$$

Its discrete form is

$$y(k) = A_1 y(k-1) + A_2 y(k-2) + A_3 y(k-3) + B_1 u(k-1) + B_2 u(k-2) + B_3 u(k-3). \tag{18}$$

With the sampling period $T = 0{\cdot}08$ s, and the characteristic parameters $a = 1{\cdot}0$, $b = 2{\cdot}5$ and $\omega = 2\pi/2{\cdot}5$, the coefficients of (18) are $A_1 = 2{\cdot}627\,771$, $A_2 = -2{\cdot}333\,261$, $A_3 = 0{\cdot}697\,676$, $B_1 = 0{\cdot}017\,203$, $B_2 = -0{\cdot}030\,862$ and $B_3 = 0{\cdot}014\,086$. Again a uniformly random bounded sequence $|u(k)| \leq b(a^2 + \omega^2)$ was applied to the system with zero initial conditions and the output of the system was recorded to produce a training data file. The network

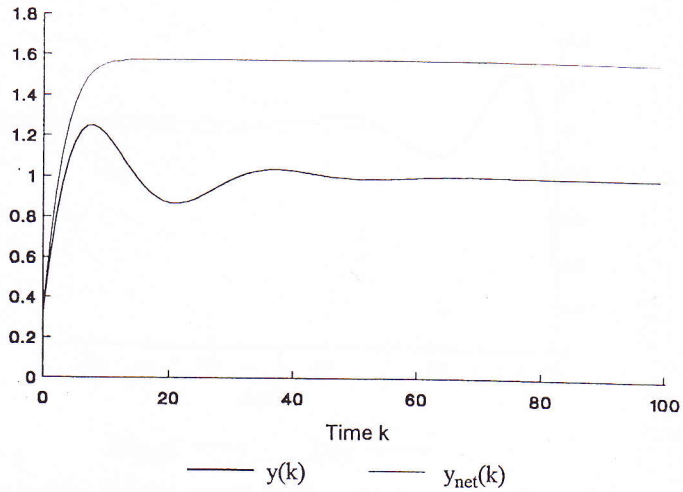

Figure 6. Response of Elman network trained by standard BP (third-order linear system).

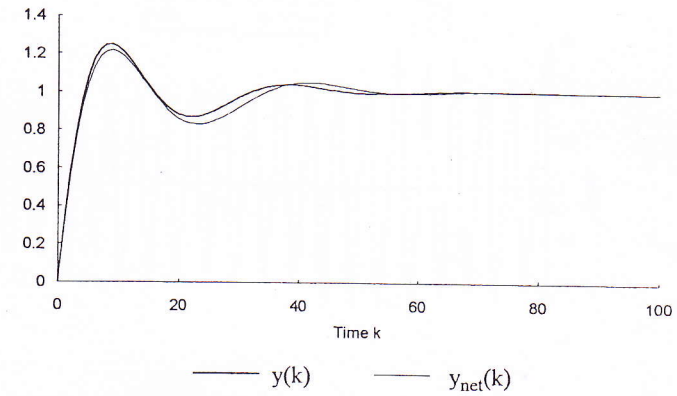

Figure 7. Response of Elman network trained by dynamic BP (third-order linear system).

was tested with a step input $u(k) = b(a^2 + \omega^2)$. The responses of the Elman network trained by the standard BP algorithm, the Elman network trained by the dynamic BP algorithm and the modified Elman network trained by the standard BP algorithm are shown in Figs 6–8. All three networks had six hidden/context units.

The training times for the dynamic BP-trained Elman network and the BP-trained modified Elman network were approximately the same in both simulation examples.

5.2. *Nonlinear system identification*

The dynamic backpropagation algorithm was also tested on the problem of identifying a nonlinear system. An Elman network possessing eight hidden units with hyperbolic activation functions was employed. The nonlinear system model is (Chen and Billings 1994):

$$y(k) = (0{\cdot}8 - 0{\cdot}5\exp(-y^2(k-1)))y(k-1) - (0{\cdot}3 + 0{\cdot}9\exp(-y^2(k-1)))y(k-2) + 0{\cdot}1\sin(3{\cdot}141\,592\,6y(k-1)) + e(k), \tag{19}$$

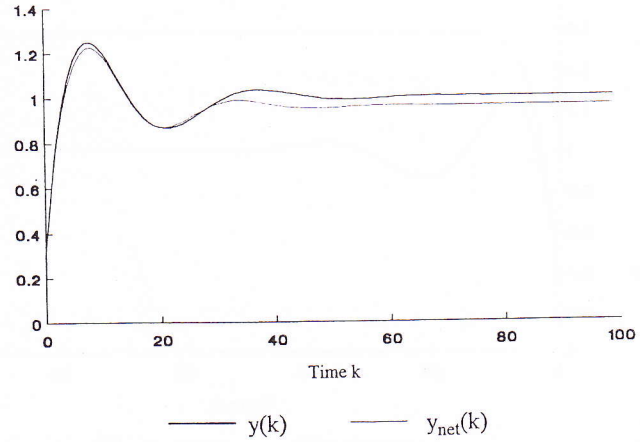

Figure 8. Response of modified Elman network with $\alpha = 0{\cdot}7$ (third-order linear systems).

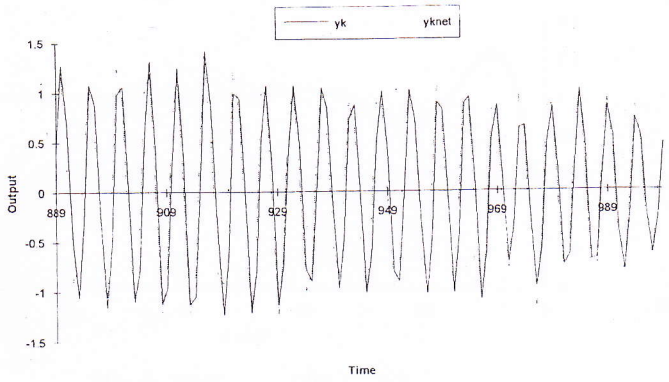

Figure 9. Response of nonlinear process (see (19)).

where $e(k)$ was a gaussian white noise sequence with zero mean and variance equal to 0·01. A database of 1000 data points was created using (19) (initial condition: $y(0) = y(-1) = 0{\cdot}1$). The first 900 data were employed as training data. The last 100 data were reserved as new data to test the trained model. The network was trained using noise $e(k)$ as the input and $y(k)$ as the desired output. The DBP algorithm was employed as the training algorithm. The response of the trained network to the new data is shown in Fig. 9.

6. Conclusion

This paper has presented a dynamic backpropagation (DBP) algorithm for training the Elman network. It has investigated the relationship between the Elman network trained by the DBP algorithm and the modified Elman network proposed by Pham and Liu (1992) and has shown that the modified Elman network is an approximate realization of the Elman network trained by the DBP algorithm. This is borne out in the simulation studies reported in the paper. Both the DBP-trained Elman network and the BP-trained modified Elman network could model systems with orders higher than 1 whereas the BP-trained original Elman network could not. The nonlinear system modelling ability of the Elman network trained with the DBP algorithm has also been illustrated in the paper for a benchmark nonlinear system.

Acknowledgment

The authors would like to thank the European Commission for its financial support under the BRITE-EURAM Focused Fundamental Research programme (PSYCHO Project, Contract No. BRE 2 CT94 0976).

References

Chen, S., and Billings, S. A., 1994, Neural networks for modelling and identification. *Advances in Intelligent Control*, edited by C. J. Harris (London, U.K.: Taylor & Francis), pp. 101–102.

Elman, J. L., 1990, Finding structure in time. *Cognitive Science*, **14**, 179–211.

Hertz, J., Krogh, A., and Palmer, R. G., 1991, Recurrent neural networks. *Introduction to the Theory of Neural Computation* (California: Addison-Wesley), Chapter 7.

Jordan, M. I., 1986, Attractor dynamics and parallelism in a connectionist sequential machine. *Proceedings of the 8th Annual Conference of the Cognitive Science Society*, pp. 531–546.

Kamijo, K., and Tanigawa, T., 1990, Stock price pattern recognition—a recurrent neural network approach. *International Joint Conference on Neural Networks*, Vol. 1, pp. 215–221.

Kuan, C.-M., 1989, Estimation of neural network models. Ph.D. thesis, University of California, San Diego.

Kuan, C.-M., Hornic, K., and White, H., 1989, Some convergence results for learning in recurrent neural networks. Discussion paper, Department of Economics, University of California, San Diego.

Pham, D. T., and Liu, X., 1992, Dynamic system modelling using partially recurrent neural networks. *Journal of Systems Engineering*, **2**, 90–97.

Pham, D. T., and Liu, X., 1995, *Neural Networks for Identification, Prediction and Control* (London, U.K.: Springer-Verlag).

Robinson, A. J., and Fallside, F., 1987, The utility driven dynamic error propagation network. CUED-F-INFENT/TR.1(1987), Engineering Department, Cambridge University, U.K.

Rumelhart, D., and McClelland, J., 1986, *Parallel Distributed Processing: Explorations in the Micro-structure of Cognition*, Vol. 1 (Cambridge, Mass.: MIT Press).

The Math Works Inc., 1994, Introducing Version 2.0 of the Neural Network Toolbox. *IEEE Control Systems Magazine*, **14**, 73.

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