Backpropagation Networks Modelling: Appropriate Structure and Convergence Speed

SONGYOT SUREERATTANAN[*], HUYNH NGOC PHIEN[**], NIDAPAN SUREERATTANAN[**] and NIKOS MASTORAKIS[***]
[*]Information Technology Group, The Bank of Thailand
273 Samsen Rd., Bangkhunporm, Pranakorn, Bangkok 10200 THAILAND
[**]Computer Science and Information Management Department, Asian Institute of Technology
P.O. Box 4, Klong Luang, Pathumthani 12120 THAILAND
***Military Institutes of University Education, Hellenic Naval Academy, Hatzikyriakou 18539, GREECE

Abstract: - The Bayesian Information Criterion (BIC) was presented to obtain the appropriate structure, via the number of hidden nodes, and a new algorithm was proposed to improve the convergence speed of backpropagation training method. The algorithm was obtained by employing the conjugate gradient method to solve the nonlinear part in the weights of the hidden layers and the Kalman filter to solve the linear part in the weights of the output layer. From simulation experiments with quarterly economic data on the exports and gross domestic product (GDP) in Thailand, it was found that the BIC and the algorithm could perform satisfactorily.
Key-words:- Backpropagation networks, Bayesian information criterion, Conjugate gradient method, Kalman filter

1 Introduction

In recent years, artificial neural networks (ANNs) have been extensively used in various fields. Among them, backpropagation (BP) networks appear to be most popular and have been widely used in many applications [1, 2]. The BP method, introduced by Rumelhart et al. [3], is *a supervised learning technique* for training multilayer feedforward neural networks. The gradient or steepest descent method is used to train a BP network by adjusting the weights in order to minimize the system error between the known output given by the user (desired output) and the output from the network (network output). To predict the future outcome values with an acceptable level of accuracy, the network has to be trained with a large sample of historical data that have been collected over a given time period.

BP networks have a number of shortcomings. One is how to select the appropriate structure of the network for a specific problem. As the number of nodes in the input and output layers are application dependent and a network with one hidden layer is sufficient in practical applications [4, 5], the remaining problem is how to choose the number of hidden nodes. Due to the use of the steepest descent method in the BP network, another shortcoming is its slow convergence (and no convergence in some cases).

In the paper, we present a method for selecting the appropriate number of hidden nodes based upon the Bayesian Information Criterion (BIC) and an algorithm for improving the convergence speed of BP networks. Since the nonlinear neural network problem can be partitioned into the nonlinear part in the weights of the hidden layers and the linear part in the weights of the output layer, the algorithm is obtained by employing the conjugate gradient method for the nonlinear part and the Kalman filter for the linear part. Experimental runs of our simulation studies were intended to assess the performance of these algorithms.

2 Proposed Method for Appropriate Structure

Once the number of nodes in the input and output layers have been decided, that normally depends upon the application under consideration, the important and difficult problem is how to optimally select the number of the hidden nodes and hidden layers. In addition, a network with one hidden layer is sufficient in practical applications [4, 5].

Since the appropriate number of hidden nodes is not known in advance, it is usually determined by *trial-and-error*. Hence, large amount of computation time is required. Generally, when the number of parameters (weights and biases) increases, the mean squared error (MSE) is expected to be reduced. Therefore, in order to compare several different models having different numbers of parameters, it is difficult to identify which network model is the best by using only the MSE [6].

Instead of the MSE, the Bayesian Information Criterion (BIC) can be utilized to select the best model from the candidate models having different number of

parameters. The BIC, developed independently by Kashyap [7] and Schwarz [8], can be expressed as follows:

$$\mathrm{BIC} = M \ln(\mathrm{MSE}) + P \ln(M) \quad (1)$$

where M is the number of training patterns and P is the number of parameters involved in the model.

It is noted that while the MSE is expected to progressively improve as more parameters are added to the model, the BIC penalizes the model for having more parameters (by a penalty term defined in the second part of Eq. 1) and therefore tends to result in a smaller model. The criterion can be used to assess the performance of the overall network, as it balances modelling error against network complexity. The proposed method is presented to systematically choose the appropriate number of hidden nodes using the procedure that gradually increases the network complexity and employs this criterion for terminating the network training as follows:

1. Create an initial network with one hidden node and randomize the weights.
2. Train the network using with a chosen method e.g. the proposed algorithm described in the next section (or the original BP algorithm) until the system error has reached an acceptable criterion. A simple stopping rule is introduced to indicate the convergence of the algorithm. It is based upon the relative error of the sum of squared errors (SE):

$$\left|\frac{\mathrm{SE}(t+1)-\mathrm{SE}(t)}{\mathrm{SE}(t)}\right| \le \varepsilon_1 \quad (2)$$

   where ε_1 is a constant that indicates the acceptable level of the algorithm and SE(t) denotes the value of SE at iteration t.
3. Check for terminating the selection of the network. A termination criterion is suggested based on the relative BIC as follows:

$$\left|\frac{\mathrm{BIC}(k+1)-\mathrm{BIC}(k)}{\mathrm{BIC}(k)}\right| \le \varepsilon_2 \quad (3)$$

   where ε_2 is a constant that indicates the acceptable level for the structure of the network and k denotes the number of hidden nodes of the network. If the relative BIC is less than or equal to ε_2 or the current BIC is greater than the previous, go to step 4; otherwise add a hidden node and randomize the weights then go to step 2.
4. Reject the current network model and replace it by the previous one, then terminate the training phase.

3 Proposed Algorithm for Improving the Convergence Speed

Recently, Scalero and Tepelenlioglu [9] proposed an improved method for training BP networks. This method is a modified form of the BP method along with a Kalman filter (KF) approach used to derive a training algorithm, which is an order of magnitude faster than the generalized delta rule. Even though this method overwhelmingly outperforms the BP method, it still uses error signals generated in the same way as in the BP method to estimate the desired output of the hidden layers.

By partitioning the nonlinear neural network problem into the nonlinear part in the weights of the hidden layers and the linear part in the weights of the output layer, a new algorithm is obtained by combining the conjugate gradient method and the KF algorithm. The conjugate gradient method and the KF algorithm are used for the nonlinear and linear parts, respectively.

The system error (overall patterns) between the desired output and the output from the network obtained just before being subject to the nonlinear activation function at the output layer is given as:

$$E = \frac{1}{2}\sum_{p=1}^{M}\sum_{k=1}^{N_L}\left(d_{pk} - y_{pLk}\right)^2 \quad (4)$$

where d_{pk} and y_{pLk} are the desired and network pre-image outputs for the kth node in the output layer L at the pth training pattern, respectively, M is the total number of training patterns and N_L is the number of nodes in the output layer. Substituting y_{pLk} by its expression, Eq. 4 becomes

$$E = \frac{1}{2}\sum_{p=1}^{M}\sum_{k=1}^{N_L}\left(d_{pk} - \sum_{i=0}^{N_{L-1}} w_{Lki} f\left(\sum_{l=0}^{N_{L-2}} w_{L-1,il} x_{p,L-2,l}\right)\right)^2 \quad (5)$$

where $x_{p,L-2,l}$ is the network output for the lth node in layer L-2 at the pth training pattern. It should be noted that from Eq. 5, the weights of the output layer are linear whereas the weights of the hidden layers are nonlinear.

To find the weights of the output layer w_{LKi}, we minimize the system error E with respect to the weights for node k in the output layer. So

$$\frac{\partial E}{\partial w_{Lki}} = 0$$

for i = 0 through N_{L-1}.
As

$$\frac{\partial E}{\partial w_{Lki}} = \frac{\partial E}{\partial y_{pLk}}\frac{\partial y_{pLk}}{\partial w_{Lki}},$$

we have

$$\frac{\partial E}{\partial w_{Lki}} = -\sum_{p=1}^{M}\left(d_{pk} - y_{pLk}\right)x_{p,L-1,i} = 0 \quad (6)$$

Equation 6 can be rewritten as

$$\sum_{p=1}^{M} d_{pk} x_{p,L-1,i} = \sum_{p=1}^{M} y_{pLk} x_{p,L-1,i}$$

Substituting the network pre-image output y_{pLk} at the output layer by its expression gives:

$$\sum_{p=1}^{M} d_{pk} x_{p,L-1,i} = \sum_{p=1}^{M} \sum_{r=0}^{N_{L-1}} w_{Lkr} x_{p,L-1,r} x_{p,L-1,i}$$

or

$$\sum_{p=1}^{M} d_{pk} x_{p,L-1,i} = \sum_{p=1}^{M} x_{p,L-1,i} \sum_{r=0}^{N_{L-1}} w_{Lkr} x_{p,L-1,r} \quad (7)$$

Changing the summation on the right-hand side to a vector in Eq. 7, we have

$$\sum_{p=1}^{M} d_{pk} x_{p,L-1,i} = \sum_{p=1}^{M} x_{p,L-1,i} \boldsymbol{x}_{p,L-1}^{T} \boldsymbol{w}_{Lk} \quad (8)$$

for i = 0 through N_{L-1}.

Defining

$$\boldsymbol{R} = \sum_{p=1}^{M} \boldsymbol{x}_{p,L-1} \boldsymbol{x}_{p,L-1}^{T} \quad (9)$$

and

$$\boldsymbol{p} = \sum_{p=1}^{M} d_{pk} \boldsymbol{x}_{p,L-1} \quad (10)$$

Then Eq. 8 becomes

$$\boldsymbol{p} = \boldsymbol{R} \boldsymbol{w}_{Lk} \quad (11)$$

or

$$\boldsymbol{w}_{Lk} = \boldsymbol{R}^{-1} \boldsymbol{p} \quad (12)$$

This results in the proposed algorithm, which can be summarized as follows:

1. Randomize all weights and biases as well as set the initial value to the inverse matrix $\boldsymbol{R}^{-1}$, where $\boldsymbol{R}$ is the correlation matrix of the network outputs in the last hidden layer.
2. For each training pattern pair ($\boldsymbol{x}_{p0}$, $\boldsymbol{o}_p$) where $\boldsymbol{x}_{p0}$ is the input vector and $\boldsymbol{o}_p$ is the desired output vector at the pth training pattern:

(a) Calculate the network pre-image output y_{pjk} and the network output x_{pjk} starting with layer j from 1 and proceeding layer by layer toward the output layer L for every node k. Usually, *the sigmoid function* is selected as an activation function:

$$y_{pjk} = \sum_{i=0}^{N_{j-1}} w_{jki} x_{p,j-1,i} \quad (13)$$

$$x_{pjk} = f(y_{pjk}) = \frac{1}{1 + \exp(-\rho y_{pjk})} \quad (14)$$

where N_j is the number of nodes in the jth layer and ρ is the sigmoid slope.

(b) Calculate the error signals for the weights at the output layer L and backtracking layer by layer from L-1 through 1:

$$e_{pLk} = f'(y_{pLk})(o_{pk} - x_{pLk}) = x_{pLk}(1 - x_{pLk})(o_{pk} - x_{pLk}) \quad (15)$$

$$e_{pjk} = f'(y_{pjk}) \sum_i e_{p,j+1,i} w_{j+1,i,k} = x_{pjk}(1 - x_{pjk}) \sum_i e_{p,j+1,i} w_{j+1,i,k} \quad (16)$$

(c) Calculate the gradient vector for each layer j from 1 through L-1:

$$\nabla E_p(\boldsymbol{w}_{jk}^t) = \frac{\partial E}{\partial \boldsymbol{w}_{jk}^t} = -e_{pjk} \boldsymbol{x}_{p,j-1} \quad (17)$$

where t denotes the present iteration number.

3. Calculate the gradient vector of all training patterns for each layer j from 1 through L-1:

$$\nabla E(\boldsymbol{w}_{jk}^t) = \sum_{p=1}^{M} \nabla E_p(\boldsymbol{w}_{jk}^t) \quad (18)$$

where M is the total number of training patterns.

4. Calculate the search direction for each layer j from 1 through L-1:

$$\boldsymbol{s}_{jk}^t = -\nabla E(\boldsymbol{w}_{jk}^t) \quad (19)$$

if t is the first iteration or an integral multiple of the dimension of $\boldsymbol{w}$; otherwise

$$\boldsymbol{s}_{jk}^t = -\nabla E(\boldsymbol{w}_{jk}^t) + \beta^t \boldsymbol{s}_{jk}^{t-1} \quad (20)$$

where β^t is computed according to a selected form such as that of Fletcher-Reeves [10].

5. Calculate the learning rate (step size) λ^t determined by an approximate line search to minimize the error function $E(\boldsymbol{w}^t + \lambda^t \boldsymbol{s}^t)$ along the search direction $\boldsymbol{s}^t$ at the tth iteration.
6. Update the weight vector for each hidden layer j from 1 through L-1:

$$\boldsymbol{w}_{jk}^{t+1} = \boldsymbol{w}_{jk}^t + \lambda^t \boldsymbol{s}_{jk}^t \quad (21)$$

7. For each training pattern:

(a) Calculate the network pre-image output y_{pjk} and the network output x_{pjk} starting with the layer j from 1 through the output layer L.

(b) Calculate the Kalman gain $\boldsymbol{k}_{pL}$ and update the inverse matrix $\boldsymbol{R}_{pL}^{-1}$ for the output layer L:

$$\boldsymbol{k}_{pL} = \frac{\boldsymbol{R}_{pL}^{-1} \boldsymbol{x}_{p,L-1}}{b_L + \boldsymbol{x}_{p,L-1}^T \boldsymbol{R}_{pL}^{-1} \boldsymbol{x}_{p,L-1}} \quad (22)$$

$$\boldsymbol{R}_{pL}^{-1} = \left[\boldsymbol{R}_{pL}^{-1} - \boldsymbol{k}_{pL} \boldsymbol{x}_{p,L-1}^T \boldsymbol{R}_{pL}^{-1}\right] b_L^{-1} \quad (23)$$

where b_L is *the forgetting factor* of the output layer.

(c) Calculate the desired pre-image output at the output layer:

$$d_{pk} = f^{-1}(o_{pk}) = \frac{1}{\rho} \ln\left(\frac{o_{pk}}{1 - o_{pk}}\right) \quad (24)$$

(d) Update the weight vector at the output layer L:

$$\boldsymbol{w}_{Lk}^{t+1} = \boldsymbol{w}_{Lk}^t + \boldsymbol{k}_{pL}(d_{pk} - y_{pLk}) \lambda_L \quad (25)$$

where λ_L is the learning rate of the output layer.

8. Repeat steps 2-7 until the system error has reached an acceptable criterion.

4 Experiments

4.1 Data Employed

Exports have become essential not only for the economic stability of a nation but also for its continuous growth, while gross domestic product (GDP) is an important indicator to measure the economic growth rate of each country. In this study, quarterly data on the exports and GDP of Thailand from 1987 to 1996 were used as shown in Fig. 1. The export and GDP data were obtained from the Bank of Thailand (BOT) and Thai Development Research Institute (TDRI), respectively. In each case, the data are divided into two parts: 1987 to 1993 for calibration (training) and 1994 to 1996 for validation (testing).

Figure 1 shows that the tendency of export and GDP data to increase over the calibration phase while it increases significantly in the validation phase. This results in good performance in the training phase, but worse in the testing phase. Therefore, trend removal for the residual series of export and GDP data is used. Before the data are presented to the network, the data after the trend removal are converted by a linear (affine) transformation to the interval [0.05, 0.95]. In this study, the input to the network may consist of the past values of exports (X) and GDP. The export value at time t+1 is treated as a function of past values of exports at times t, t-1, t-2, and t-3 and GDP at times t and t-1. The GDP at time t+1 is also treated as a function of past values of GDP at times t, t-1, t-2, and t-3 and exports at times t and t-1. The forecasting equations with lead time of one quarter of exports and GDP are given as follows:

$$X(t+1) = g(X(t), X(t-1), X(t-2), X(t-3), GDP(t), GDP(t-1))$$
$$GDP(t+1) = g(GDP(t), GDP(t-1), GDP(t-2), GDP(t-3), X(t), X(t-1))$$

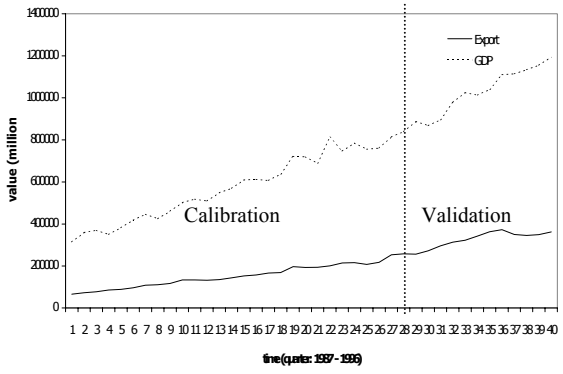

Fig. 1 Quarterly data on exports and GDP of Thailand

4.2 Experimental Conditions

In order to compare the performance of various algorithms, the same initial weights were used. During the training phase, the learning rate and temperature learning rate were set to 0.01, the momentum and temperature momentum coefficients were set to 0.5. The forgetting factor of 0.99 was found suitable. The temperature of each neuron was set randomly to lie within the range [0.9, 1.1] and the sigmoid slope was set to 1. The method described in Section 2 is used to determine the number of the hidden nodes. The values adopted for ε_1 and ε_2 were 0.0001 and 0.01, respectively. To find the optimal step size of the conjugate gradient method, the approximate line search method with backtracking by quadratic and cubic interpolations of Dennis and Schnabel [11] was used. To calculate the search direction, the formula of Fletcher-Reeves [10] was employed based on its good performance obtained in preliminary experiments.

4.3 Performance Criterion

In order to evaluate the performance of the network model, the efficiency index (EI) introduced by Nash and Sutcliffee [12] was used:

$$EI = SR / ST \tag{26}$$

$$SR = ST - SE \tag{27}$$

$$ST = \sum_{i=1}^{M} \left(y_i - \bar{y} \right)^2 \tag{28}$$

$$SE = \sum_{i=1}^{M} \left(y_i - \hat{y}_i \right)^2 \tag{29}$$

$$\bar{y} = (1/M) \sum_{i=1}^{M} y_i \tag{30}$$

where SR = Variation explained by the model,
ST = Total variation,
SE = Total sum of squared errors,
y_i = Desired output at time i,
$\bar{y}$ = Mean value of the desired output,
$\hat{y}_i$ = Network output at time i,
M = Number of training patterns.

4.4 Results

Firstly, we trained the network by using the BIC algorithm as described in Section 2. For the BIC method, the 6-1-1 network is initially selected. As the algorithm is terminated with the structure 6-3-1 network for exports and the structure 6-2-1 for GDP, the 6-2-1 network for exports and the 6-1-1 network for GDP are the best, as illustrated in Table 1.

Earlier, Sureerattanan and Phien [13] proposed an algorithm (referred to as Algorithm 1) to improve the convergence speed of BP networks by applying the adaptive neural model with the temperature momentum term to the KF algorithm with the momentum term. After the appropriate structure of the network is obtained, we compare the BP, KF, conjugate gradient, Algorithm 1 and the proposed algorithm (referred to as Algorithm 2). Figures 2-3 show the learning curve between these system errors and the iteration numbers for the algorithms during training of exports and GDP, respectively. The results confirm that Algorithms 1 and 2 converge very fast with a small value for system error. The calculated results of the efficiency index of each algorithm are provided in Tables 2 and 3 for training and testing phases, respectively. The total computation time of the algorithms for their convergence (on a PC Pentium Pro 180 MHz) is given in Table 4. It is clear that Algorithm 2 required the least computation time for convergence. The observed and forecast values of Algorithm 2 in the calibration and validation phases are shown in Figs. 4 and 5, respectively.

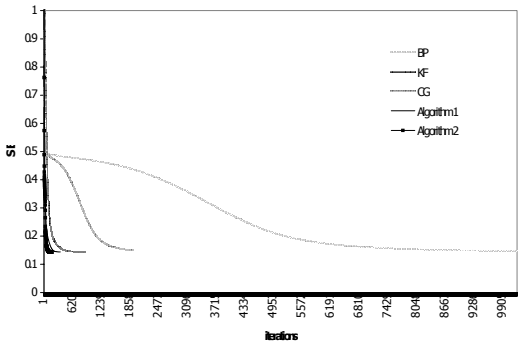

Fig. 2 Learning curve of exports for BP, KF, Conjugate gradient, Algorithms 1 and 2

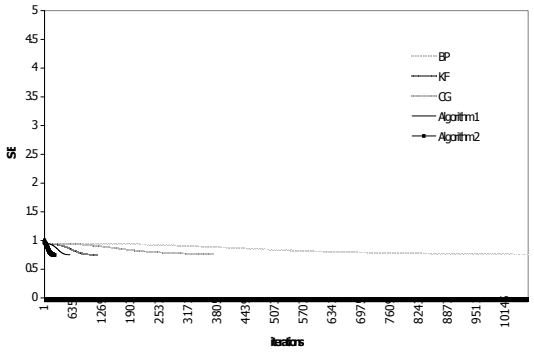

Fig. 3 Learning curve of GDP for BP, KF, Conjugate gradient, Algorithms 1 and 2

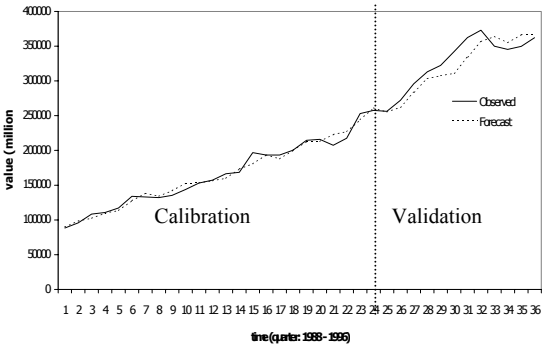

Fig. 4 Comparison between observed and forecast values for exports (1988-1996)

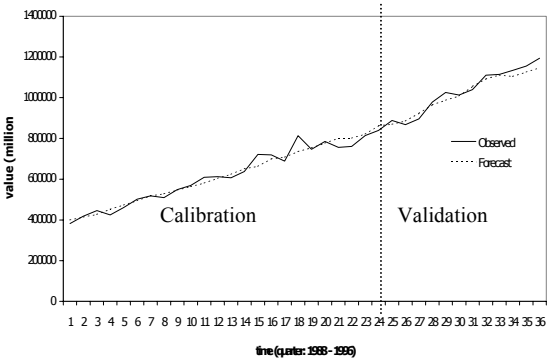

Fig. 5 Comparison between observed and forecast values for GDP (1988-1996)

Table 1 Computed values of efficiency index (EI) and BIC

Exports and GDP	6-1-1		6-2-1		6-3-1	
	EI	BIC	EI	BIC	EI	BIC
Exports	0.93	-65.44	0.98	**-68.99**	0.98	-43.74
GDP	0.96	**-54.62**	0.96	-29.30		

Table 2 Comparison between BP, KF, Conjugate gradient, Algorithms 1 and 2 in calibration phase

Exports and GDP	BP			KF			Conjugate gradient			Algorithm 1			Algorithm 2		
	SE	EI	Epoch	SE	EI	Epoch	SE	EI	Epoch	SE	EI	Epoch	SE	EI	Epoch
Exports	0.15	0.98	10,215	0.14	0.98	890	0.15	0.98	1,928	0.14	0.98	354	0.14	0.98	169
GDP	0.76	0.96	1,0637	0.75	0.96	1,166	0.76	0.96	3,712	0.75	0.96	561	0.75	0.96	222

Table 3 Efficiency indices of BP, KF, conjugate gradient, and Algorithms 1 and 2 in validation phase

Exports and GDP	BP	KF	Conjugate gradient	Algorithm 1	Algorithm 2
Exports	0.77	0.77	0.76	0.77	0.79
GDP	0.92	0.94	0.92	0.94	0.94

Table 4 Total computation time (in seconds) of BP, KF, Conjugate gradient, Algorithms 1 and 2

Exports and GDP	BP	KF	Conjugate gradient	Algorithm 1	Algorithm 2
Exports	57	19	12	8	2
GDP	59	23	20	11	2

5 Summary

We presented the Bayesian Information Criterion (BIC) for choosing the appropriate number of hidden nodes and algorithm for speeding up the convergence of BP networks. The algorithm was devised by employing the conjugate gradient method to solve the nonlinear part and the Kalman filter algorithm to solve the linear part. The experimental results show that the BIC can be employed to determine the appropriate number of the hidden nodes, and the proposed algorithm can improve the convergence speed of the BP networks and also gives good performance in testing phase.

References:

[1] Gershenfield, N.A. and Weigend, A.S., The Future of Time Series, *Technical Report, Palo Alto Research Center*, 1993.

[2] Phien, H.N. and Siang, J.J., Forecasting Monthly Flows of the Mekong River using Back Propagation, *Proc. IASTED Int. Conf.*, 1993, pp.17-20.

[3] Rumelhart, D.E., Hinton, G.E., and Williams, R.J., Learning Internal Representations by Error Propagation, in *Parallel Distributed Processing: Explorations in the Microstructure of Cognition: Vol.1: Foundations*, Rumelhert, D.E. and McClelland, J.L. (Eds.), MIT Press, Cambridge, Massachusetts, 1986, pp.318-362.

[4] Hornik, K., Stinchcombe, M., and White, H., Multilayer Feedforward Networks are Universal Approximators, *Neural Networks*, Vol.2, 1989, pp.359-366.

[5] Patterson, D.W., *Artificial Neural Networks: Theory and Applications*, Prentice Hall, Singapore, 1996.

[6] Brown, M. and Harris, C., *Neurofuzzy Adaptive Modelling and Control*, Prentice Hall, UK., 1994, pp.326-327.

[7] Kashyap, R.L., A Bayesian Comparison of Different Classes of Dynamic Models using Empirical Data, *IEEE Trans. Automatic Control*, Vol.AC-22, No.5, 1977, pp.715-727.

[8] Schwarz, G., Estimating the Dimension of a Model, *The Annals of Statistics*, Vol.6, No.2, 1978, pp.461-464.

[9] Scalero, R.S. and Tepedelenlioglu, N., A Fast New Algorithm for Training Feedforward Neural Networks, *IEEE Trans. Signal Processing*, Vol.40, No.1, 1992, pp.202-210.

[10] Fletcher, R. and Reeves, C.M., Function Minimization by Conjugate Gradients, *Computer J.*, Vol.7, 1964, pp.149-154.

[11] Dennis, J.E. and Schnabel, R.B., *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*, Prentice-Hall, Inc., New Jersey, USA, 1983.

[12] Nash, J.E. and Sutcliffee, J.V., River Flow Forecasting through Conceptual Models, *J. Hydrology*, Vol.10, 1970, pp.282-290.

[13] Sureerattanan, S. and Phien, H.N., Speeding up the Convergence of Backpropagation Networks, *Proc. IEEE Asia-Pacific Conference on Circuits and Systems: Microelectronics and Integration Systems*, Chiangmai, Thailand, 1998, pp.651-654.