Self-Adaptive Learning Algorithm for Training Multi-Layered Neural Networks and Its Applications

다층 신경회로망의 자기 적응 학습과 그 응용

Wan-Sup Cheung*, Moon-Jae Jho* and Joseph K. Hammond**

정 완 섭*, 조 문 재*, Joseph K. Hammond**

ABSTRACT

A problem of making a neural network learning self-adaptive to the training set supplied is addressed in this paper. This arises from the aspect in choice of an adequate stepsize for the update of the current weigh vectors according to the training pairs. Related issues in this attempt are raised and fundamentals in neural network learning are introduced. In comparison to the most popular back-propagation scheme, the usefulness and superiority of the proposed weight update algorithm are illustrated by examing the identification of unknown nonlinear systems only from measurements.

요 약

본 논문에서는 외부로부터 제공되는 학습데이타에 신경회로망의 자기적응화(self-adaptation)를 이룩하기 위한 접근론이 기술된다. 이러한 문제점은 신경회로망의 학습이론, 즉 현재의 학습 데이타에 적절한 신경회로망의 가중치 벡터들(weight vectors)의 개선 방법론에 기인된다. 이들에 관련된 문제점들의 이론적 검토와 아울러 신경회로망의 학습에 대한 근본적인 요소들이 재조명된다. 현재 가장 널리 이용되고 있는 후방 전달(back-propagation) 학습법과 비교함으로써, 본 연구에서 제안된 자기적응 학습법의 유용성과 우위성을 컴퓨터 모의시험 결과로 입증하게 된다.

I. Introduction

The use of a 'brain-like' model has been recently introduced to 'solve' dynamical systems problems in many applied science fields. The model, referred to as a neural network, is becoming a significant way of providing a systematic basis for tackling a variety of problems. Among many possible neural network models, much attention has centred on multi-layered neural networks, which have been proven to be very successful in pattern recognition [1-3]. This is essentially due to the ability of (adaptively) approximating analytically unknown nonlinear relationship between 'input' patterns and 'desired output' patterns. They provide the major motivation in approaching nonlinear systems problems in what will follow.

Specifically, given a time series of measured inputs $\{u_n\}$ to, and outputs $\{y_n\}$ from a dynamical system, these may be a sequence of paired inputs and outputs corresponding to the input patterns and the desired output patterns. Looking for

* Acoustics and Vibration Lab, KRISS
** Institute of Sound and Vibration Research University of Southampton
접수일자 : 1993년 12월 23일

these associations is essentially pattern recognition. Thus, it is logical to examine the effectiveness of neural network models in system identification. In Fig.1, the input training patterns $\{s_{I,n}\}$ is seen to be reconstructed from the measured system inputs and output using three delay taps, that is

$$s_{I,n} = \{y_n, y_{n-1}, y_{n-2}, u_n, u_{n-1}, u_{n-2}\} \quad \text{for time index } n, \tag{1}$$

and the desired model outputs $\{y_{d,n}\}$ is directly assigned to the measured system outputs

$$y_{d,n} = y_{n+1}. \tag{2}$$

This reconstruction is referred to as the "time-invariant imbedding method" [4]. The three delaye tpas in (1) are chosen to 'safely' reconstruct the input patterns in neural network-based dynamic system modeling. This choice is due to Takens and Maine theorems [5,6] for chaotic systems so as to obtain a "smooth" trajectory without any crossing point in reconstructed state space, but they may be not necessarily limited to such systems. The choice of three delays will be applied to deterministic, non-chaotic systems in this study.

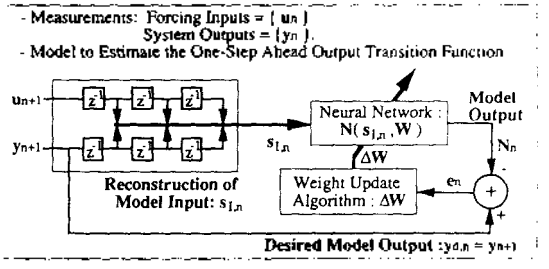


Fig. 1. Conceptual setup of neural network-based system identification model.

In Fig. 1, the neural network is chosen to match the measured system outputs $\{y_{n+1}\}$ for its inputs $s_{I,n} = \{y_n, y_{n-1}, y_{n-2}, u_n, u_{n-1}, u_{n-2}\}$. This is a natural representation of discrete nonlinear dynamical systems, i.e. the one-step ahead output transition function

$$y_{n+1} \longleftrightarrow N_n = N(s_{I,n}, W). \tag{3}$$

Here, a problem to be addressed in this study is obvious : How to estimate the set of neural network weigths W well matched with the series of training pairs $\{s_{I,n}$ and $y_{d,n}\}$? In Section 2, the fundamentals of multi-layered neural networks and the difficulties arising in using the popular back-propagation and its relatives are highlighted. In Section 3, a new approach, referred to as the self-adaptive learning algorithm, is suggested to overcome them. In Section 4, simulation results and discussions are made in comparison with those of the conventional algorithms. Finally, concluding remarks are summarised.

Ⅱ. Multi-Layered Neural Network and Learning Algorithms

2.1 Multi-Layered Neural Networks

In this paper, a three-layered neural network, denoted by $\mathcal{N}_{N_I-N_1-N_2}$ where the subscripts N_I, N_1 and N_2 are the size of the input layer, the hidden layer and the output layer respectively, is considered. It performs a cascaded nonlinear mapping of the input vector s_I onto the output N as shown in Fig. 2.

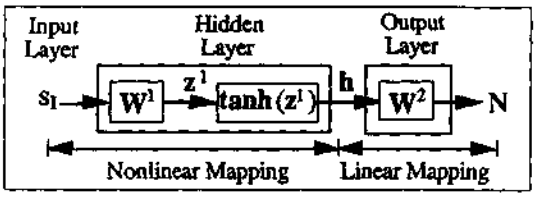


Fig. 2. Three-layered neural network, $\mathcal{N}_{N_I-N_1-N_2}$.

Two weight matrices W^1 and W^2 corresponding to the hidden layer and the output layer participate in the three-layered neural network. The input s_I is projected to the first weight W^1 and then

the hidden layer output, denoted by the vector $\mathbf{h}$, is obtained by evaluating a nonlinear mapping bounded in $[-1, 1]$,

$$\mathbf{h} = \tanh(\mathbf{W}^1 \cdot \mathbf{s}_I) = [\tanh(\mathbf{w}_1^1 \cdot \mathbf{s}_I), \tanh(\mathbf{w}_2^1 \cdot \mathbf{s}_I), \ldots, \tanh(\mathbf{w}_{N_1}^1 \cdot \mathbf{s}_I)]^T. \quad (4)$$

This hidden layer mapping converts the input space of dimension N_I onto another bounded space with dimension N_1. The functional characteristics of this mapping is wholly dependent on the hidden layer matrix $\mathbf{W}^1$ so that different choices of the weight result in different nonlinear functionals described by the hidden layer state $\mathbf{h}$. This is quite a useful property in neural network-based nonlinear modelling, which has provided the major motivation of exploring the.

Specifically, the second weight matrix $\mathbf{W}^2$ is used to reconstruct the output space of dimension N_2 from the bounded hidden-layer space of size N_1 in the hidden layer. It is the reason for selecting a simple linear mapping in the output layer, referred to as the "weighted-sum" scheme,

$$\mathbf{N} = \mathbf{W}^2 \cdot \mathbf{h} \quad (5)$$

This operation is one of fundamentals in the elementary unit of neural networks, called the neuron. This scheme makes some difference in the nonlinear system modelling problems. When the dynamic range of the desired model output is larger than 1.0, the weighted-sum scheme can cover the range. But, when the output layer consists of the sigmoid unit as in (4) then the maximum range of the output is limited in $[-1, 1]$ so that this network can not cover the output range of larger than 1.0. The linear output model shown in Fig. 3 is seen to be the more plausible model for nonlinear dynamical systems representation.

The above three-layered neural network basically performs the two-stage mapping : the bounded nonlinear mapping of the input state onto the hidden layer state and the linear mapping to reconstruct the network output from the bounded hidden state. This feature is exploited to estimate an unknown nonlinear relationship between the measured inputs and outputs for unknown nonlinear dynamical systems. Here one point is obvious that the bounded hidden layer state (4) is understood to be basis functionals for the reconstruction of the network output and that their functional characteristics can be determined by selecting the hidden layer weight matrix. The network output is seen to be reconstructed by the weighted-sum of the basis functionals using the weight matrix of the output layer. This two-stage nonlinear mapping leads to the estimation of an unknown discrete-time representation imbedded in nonlinear system measurements. It is apparent that a key issue here is how to estimate the set of neural network weights adequate for a series of measurements of nonlinear dynamical systems. It is referred to as the neural network learning algorithm, which is further addressed in the rest of this section.

2.2 Back-Propagation Algorithm and Its Relatives

Most learning laogrithms for multi-layered neural networks are based on the geometrical features of output errors between the desired and current outputs of the neural networks. Unfortunately, the errors of the multi-layered neural networks are non-linearly dependent on the weights. This does not always enable us easily to see such geometrical features as local minima or a global minimum. However, the simplest scheme, that is the steepest descent method, has been widely used to estimate the weight as, for example, in the work of Nuguen and Widrow [7] and Narendra and Parthasaraty [8]. It is conceptually identical to the back-propagation algorithm, introduced by Rummelhart *et al.* [9]. Given the squared value of the model output error

$$J_n = \frac{1}{2} \| \mathbf{y}_{d,n} - \mathbf{N}(\mathbf{s}_{I,n}, \mathbf{W}_n) \|^2, \quad (6)$$

the weight update scheme of the back-propagation algorithm is described by

$$\mathbf{W}_{n+1} = \mathbf{W}_n + \mu \cdot \mathbf{D}_n = \mathbf{W}_n - \mu \cdot \partial J_n / \partial \mathbf{W}_n. \quad (7)$$

A positive constant μ is the learning rate (step size) and $\mathbf{D}_n$ denotes the search direction.

At the beginning of the research, the back-propagation algorithm was chosen to understand the fundamentals of training the above three-layered neural network for nonlinear system modelling. Two well-known systems-the van der Pol oscillator and the Henon map-were chosen for study. When the weight update scheme (7) is used to identify nonlinear systems form measurements, several limitations arise (refer to reference [10] for detailed information). Important factors first encountered from these attempts are as follows: the initially chosen weights $\mathbf{W}_0$ and the learning rate μ for each application, referred to as the neural network training factors, play critical roles in the asymptotic accuracy of chosen neural network models for system representation and their convergence rate in estimating the adequate weight vectors.

An intuitive weight estimation scheme was first used in the research, referred to as the iterative learning algorithm. Fig. 3 shows the conceptual set-up of the algorithm. It involves a delicate 'house-keeping' scheme for choosing the initial weights $\mathbf{W}_{0,itn}$ for each iteration, searching the best fit weight set, and decreasing the current learning rate.

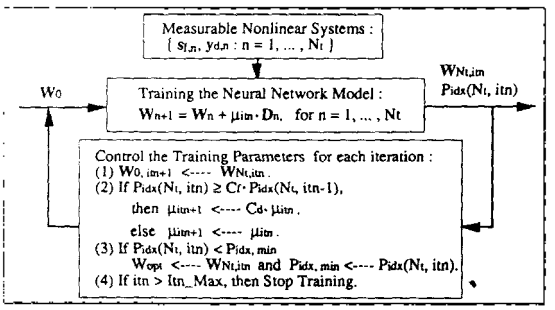


Fig. 3. Schematic diagram of the iterative learning algorithm: C_f = the control factor of the performance factor, C_d = the decay constant of the reduction rate, itn = the iteration index, and μ_{itn} = the learning rate (step-size) at iteration index itn.

Fig. 3. Schematic diagram of the iterative learning algorithm: C_f = the control factor of the performance factor, C_d = the decay constant of the reduction rate, itn = the iteration index, and μ_{itn} = the learning rate (step-size) at iteration index itn.

In Fig. 3, the performance index denoted by $P_{idx}(\cdot)$ is defined by the mean squared value of normalised errors at the time index n

$$P_{idx}(n, itn) = \frac{1}{n} \sum_{i=1}^{n} \{ \sum_{i=1}^{N_2} \| y_{d,i}(n) - N_i(n) \|^2 / \sigma_i^2 \} \quad (8)$$

where σ_i^2 denotes the variance of the i-th desired output and 'itn' is the iteration index. Specifically, the learning rate μ_{itn+1} for the next iteration is determined by the comparison between the current performance index $P_{idx}(Nt, in)$ and the reference value $C_f \cdot P_{idx}(n, itn-1)$, that is the previous one multiplied by the control factor C_f (slightly larger than one, e.g. $C_f = 1.002$). If $P_{idx}(Nt, itn) \geq C_f \cdot P_{idx}(n, itn-1)$, then the learning rate is relatively decreased by the decay factor $C_d < 1$. Of cause, if the decrease mechanism of the learning rate is chosen to be inactive in the above house-keeping scheme, the iterative algorithm becomes identical to the back-propagation algorithm.

The starting weight $\mathbf{W}_0$ at the first iteration is initialised by a 'small' random variable uniformly distributed in $[-0.01, 0.01]$ so as to minimise the effect of choice of the initial weights on the performance of the learning algorithm. Moreover, the initial learning rate is assigned by a large value in order to obtain a 'fast' searching at the beginning of training stages. Therefore, the iterative learning scheme shown in Fig. 3 can reduce

at least the amount of labour needed to select the adequate network training factors-the initial weight and the learning rate for each application through repeated trial-and-errors.

The learning rate in this learning algorithm gradually decreases whenever the performance is no longer improved. This scheme seems to be safer for 'fine-tuning' of the network weights than the global constant case in the back-propagation or its relatives [7-9]. The smaller learning rate is intuitively understood to be effective when the estimated weights are close to the 'minimal' condition. The geometrical feature in this specific case may be seen by a quadratic approximation of the cost function. This will be further addressed in the following sections.

Ⅲ. Self-Adaptive Learning Algorithm

3.1 Local Minimum and Self-Adaptive Learning Rate

The back-propagation algorithm and the iterative learning algorithm above consist of two basic strategies : the search direction and the step-size for the weight update. To improve the searching direction is a multi-dimensional search problem while the determination of the step-size is an one-dimensional one. The latter problem is much simpler than the former in a sense of optimisation (minimisation/maximisation)[15]. Howeverm little is known about this in the neural network field. Specifically, this study considers the step-size (learning rate) as an one-dimensional optimisation variable which will be determined for each training pair. Given the squared value of the normalised error attempted to be minimised

$$J_n = \frac{1}{2} \sum_{i=1}^{M} \| y_{d,i}(n) - N_i(n) \|^2 / \sigma_i^2 \qquad (9)$$

then the search direction is given as the steepest descent direction of (9) with respect to the current weight

$$D_n = -\partial J_n / \partial W_n. \qquad (10)$$

The normalised form (7) is chosen in this research due to the advantages that it can be used to test not only the performance of a trained neural network but also to calculate the chi-squared value for the goodness-of-fit of the estimated weight. Given the current learning rate (to be determined) and the current search direction (10), the weight is updated in the same way as the above algorithm

$$W_{n+1} = W_n + \mu_n \cdot D_n = W_n - \mu_n \cdot \partial J_n / \partial W_n. \qquad (11)$$

To approach the problem of determining the current learning rate μ_n, this paper considers an intermediate cost function which is obtained by substituting (11) into (9)

$$J_1(n, \mu_n) = \frac{1}{2} \sum_{i=1}^{M} \| y_{d,i}(n) - N_i(s_{I,n}, W_n + \mu_n \cdot D_n) \|^2 / \sigma_i^2$$
$$= \frac{1}{2} \sum_{i=1}^{M} \| e_i(n, \mu_n) / \sigma_i^2 \|^2 \qquad (12)$$

It is noted from (12) that the learning rate μ_n can be obtained in such a way to minimise the intermediate cost function. This shows that the current learning rate can be determined from the 'local' minimum of the intermediate cost function $J_1(n, \mu_n)$ with respect to the unknown μ_n. This condition may be described by

$$\partial J_1(n, \mu_n) / \partial \mu_n = 0 \quad \text{and} \quad \partial^2 J_1(n, \mu_n) / \partial^2 \mu_n \geq 0. \qquad (13)$$

This is the sufficient condition for the (local) minimum of the cost function with a 'locally bowl-like' geometrical shape and is used to determine the unknown learning rate. Here one point is obvious that the learning rate is adaptively determined from the current training pair $\{s_{I,n}, y_{d,n}\}$ and the current network weights W_n. To empha-

ause this feature, it is referred to as the 'self adaptive' learning rate.

3.2 Self-Adaptive Learning Rates

3.2.1 Self-Adaptive Learning Rate for a Single-Output Neural Network

First, a single-output neural network denoted by $\mathcal{N}_{N_0-N_1-1}$ is considered. In this case $N2=1$ in equation (12). By applying the first sufficient condition of (13) to the cost function (12), we obtain

$$e_1(n, \mu_n) = 0 \quad \text{and} \quad \partial N(n, \mu_n)/\partial \mu_n = 0 \tag{14}$$

The first equation is easily shown to satisfy the positive semi-definiteness of (13) since

$$\partial^2 J_1(n, \mu_n)/\partial^2 \mu_n = \| \partial N(n, \mu_n)/\partial \mu_n \|^2 \geq 0 \tag{15}$$

This is sufficient to determine the current learning rate. But, semi-definiteness is not seen from the second equation of (14). Thus, the learning rate is obtained by solving the following equation which is approximated to the first order

$$\begin{aligned} e_1(n, \mu_n) = 0 &= y_{d,n} - [\mathbf{W}_n^o + \mu_n \cdot \mathbf{D}_n^o] \cdot \mathbf{h}(\mathbf{s}_{I,n}, \mathbf{W}_n^I + \mu_n \cdot \mathbf{D}_n^I) \\ &\simeq e_n - \mu_n \cdot [\mathbf{D}_n^o \cdot \mathbf{h}_n + \mathbf{W}_n^o \cdot \Delta \mathbf{h}_n] \end{aligned} \tag{16}$$

where $e_n = [Y_{d,n} - \mathbf{W}_n^o \cdot \mathbf{h}(\mathbf{s}_{I,n}, \mathbf{W}_n^I)]$, the model output error

$\mathbf{W}_n^o$ and $\mathbf{W}_n^I$ = the weights of the output and hidden layers,

$\mathbf{D}_n^o$ and $\mathbf{D}_n^I$ = the search directions of the output and hidden layer weight,

$\mathbf{h}_n$ = the output of the hidden layer,

and $\Delta \mathbf{h}_n$ = the first order approximation of the hidden layer output.

The superscripts 'o' and 'I' denote the indices of the output and hidden layer. The search direction of the output layer and the first order approximation are given as

$$\mathbf{D}_n^o = -\partial J_1(n, \mu_n)/\partial \mathbf{W}_n^o = (e_n/\sigma^2) \cdot \mathbf{h}_n \tag{17}$$

and

$$\Delta \mathbf{h}_n = \mathbf{h}(\mathbf{s}_{I,n}, \mathbf{W}_n^I + \mu_n \cdot \mathbf{D}_n^I) - \mathbf{h}_n = [\Delta h_{1,n}, \ldots, \Delta h_{N1,n}]^T \tag{18}$$

where the i-th element of $\Delta \mathbf{h}_n$ is

$$\Delta h_{i,n} = (e_n/\sigma^2) \cdot \mathbf{W}_{i,n}^o \cdot (1 - h_{i,n}^2)^2 \cdot \| \mathbf{s}_{I,n} \|^2 \tag{19}$$

By substituting equations (17)-(19) into (16), we obtain the learning rate at time n

$$\begin{aligned} \mu_n &= e_n / [\mathbf{D}_n^o \cdot \mathbf{h}_n + \mathbf{W}_n^o \cdot \Delta \mathbf{h}_n)] \\ &= \sigma^2 / [\| \mathbf{h}_n \|^2 + \sum_{i=1}^{N_1} |\mathbf{W}_{ij,n}^o|^2 \cdot (1 - h_{i,n}^2)^2 \cdot \| \mathbf{s}_{I,n} \|^2]. \end{aligned} \tag{20}$$

It shows that the current learning rate is determined by the current states of the neural network and that the learning rate is self-adaptive to the current training pair, not the constant chosen by the designer. It should be noted that the self-adaptive learning rate is proportional to the variance of the desired output. If a unit variance is chosen, i.e., $\sigma^2 = 1$, then one obtains the self-adaptive learning rate for the popular back-propagation algorithm.

3.2.2 Self-Adaptive Learning Rate for a Multiple-Output Neural Network

The three layered neural network with the with multiple linear outputs denoted by $\mathcal{N}_{N_0-N_1-N_2}$ is considered. In this case, the sufficient condition (13) is expressed by

$$\partial J_1(n, \mu_n)/\partial \mu_n = -\sum_{i=1}^{N_2} [e_{1,i}(n, \mu_n) \cdot \partial N_i(n, \mu_n)/\partial \mu_n]/\sigma_i^2 = 0 \tag{21}$$

and

$$\begin{aligned} \partial^2 J_1(n, \mu_n)/\partial^2 \mu_n = &\sum_{i=1}^{N_2} [\partial N_i(n, \mu_n)/\partial \mu_n]^2/\sigma_i^2 \\ &- \sum_{i=1}^{N_2} [e_{1,i}(n, \mu_n) \cdot \partial^2 N_i(n, \mu_n)/\partial^2 \mu_n]/\sigma_i^2 \geq 0. \end{aligned} \tag{22}$$

They are not as simple as the case of the previous single-output network model. To determine the current learning rate, the first order approximation of the output of the neural network

$$N_i(\mathbf{s}_{i,n}, \mathbf{W}_n + \mu_n \cdot \mathbf{D}_n) \cong$$
$$\mathbf{W}^o_{i,n} \cdot \mathbf{h}_n + \mu_n \cdot [\, \mathbf{D}^o_{i,n} \cdot \mathbf{h}_n + \mathbf{W}^o_{i,n} \cdot \Delta \mathbf{h}_n \,] \qquad (23)$$

is used. The subscript 'i' denotes the index of neurons. Substituting (23) into (21), the approximate cost function is obtained as

$$J_1(n, \mu_n) \cong \sum_{i=1}^{N_2} \{ e_{i,n} - \mu_n \cdot [\, \mathbf{D}^o_{i,n} \cdot \mathbf{h}_n + \mathbf{W}^o_{i,n} \cdot \Delta \mathbf{h}_n \,] \}^2 / 2\sigma_i^2 \qquad (24)$$

which is a quadratic form that satisfies positive semi-definiteness. In (24), $e_{i,n}$ is the i-th model output error evaluated for the n-th training pair. It means that the current learning rate is determined by the minimum position of (24). Thus, we obtain the learning rate for the multiple output neural network model

$$\mu_n = \sum_{i=1}^{N_2} (e_{i,n}/\sigma_i^2) \cdot \{ \mathbf{D}^o_{i,n} \cdot \mathbf{h}_n + \mathbf{W}^o_{i,n} \cdot \Delta \mathbf{h}_n \} \Big/ \sum_{i=1}^{N_2} \{ \mathbf{D}^o_{i,n} \cdot \mathbf{h}_n + \mathbf{W}^o_{i,n} \cdot \Delta \mathbf{h}_n \}^2 / \sigma_i^2$$
$$\cong \sum_{i=1}^{N_2} (e_{i,n}^2/\sigma_i^2) \cdot [\, \{ \|\mathbf{h}_n\|^2 + \sum_{j=1}^{N_1} |\mathbf{W}^o_{ij,n}|^2 \cdot (1 - h_{j,n}^2)^2 \cdot \|\mathbf{s}_{i,n}\|^2 \} / \sigma_i^2 \,] \Big/ \sum_{i=1}^{N_2} (e_{i,n}^2/\sigma_i^2) \cdot [\, \{ \|\mathbf{h}_n\|^2 + \sum_{j=1}^{N_1} |\mathbf{W}^o_{ij,n}|^2 \cdot (1 - h_{j,n}^2)^2 \cdot \|\mathbf{s}_{i,n}\|^2 \} / \sigma_i^2 \,]^2 \qquad (25)$$

It shows that the current learning rate is determined from the normalised output error and the current states of the neural network. The contribution of each error to the learning rate is related to the squared value of the normalised error. It is obviously seen that the current learning rate (25) is self-adaptive to the current training pair, not the constant selected by the supervisor of the network. It is interesting to note that in case of $N_O = 1$ the learning rate yields the same result as the learning rate (20) of the single-output network. If all $\sigma_i^2 = 1$, then the self-adaptive learning rate for the back-propagation algorithm is obtained. The weight update scheme based on the self-adaptive learning rate is referred to as the self-adaptive learning algorithm.

Ⅳ. Simulation Results and Discussions

To examine the learning algorithms presented above, two nonlinear systems are considered which have different characteristics : the Henon map and the van der Pol oscillator.

4.1 Henon Map

This is described by the plane motion in the delayed coordinate $\mathbf{x}_n = [\, x_{1,n}, x_{2,n} \,]^T$

$$x_{2,n} = 1.4 - x_{1,n}^2 + 0.3\, x_{1,n-1}, \quad x_{i,n} = x_{2,n-1} \qquad (26)$$

This motion does not involve any external input. As is well known, this system has unpredictable long-term behaviour [11-14], i.e, random characteristics arising from the deterministic process of (26). This raises a basic question as to whether the motion can indeed be predicted by parametric models. Farmer and Sidorowich [11] studied this using linear auto-regressive models and pointed out their limitations. The use of the neural network model as a parametric nonlinear model, which provides the ability of adaptively building up basis functionals and then reconstructing the model output from the functionals, will be examined in this section.

For the above problem, the sequence of the training pairs whose inputs are reconstructed by the three delayed taps

$s_n = [x_n^T, x_{n-1}^2, x_{n-1}x_{n-2}, x_{n-2}^2, \cdots, x_{1,n-1}, x_{1,n-2}, x_{1,n-3}]^T$

and $y_{d,n} = x_{2,n} (= x_{1,n+1})$

for $n = 3, \cdots, N_t + 2$ $(N_t = 512)$ (27)

are used to train the neural network model M_{4-12-1}. Nt is the record length of the training set. This training set is used to compare the relative performance between three different algorithms-the back-propagation, iterative learning and self-adative learning algorithm. The network weights of the model are initialised by the small random variable uniformly distributed in $[-0.01, 0.01]$ to minimise the effect of choice of initial weights.

Fig. 4 (a) shows the performance indices (the mean squares of normalised model errors) of the first two algorithms during 400 iterations. At the last iteration, the performance indices are Pidx $(Nt, 400) = 9.57 \times 10^{-3}$ for the back-propagation algorithm with the constant learning rate $(\mu = 0.1)$ and Pid $\times (Nt, 400) = 2.75 \times 10^{-3}$ for the iterative algorithm respectively. The latter algorithm is shown to give a more accurate model than the former. The reason is due to the delicate housekeeping scheme shown in Fig. 3. It reduces the learning rate of the next iteration whenever the change of the performance index is not improved any more.

Fig. 4 (b) shows the trend of the learning rate over the iteration. This reduction scheme for the 'fine-tuning' of the network weights is shown to be more effective than the constant learning rate employed in the back-propagation algorithm. Specifically, the steep decrease of the performance index is seen each modification of the learning rate in Fig. 4.

From the results above, the learning parameters (μ, C_f, and C_d) are shown to participate in the algorithms. They were predetermined through repeated trial-and-error. In practice, much computation time and expert experience are essential in finding adequate parameters for each application.

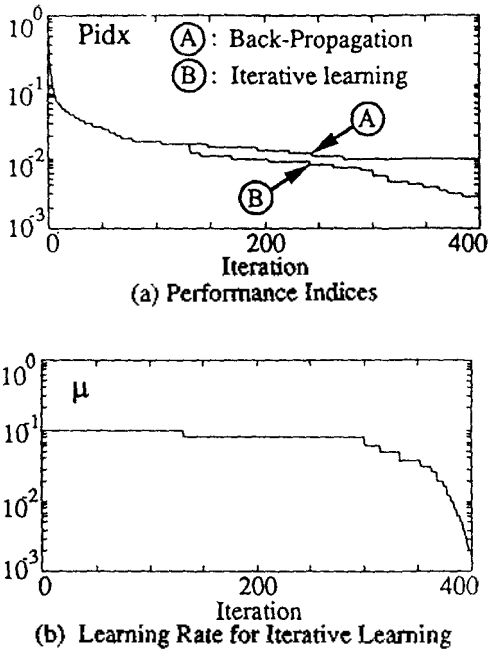


Fig. 4. Comparison between the back-propagation algorithm and the iterative learning algorithm: The constant learning rate in the back-propagation algorithm, $\mu = 0.1$, the control factor of the performance index, $C_f = 1.002$, and the decay constant of the learning rate, $C_d = 0.7943$.
(a) Performance Indices
(b) Learning Rate for Iterative Learning

From the argument that the underlying difficulty in neural network modelling may be resolved if the learning rate is self-adaptive to the sequence of training pairs, the learning prolem in the optimisation sense has been reconsidered and the self-adaptive learning algorithm has been presented in Section 3.

The neural network model for the Henon mapping of (26) has a single output. Thus, the self-adaptive learning rate for each training pair is computed by the formulation for single output network models, described in (20). Fig. 5 (a) shows the comparison of the performance between the iterative learning algorithm and the self-adaptive one. This indicates that the self-adaptive learning algorithm provides faster convergence than the

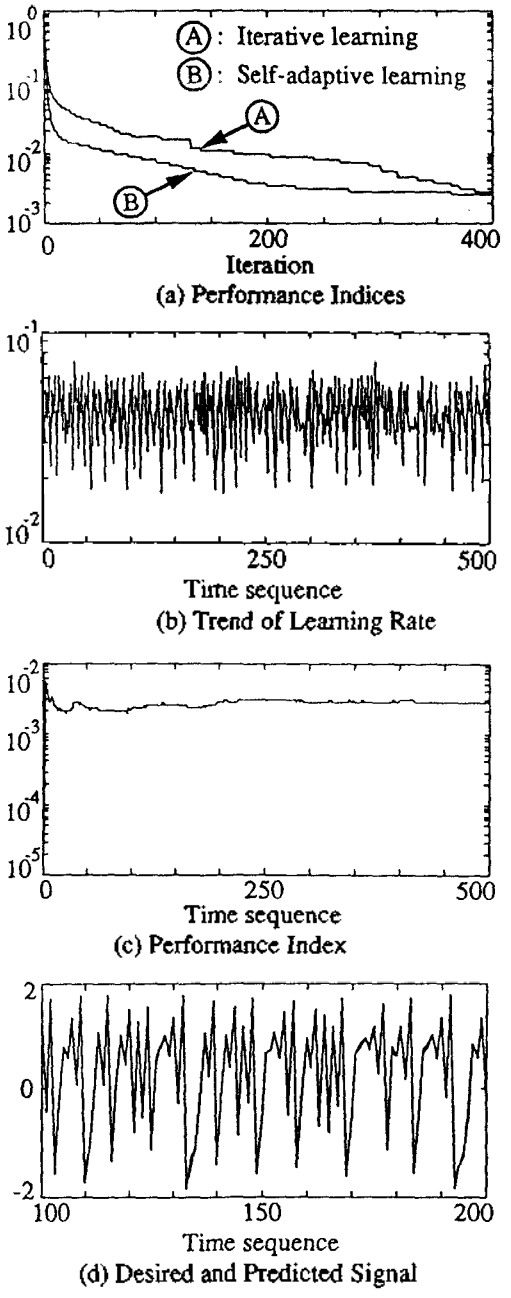


Fig. 5. Comparison between the iterative learning algorithm and the self-adaptive learning algorithm : The control factor of the performance index $C_f = 1.002$ and the decay constant of the learning rate, $C_d = 0.7943$.
(a) Performance Indices (b) Trend of Learning Rate
(c) Performance Index
(d) Desired and Predicted Signal

iterative one. At the final iteration, the performance index is found to be $P_{idx}(N_t, 400) = 2.62 \times 10^{-4}$ which is only a little different to that of the iterative algorithm ($Pidx(Nt, 400) = 2.75 \times 10^{-4}$). Fig. 5 (b) illustrates the trend of the self-adaptive learning rate over the training sequence.

Fig. (c) shows the performance index when the network weights obtained after 400 iterations are used to predict the time series. Any change of the performance index in this case is not seen over the time sequence. It may mean that the estimation process of the network weights (at final iteration) has reached the steady state. The desired and predicted signals are plotted in Fig. 5 (d). Any noticeable difference is not observed. It reveals that the use of the three-layered neural network model is successful in developing a mathematical model to adequately describe the 'chaotic' signal generated by the Henon mapping.

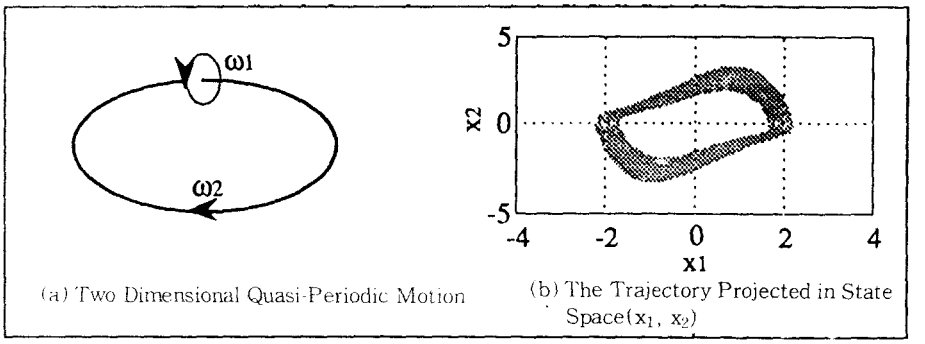
4.2 The van der Pol Oscillator

A periodic forcing input $u(t)$ is introduced to this system, which is described in continuous time

$$\dot{x}_1 = \dot{x}_1, \qquad \dot{x}_2 = (1 - x_1^2) \cdot x_2 - x_1 + u(t)$$
$$u(t) = 0.5 \sin(1.1t). \tag{28}$$

The dynamical behaviour of this system manifests a two dimensional 'quasi-periodic' motion depicted in Fig. 6 (a), which lies on a 'torus' in three-dimensional space $\{x_1, x_2, u\}$. When it is projected onto the state space $\{x_1, x_2\}$ the trajectory is confined to an annular-like region and is uniformly distributed in this region, as shown in Fig. 6 (b). When the inputs $\{u_n\}$ and the states $\{x_{1,n}, x_{2,n}\}$ are measurable at every time interval ΔT, a problem of estimating a discrete system representation, i.e. the equivalent mapping of the continuous system (28) onto the discrete form.

To approach the problem, the three-layered neural network model $\mathcal{N}_{10-30-2}$ is considered as a nonlinear parametric model for the one-step ahead



(a) Two Dimensional Quasi-Periodic Motion

(b) The Trajectory Projected in State Space(x_1, x_2)

Fig. 6. Two dimensional quasi-periodic motion.

state transition function. The sequence of the network training pairs is reconstructed by the three-delay taps of inputs $\{u_n\}$ and outputs $x_n = [x_{1,n}, x_{2,n}]^T$ sampled at every time interval $\Delta T = 0.1$ [s] as

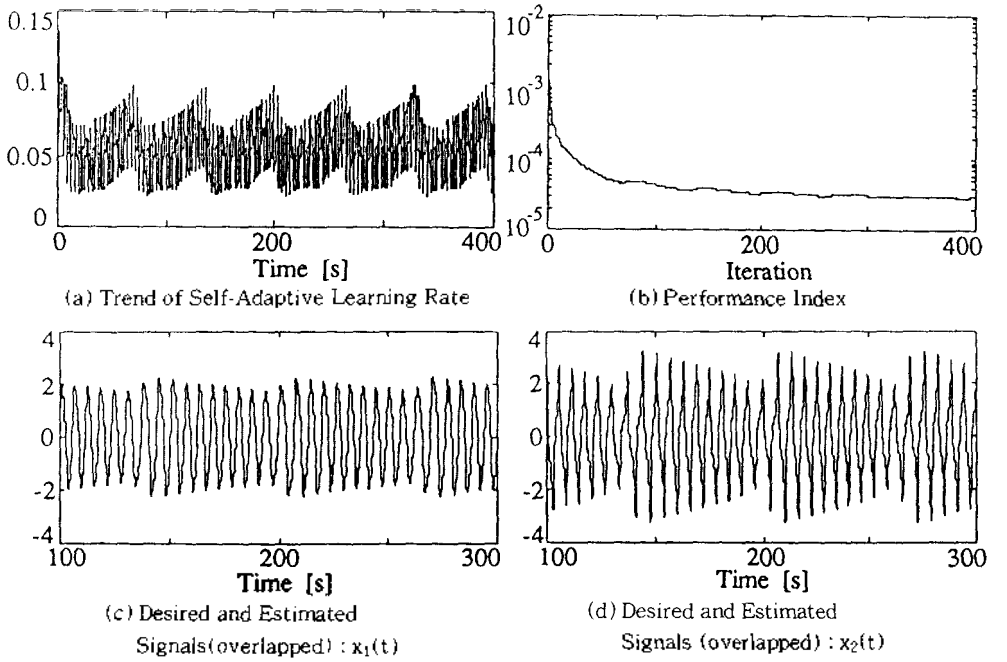
$$s_{1,n} = [x_{1,n}, x_{2,n}, x_{1,n-1}, x_{2,n-1}, x_{2,n-2}, u_{n+1}, u_n, u_{n-1}, u_{n-2}]^T$$

and $y_{d,n} = x_{n+1} = [x_{1,n+1}, x_{2,n+1}]$

for $n = 2, \cdots, N_t + 1$ $(N_t = 4096)$ (29)

As given in (29), the network has two outputs. Thus, the self-adaptive learning rate needed to update the current network weights for each training pair is computed by the expression of (25) for multiple output neural network models. This leads to straightforward estimation of the nuknown state transition function $x_{n+1} = N(s_{1,n}, W)$ without any of the difficulties of finding adequate learning parameters encountered in the previous example.

Fig. 7 (a) shows an example of the self-adapt-



(a) Trend of Self-Adaptive Learning Rate

(b) Performance Index

(c) Desired and Estimated Signals(overlapped) : $x_1(t)$

(d) Desired and Estimated Signals (overlapped) : $x_2(t)$

Fig. 7. Simulation results of the self-adaptive learning algorithm.

ive learning rate evaluated by the expression (25) for the training set. During 400 repeated iterations, this self adaptive learning rate is used to estimate the network weights. Fig. 7 (b) shows the performance achieved using the learning rate. At the final iteration, the performance index is $P_{idx}(N_t, 400) = 3.0 \times 10^{-5}$. It is a satisfactory result in the statistical sense [15] since the chi-square value ($N_t \times P_{idx}(N_t, 400) = 0.12$) is of the order of 10^{-1}. This becomes more evident as observed from the comparison of the desired and estimated outputs in Fig. 7 (c) and (d). Any difference in this result is invisible.

V. Concluding Remarks

The basic issue of how to determine the learning rate (step size) to update the network weights is addressed and two approaches, referred to as the iterative learning algorithm and the self-adaptive learning algorithm, are proposed. The iterative learning algorithm gives a safe way of finding the unkown local minimum of model output errors by gradually reducing the learning rate. This approach basically involves the exploitation of a local geometrical feature of error surfaces. The local approximation of the output errors leads to the self-adaptive learning algorithm. The effectiveness of the proposed algorithms is illustrated in the computer simulation results presented in this paper.

It is interesting to note that the use of the neural network model above does not need any prior system knowledge. Moreover, the simple three-layered neural network model is shown to work well. The reconstruction of the model input vector using the three delayed taps and the determination of the size of the hidden layer, $N_l = 3 \times N_I$ (N_I = the size of the input vector), have been made heuristically, not theoretically. These heuristic rules are shown be successful in the results above, but further effort to more logically understand them must be made in future work.

References

1. B. Widrow, R.G. Winter, and R.A. Baxter, "Layered Neural Nets for Pattern Recognition," *IEEE Trans. Acoust., Speech, Signal Processing*, Vol. 36, No. 7, pp. 1109-1118, 1988.
2. B. Widrow and R.G. Winter, "Neural Nets for Adaptive Filtering and Adaptive Pattern Recognition," *IEEE Computer*, pp. 25-39, March 1988.
3. B. Widrow and M.A. Lehr, "30 Years of Adaptive Neural Networks : Perceptron, Madaline, Backpropagation," *Proceedings of the IEEE*, Vol. 78, No. 9, pp. 1415-1442, 1990.
4. J.P. Eckmann and D. Ruelle, "Ergodic Theory of Chaos and Strange Attractors," *Reviews of Modern Physics*, Vol. 57, No. 3, pp. 617-656, 1985.
5. F. Takens, "Detecting Strange Attractors in Turbulence," *Lecture Notes in Mathematics 898*, Springer : Berlin, pp. 366-381, 1981.
6. R. Mane, "On the Dimension of the Compact Invariant Sets of Certain Nonlinear Maps," *Lecture Notes in Mathematics 898*, Springer : Berlin, pp. 230-242, 1981.
7. D.H. Nguyen and B. Widrow, "Neural Networks for Self Learning Control Systems," *IEEE Cont. Syst. Mag.*, Vol. 10, No. 3, pp. 18-23, Apr. 1990.
8. K.S. Narendra and K. Parthasaraty, "Identification and Control of Dynamic Systems Using Neural Networks," *IEEE Trans. Neural Network*, Vol. 1, No. 1, pp. 4-27, 1990.
9. D.E. Rumelhart, G.E. Hinton, and R.J. Williams, "Learning Internal Representation by Error Propagation," *Parallel Distributed Processing, Explorations on the Microstructure of Cognition*, Vol. 1, MIT Press, Cambridge : MA, pp. 318-362, 1986.
10. Wan-Sup Cheung, *Identification, Stabilisation and Control of Nonlinear systems using Neural Network-Based Nonlinear Modelling*, Ph. D. thesis, Signal Processing and Control Group, ISVR, University of Southampton, England, March 1993.
11. J.D. Farmer and J.J. Sidorowich, "Predicting Chaotic Time Series," *Physical Review Letters*, Vol. 59, No. 8, pp. 845-848, 1987.
12. T.S. Parker and L.O. Chua, "Chaos : A Tutorial for Engineers," *Proceedings of the IEEE*, Vol. 75, No. 8, pp. 982-1008, 1987.

F. W. Lauterborn and U. Parlitz, "Methods of Chaos Physics and Their Applications to Acoustics," *Journal of the Acoustical Society of America*, Vol. 84., No. 6, pp. 1975 1993, 1988.

14. J.K. Hammond, *Lecture Notes: Nonlinear Systems*, ISVR, University of Southampton, England, 1990.

15. W.H. Press, B.P. Flannery, S.A. Teukolsky, and W.T. Vettering, *Numerical Recipes: The Art of Scientific Computing*, Cambridge University Press, 1986.

▲Wan Sup Cheung

Wansup Cheung received the B.S.degree(1982) in nuclear engineering, the M.A.degree in mechanical engineering at Hanyang University (Seoul, Korea), and the M. Phil. and Ph.D.degree (1993) in Signal Processing and Control at ISVR (Southampton, England). Since 1984, he has been a senior research in the Acoustics and Vibration Group at Korea Research Institute of Standards and Science (KRISS), Taejon, Korea, In 1992, he joined the ISVR reserach staff for the development of adaptive prametric models for nonlinear systems and also two years in the project of optimal transmitter signal designing for underwater radar systems at ISVR. His reserch interests include digital signal processing in acoustics and mechanical vibration, active noise/vibration control, and self-adaptive parametric modelling for nonlinear dynamical systems.

▲Moon Jae Jho

Moonjae Jho has been a senior research in the Acoustics and Vibration Group at Korea Research Institute of Standards and Science (Taejon, Korea) since 1982. He received the M.S. degree and the Ph.D. degree in electronic engineering from Yonsei University in 1981 and 1990, respectively. He is on the Editorial Board of the *Journal of Acoustical Society of Korea*. His major research area include the topics related to ultrasonic transducers, microphones, and noise control.

▲Joseph K. Hammond

J. K. Hammond received the B.S. degree, the M. A. degree, and the M.Phil. and Ph.D. degree at Institute of Sound and Vibration Research (ISVR), Southampton, England. Since he had established the Signal Processing and Control Group in ISVR, he opened new application fields in digital signal processing and active control in acoustics, mechanical vibration, and underwater radar systems. In 1989, he was awarded as the Professor title in Signal Processing due to his achievements in such fields. Since 1992, he has been a director in ISVR. He is on the Editorial Board of the *Journal of Sound and Vibration and the Journal of Mechanical Systems and Signal Processing*. He has published over 100 papers in the area of signal processing and control in acoustics and vibration, optimal control of autonomous vehicle control, and multidimensional signal processing of underwater radar systems. In addition to his previous research fields, his recent interest includes parametric modelling and control of nonlinear systems.