Convergence Analysis of Adaptive Recurrent Neural Network

Hong Li*, Ali Setoodehnia**
*(Department of Computer Systems Technology, New York City College of Technology, New York, USA)
** (Department of Electronics Engineering Technology, ECPI University, USA)

ABSTRACT
This paper presents analysis of a modified Feed Forward Multilayer Perceptron (FMP) by inserting an ARMA (Auto Regressive Moving Average) model at each neuron (processor node) with the Backpropagation learning algorithm. The stability analysis is presented to establish the convergence theory of the Backpropagation algorithm based on the Lyapunov function. Furthermore, the analysis extends the Back propagation learning rule by introducing the adaptive learning factors. A range of possible learning factors is derived from the stability analysis. Performance of such network learning with adaptive learning factors is presented and demonstrates that the adaptive learning factor enhance the performance of training while avoiding oscillation phenomenon.

Keywords—Adaptive learning, Back propagation, Neural networks, Stability analysis

I. INTRODUCTION
In the last few decades, Artificial Neural Networks (ANNs) have been successfully applied to many real world applicationsand known to be useful in various tasks of modeling nonlinear systems, such as signal processing, pattern recognition, optimization, weather forecasting, to name a few. ANN is a set of processing elements (neurons or perceptrons) with a specific topology of weighted interconnections between these elements and a learning law for updating the weights of interconnection between two neurons. To respond to the increased demand of system identification and forecasting with large set of data, many different ANNs structures and learning rules, supervised, or unsupervised, have been proposed to meet various needs as robustness and stability. The FMP networks have been shown to obtain successful results in system identification and control [1, 2]. The Lyapunov function approach was used to obtain stability analysis of Backpropagation training algorithm of such network in [3]. The major drawback of the FMP is that it requires large number of input datafor training to achieve sufficient performance. Recurrent neural networks have been shown successful in identification of time varying systems along with the stability analysis in [4, 5]. However, the training process can be very sensitive to initial conditions such as number of neurons, the number of layers, and value of weights, and learning factors which are often chosen by trial and error. This paper presents a modified FMP architecture which inserts a dynamic filtering capability, ARMA local feedback at each neuron in the FMP structure. The Backpropagation algorithm is used for learning – weight adjusting. Stability analysis will be derived using Lyapunov function. It turns out that the learning factor must be within a range of values in order to guarantee the convergence of the algorithm. In the simulation, instead of selecting a learning factor by trial and error, authors define an adaptive learning factor which satisfies the convergence condition and adjust connection weight accordingly. The simulation results are presented to demonstrate the performance.

II. BASIC PRINCIPLE OF ARMA-LFMP NETWORK
An identification problem can be outlined as it is shown in Figure 1. A set of data is collected from the PLANT: input data and corresponding output data observed, or measured as target output of the identification problem. The set is often called “training set”. A neural network model with parameters, called weights, is designed to simulate the PLANT. When the output from the neural network is calculated, an error between the network output and the target output is generated. The learning process of neural network is to modify the network to minimize the error. Consider a system with \( n \) inputs \( X = \{ X_1, \ldots, X_n \} \) and \( m \) output units \( Y = \{ Y_1, \ldots, Y_m \} \). In a typical neuron, called perceptron, the output \( Y \) is expressed as:

\[
Y = F(Z) = \frac{1-e^{-Z}}{1+e^{-Z}} = \sum_{i=1}^{m} W_i \tag{1}
\]

Fig.1 Outline of ANN identification problem
where \(w_i\) is called connection weight from input \(i\); \(F\) is a nonlinear sigmoid function with a constant number \(\theta\), call it slope.

A FMP network combines number of neurons, called nodes, feed forward to next layer of nodes, illustrated in Figure 2. Suppose \(l\) is the number of nodes in the \(lth\) layer, each output from the \((l-1)th\) layer will be used as input for the next layer, that can be expressed as:

\[
X_j^l = F(Z_j); Z_j = \sum_{i=1}^{N_{l-1}} W_{li} X_i^{l-1}; \quad j = 1, 2, ..., N_l
\]

, where \(l = 1, ..., L\) is the number of layers in the network and \(W\) is the connection weight from the \(ith\) node in \(l-1\) layer to the \(jth\) node in \(l\) layer.

In this structure in Figure 3, an ARMA model is inserted at each node in the Local feedback FMP network. The outputs from ARMA model are used as inputs to the FMP neural network node. The output at the \(j\)-th node in the \(l\)-th layer with an ARMA model is expressed as:

\[
X_j^l(p) = F(Z_j^l(p))
\]

and

\[
Z_j^l(p) = \sum_{i=1}^{N_{l-1}} X_i^l(p) + \sum_{d=1}^\infty \nu_j^d X_i^{l-d}(p - d) \]

where \(X_i^l(p)\) is the output of nonlinear sigmoid function for the hidden layer and output layer, and is also used as an input to ARMA model; \(\nu\) represents the output of ARMA model; \(a\) and \(b\) are connection weights of the ARMA, \(\nu\) is weight of local feedback at each node and \(DA\) and \(DB\) are number of delays for AR and MA processes respectively.

The back-propagation algorithm has become the standard algorithm used for training feed-forward multi-layer perceptrons. It is a generalized Least Mean Square algorithm that minimizes the mean squared error between the target output and the network output with respect to the weights. The algorithm looks for the minimum of the error function in the weight space using the method of gradient descent. The combination of weights which minimizes the error function is considered to be a solution of the learning problem. A proof of the Backpropagation algorithm was presented in [6] based on a graphical approach in which the algorithm reduces to a graph labeling problem.

The total error \(E\) of the network over all training set is defined as:

\[
E = \frac{1}{T} \sum_{k=1}^{T} \sum_{p=1}^{N_k} e_k^2(p)
\]

where \(e_k^2(p)\) is the error associated with \(p\)th pattern at the \(k\)th node of output layer.

\[
d_k^2(p) = (d_k(p) - Y_k^l(p))^2
\]

where \(d_k(p)\) is the target at \(k\)th node and \(Y_k^l\) is the output of network at the \(k\)th node.

The network connection weights \(a_{ij}, b_{ij}, \) and \(\nu\) between neurons \(i\) in layer \(l-1\) and neuron \(j\) in Layer \(l\) (\(l = 1, ..., L\)) are updated iteratively by the Gradient Descent Rule

\[
\Delta a_{ij} = -\mu \frac{\partial E}{\partial a_{ij}}
\]

\[
\Delta b_{ij} = -\mu \frac{\partial E}{\partial b_{ij}}
\]

\[
\Delta \nu_{ij} = -\mu \frac{\partial E}{\partial \nu_{ij}}
\]

where \(\mu\) is the learning factor. Substituting (7) into (8), (9) and (10), the above updating equation can be expressed as follow:

\[
\Delta a_{ij} = -\frac{2\mu}{T} \sum_{k=1}^{N_k} \sum_{p=1}^{T} e_k^2(p) \frac{\partial X_k^l(p)}{\partial a_{ij}}
\]

\[
\Delta b_{ij} = -\frac{2\mu}{T} \sum_{k=1}^{N_k} \sum_{p=1}^{T} e_k^2(p) \frac{\partial X_k^l(p)}{\partial b_{ij}}
\]

\[
\Delta \nu_{ij} = -\frac{2\mu}{T} \sum_{k=1}^{N_k} \sum_{p=1}^{T} e_k^2(p) \frac{\partial X_k^l(p)}{\partial \nu_{ij}}
\]

The rate of change of an output from \(k\)-th node of \(l\)-th layer with respect to connection weights \(a\), \(b\), and \(\nu\) in \(l\)th layer can be expressed as:
Further calculation leads to the expressions of \( \frac{\partial x}{\partial t} \), and \( \frac{\partial x}{\partial v} \), rate of change of an output from \( k \)-th node of \( l \)-th layer with respect to connection weights \( a, b, \) and \( \nu \) in layer \( l \) for \( n < l \):

\[
\frac{\partial x_k^l(p)}{\partial v_{lj}^l} = F'(Z_j^l(p)) \left[ \sum_{r=1}^{L-1} \frac{\partial x_r^{l-1}(p)}{\partial a_{ij}^l} + \frac{\partial x_r^{l-1}(p)}{\partial b_{ij}^l} \right] \frac{\partial x_{lj}^l(p)}{\partial v_{lj}^l}
\]

where

\[
\frac{\partial x_k^l(p)}{\partial a_{lj}^l} = \sum_{i=1}^{n} \frac{\partial x_i^l(p)}{\partial a_{ij}^l} + \frac{\partial x_i^l(p)}{\partial b_{ij}^l}
\]

Further calculation leads to the expressions of \( \frac{\partial x}{\partial t} \), and is the derivative of an output from \( k \)-th node of \( l \)-th layer with respect to connection weights \( a, b, \) and \( \nu \) in layer \( l \) for \( n < l \):

\[
\frac{\partial x_k^l(p)}{\partial v_{lj}^l} = F'(Z_j^l(p)) \left[ \sum_{r=1}^{L-1} \frac{\partial x_r^{l-1}(p)}{\partial a_{ij}^l} + \frac{\partial x_r^{l-1}(p)}{\partial b_{ij}^l} \right] \frac{\partial x_{lj}^l(p)}{\partial v_{lj}^l}
\]

(11)

III. STABILITY ANALYSIS

Stability for nonlinear systems refers to the stability of a particular solution. There may be one solution which is stable and another which is not stable. There are no inclusive general concepts of stability for nonlinear systems. The behavior of a system may depend drastically on the inputs and the disturbances. However, Lyapunov developed a theory to examine the stability of nonlinear systems.

The definition of Lyapunov function and Lyapunov theorem are quoted below [7]:

**Definition 1 (Lyapunov function):** A scalar function \( V(x) \) is a Lyapunov function for the system \( x(t+1) = f(x(t)) \) if the following conditions hold:

1. \( V(0) = 0 \) and \( V(x) \) is continuous in \( x \)
2. \( V(x) \) is positive definite, that is, \( V(x) \geq 0 \) with \( V(x) = 0 \) only if \( x = 0 \)
3. \( \Delta V(x) = V(f(x(t)) - V(x(t)) \) is negative definite, that is, \( V(f(x(t)) - V(x(t)) \leq 0 \) with \( \Delta V(x) = 0 \) only if \( x = 0 \)

**Theorem 1 (Lyapunov Theorem):** The solution for the system given by (11) is asymptotically stable if there exists a Lyapunov function in \( x \).

The stability of the learning process in an identification approach leads to a better modeling and a guaranteed reached performance. According to Lyapunov theorem, the determination of stability depends on the selection and verification of a positive definite function. For the systems defined in (3) – (5), assume that the Backpropagation learning rule is applied and the error function and weights updating rule are defined in (6) – (10), then define

\[
V(t) = \sum_{j=1}^{N_l} \sum_{p=1}^{T} \epsilon_j^p(t)
\]

(23)

The proof is given in the following theorem that \( V(t) \) satisfies the Lyapunov condition.

**Theorem 2:** Assume that the nonlinear function \( F(\cdot) \) is continuous and differentiable, the ARMA-LFMP is defined in (3)-(5), rewrite the learning rule (8) - (10) in weights vector form as:

\[
\Delta A^l = -\frac{2\mu}{N_l I} \sum_{k=1}^{N_l} \sum_{s=1}^{T} \epsilon_k^s(p) \frac{\partial x_k^s(p)}{\partial A^l}
\]

(24)
where \( A_l \) and \( \nu_l \) are weight vectors in \( l \)th layer, then the system is stable under the condition:

\[
\mu < \frac{2N_l}{N_L} \sum_{k=1}^{N_l} \left[ \left\| \frac{e_{p_j} e_{p_j}}{\partial A_l} \right\|^2 + \left\| \frac{\theta_{p_j} \theta_{p_j}}{\partial \nu_l} \right\|^2 \right] \quad (27)
\]

Proof: Assume that the Lyapunov function is defined in (23), calculation of \( \Delta \) leads to:

\[
\Delta V(t) = V(t + 1) - V(t)
\]

\[
= \frac{1}{N_l T} \sum_{j=1}^{N_l} \sum_{p=1}^{T} \left[ e_{p_j}(t + 1) - e_{p_j}(t) \right]^2
\]

Apply the first order Taylor expansion of with respect to weight vectors

\[
\Delta e_{p_j}(t) = \sum_{l=1}^{L} e_{p_j}(t) \Delta A_l^t + \left\| \frac{e_{p_j} e_{p_j}}{\partial A_l} \right\| \Delta^t + \left\| \frac{\theta_{p_j} \theta_{p_j}}{\partial \nu_l} \right\| \Delta^t
\]

Substitute (24), (25), and (26) into (29),

\[
\Delta e_{p_j}(t) = e_{p_j}(t) \Delta A_l^t + \left\| \frac{e_{p_j} e_{p_j}}{\partial A_l} \right\| \Delta^t + \left\| \frac{\theta_{p_j} \theta_{p_j}}{\partial \nu_l} \right\| \Delta^t
\]

Then, substitute the (30) into (28),

\[
\Delta V(t) = V(t) - V(t + 1)
\]

For simplicity, let

\[
\alpha = \sum_{j=1}^{N_l} \left[ \left\| \frac{e_{p_j} e_{p_j}}{\partial A_l} \right\|^2 + \left\| \frac{\theta_{p_j} \theta_{p_j}}{\partial \nu_l} \right\|^2 \right] \quad (33)
\]

Then apply it into (32) and consider that

\[
\left\| \frac{e_{p_j} e_{p_j}}{\partial A_l} \right\|, \left\| \frac{\theta_{p_j} \theta_{p_j}}{\partial \nu_l} \right\|
\]

then

\[
\Delta V(t) \leq \frac{2N_l}{N_L} \sum_{k=1}^{N_l} \left[ \left\| \frac{e_{p_j} e_{p_j}}{\partial A_l} \right\|^2 + \left\| \frac{\theta_{p_j} \theta_{p_j}}{\partial \nu_l} \right\|^2 \right]
\]

\[
= \frac{2N_l}{N_L} \sum_{k=1}^{N_l} \left[ \left\| \frac{e_{p_j} e_{p_j}}{\partial A_l} \right\|^2 + \left\| \frac{\theta_{p_j} \theta_{p_j}}{\partial \nu_l} \right\|^2 \right]
\]

To ensure that the function \( V \) satisfies the Lyapunov condition, let the right-hand side of (34) be less than zero, and consider that

\[
\left\| \frac{e_{p_j} e_{p_j}}{\partial A_l} \right\|, \left\| \frac{\theta_{p_j} \theta_{p_j}}{\partial \nu_l} \right\|
\]

then we obtain the condition

\[
\mu < \frac{2N_l}{N_L} \sum_{k=1}^{N_l} \left[ \left\| \frac{e_{p_j} e_{p_j}}{\partial A_l} \right\|^2 + \left\| \frac{\theta_{p_j} \theta_{p_j}}{\partial \nu_l} \right\|^2 \right]
\]

Therefore, the ARMA-LFMP system defined in (3)–(5) is stable when the learning factor in Backpropagation learning rule described in (8)–(10) satisfies the condition (35).

For purpose of simplifying the simulation, instead of calculating all \( \xi \), \( \nu \), and \( \nu \) for \( l = 1, \ldots, L; j = 1, \ldots, \), the following corollary will identify an upper bound of

\[
\sum_{j=1}^{N_L} \left[ \left\| \frac{e_{p_j} e_{p_j}}{\partial A_l} \right\|^2 + \left\| \frac{\theta_{p_j} \theta_{p_j}}{\partial \nu_l} \right\|^2 \right] \quad (36)
\]

First, for the output layer \( L \), apply infinite norm in (11) and the notation \( |y|^\infty = \sum_{d=1}^{N_L} \), calculation leads to

\[
\left\| \frac{\partial y}{\partial \nu_l} \right\| \leq \frac{\theta}{2} \left[ 1 + |y|^\infty \right] \quad (36)
\]

From (5),

\[
\left\| x_{p,j} \right\| \leq \sum_{d=1}^{N_L} a_{p,j} \left\| x_{p,j} \right\| + \sum_{d=1}^{N_L} b_{p,j}
\]
Apply (47), (48) and (49) into condition (35), we obtain a more restrictive condition as follows:

$$\mu \leq \frac{N_{L}}{6\sum_{k=1}^{N_{L}} \beta_{k}^{\gamma_{k+1}} (2+\beta_{k}^{\gamma_{k+1}}|y(k)|)}$$

(50)

Corollary 1: The ARMA-LFMP system converges if the following conditions are satisfied:

$$2-\theta |y| > 0, \quad (1-|\hat{A}_{ij}|) > 0,$$

(51)

and

$$\mu \leq \frac{N_{L}}{6\sum_{k=1}^{N_{L}} \beta_{k}^{\gamma_{k+1}} (2+\beta_{k}^{\gamma_{k+1}}|y(k)|)}$$

(52)

IV. SIMULATION

In this section, an example of chaotic system known as Henon system is considered to demonstrate the effectiveness of developed methods in this paper. The Hénon map is a discrete-time dynamical system described as:

$$X(k+1) = 1 - a X^{2}(k) + Y(k)$$

$$Y(k+1) = bX(k)$$

It is one of the most studied examples of dynamical systems that exhibit chaotic behavior. The Hénon map takes a point \((x(k), y(k))\) in the plane and maps it to a new point. The map depends on two parameters, \(a\) and \(b\), which for the classical Hénon map have values of \(a = 1.4\) and \(b = 0.3\). For the classical values the Hénon map is chaotic.

In this simulation, consider the system and a three-layer neural network structure was selected for two inputs and two output with number of nodes as 5, 5 and 2 in layer 1, 2 and 3 respectively. 100 patterns of data were generated and used for learning. After number of trial and error attempts, with slope set as 0.6 and learning factor set as constant .01, and random generated initial weights, the system reached to absolute error 0.0899999 after 2177311 iterations.

With adaptive learning factor, it took 373317 number of iteration to reach the same threshold of 0.0899999. It is also observed that the error decreases steadily while the adaptive learning factor is applied and the oscillation of error was observed while a predefined constant learning factor is applied.

The Figure 4 demonstrated 100 patterns of data generated from Henon system comparing with simulated data from the neural network described above.

The adaptive learning factor guarantees that the errors will steadily decrease. The drawback is that it increased calculation since an updated learning factor needs to be calculated at every weights update based on Backpropagation algorithm. Applying the constant learning factor avoids the calculation, but a proper constant learning factor has to be identified through
trial and error. In the Figure 5, the plot demonstrated comparison of errors from learning process with constant learning factor and the adaptive learning factor. The constant learning factor was selected with value of 0.1. To compare the effectiveness of adaptive learning method, the 0.1 was used as initial learning factor in the adaptive method. Points for the plot were taken from the errors of every 1000th of iteration.

Figure 5. Comparison of errors from learning with adaptive and constant learning factor

REFERENCES


Switching, Journal of Inequalities and Applications 2010


