International Journal of Electrical, Electronic and Communication Sciences
ISSN: 2517-9438
Vol:1, No:12, 2007

Optimization of a Three-Term Backpropagation Algorithm Used for Neural Network Learning

Yahya H. Zweiri

Abstract**—** The back-propagation algorithm calculates the weight changes of an artificial neural network, and a two-term algorithm with a dynamically optimal learning rate and a momentum factor is commonly used. Recently the addition of an extra term, called a proportional factor (PF), to the two-term BP algorithm was proposed. The third term increases the speed of the BP algorithm. However, the PF term also reduces the convergence of the BP algorithm, and optimization approaches for evaluating the learning parameters are required to facilitate the application of the three terms BP algorithm. This paper considers the optimization of the new back-propagation algorithm by using derivative information. A family of approaches exploiting the derivatives with respect to the learning rate, momentum factor and proportional factor is presented. These autonomously compute the derivatives in the weight space, by using information gathered from the forward and backward procedures. The three-term BP algorithm and the optimization approaches are evaluated using the benchmark XOR problem.

Keywords**—** Neural Networks, Backpropagation, Optimization.

I. Introduction

BACKPROPAGATION(BP) algorithm is used for training artificial neural networks [2]. Training is usually carried out by iterative updating of weights based on the error signal. The negative gradient of a mean-squared error function is commonly used. In the output layer, the error signal is the difference between the desired and actual output values, multiplied by the slope of a sigmoidal activation function. Then the error signal is back-propagated to the lower layers. BP is a descent algorithm, which attempts to minimize the error at each iteration. The weights of the network are adjusted by the algorithm such that the error is decreased along a descent direction. Traditionally, two parameters, called learning rate (LR) and momentum factor (MF), are used for controlling the weight adjustment along the descent direction and for dampening oscillations. The BP algorithm is used for many applications. However, its convergence rate is relatively slow, especially for networks with more than one hidden layer. The reason for this is the saturation behaviour of the activation function used for the hidden and output layers. Since the output of a unit exists in the saturation area, the corresponding descent gradient takes a very small value, even if the output error is large, leading to very little progress in the weight adjustment. The selection of the LR and MF is arbitrary, because the error surface usually consists of many flat and steep regions and behaves differently from application to application. Large values of the LR and MF are helpful to accelerate learning. However, this increases the possibility of the weight search jumping over steep regions and moving out of the desired regions.

Manuscript received July 17, 2006.

Y.H. Zweiri is with Department of Mechanical Engineering, Mu'tah University, Karak, Jordan. Tel: (+00962) 32372594, EX:3022, E-mail: (yahya.zweiri@kcl.ac.uk) or (yhzweiri@mutah.edu.jo)

The problem of improving the efficiency and convergence rate of the back-propagation algorithm has been investigated by a number of researchers. For example, [3] does not use higher-order derivatives but determines individual learning rates for each component of the weights vector separately. Three new parameters are required and like the conventional BP, convergence rates are slow. In addition, a large number of trial runs are required before arriving at the right parameter. [4] has proposed a different cost function, whilst [5] proposed variables are re-scaled dynamically. [6], [7] considered dynamic learning rate and momentum factor optimization using derivative information. [8] used a genetic algorithm for self-adaptation to accelerate the steepest descent rate, by slightly modifying the learning rate of the previous step. [9] presented a new incremental learning method for pattern recognition, employing bounded weight modification and structural adaptation learning rules and applies initial knowledge to constrain the learning process. [10] investigated the behaviour of the BP algorithm with a small constant learning rate with stationary, random input environments. The sequence of weight estimates is approximated by ordinary differential equations, in the sense of weak convergence of a random processes as a small number (learning rate) tends to zero. [11] proved the companion Rosenblatt's perceptron convergence (PC) theorem for feed-forward networks, stating that the BP algorithm converges to an optimal solution for linearly separable patterns. [12] presented a constraint to be satisfied in addition to the demand for minimization of the cost function, and used Lagrangian multipliers in order to improve convergence. [13] gives a detailed analysis of the delta rule algorithm, indicating why one implementation leads to a stable numerical process. In [14], the necessary and sufficient conditions for the stability behaviour of the three-term backpropagation algorithm are established.

This paper presents efficient BP learning using simultaneously optimized Learning Rate (LR), Momentum Factor (MF) and Proportional Factor (PF) terms. A set of recursive formulae is used for calculating the derivatives of the optimization target with respect to LR, MF and PF. This behaves as a feed-forward procedure in the BP algorithm and does not increase the computational complexity. A group of approaches exploiting the derivatives with respect to LR, MF and PF are presented. The approaches are applied to an example problem and shown that the convergence rate is significantly improved compared to the plain three-term BP algorithms.

The paper is organized as follows. First, the back-propagation algorithm is given. Then, the proportional factor term is proposed. Three approaches are presented to estimate optimal values for the LR, MF and PF terms. Finally, conclusions are drawn.

II. Background

The back-propagation algorithm for multi-layer neural networks is a gradient descent procedure used to minimize a least-square objective function (error function). Assume a batch of training sample pairs: $(I_1, T_1), \cdots, (I_n, T_n)$, where I_s, $1 \leq s \leq n$, represent the sth input in the batch, and $T_s, 1 \leq s \leq n$, is the corresponding desired output (target). For arbitrary hidden layers neurons, the least-square objective function in the networks weight space is

$$E = \frac{1}{nZ_M} \sum_{s=1}^{n} [T_s - O_s^M]^{\mathcal{T}} [T_s - O_s^M], \tag{1}$$

where O_s^M is the output vector of an M-layered network with I_s as input, and Z_M is the number of output neurons.

Let W be a vector formed by all the network weights and $\nabla E(W(k))$ be the gradient of E at $W = W(k)$, with $k = 1, 2, 3, \cdots, N$, being the iteration number of the weights vector. The conventional back-propagation algorithm with a momentum term can be simply described as

$$\Delta W(k) = \alpha(-\nabla E(W(k))) + \beta \Delta W(k-1), \tag{2}$$

where α and β are the learning rate (LR) and momentum factor (MF) respectively, and $\Delta W(k) = W(k+1) - W(k)$. The feed-forward computations of the network with I_s presented to the input layer are given by

$$o_{s,i}^m = f([W_i^m(k+1)]^{\mathcal{T}} O_s^{m-1}), \tag{3}$$

where $o_{s,i}^m$, $1 \leq i \leq Z_m$, denotes the ith output of layer m, $1 \leq m \leq M$; $f(\cdot)$ is the activation function (usually chosen as the logistic or tanh functions); $W_i^m(k+1)$ is a sub-vector of $W(k+1)$, which consists of all the weights from the neurons of layer $m-1$ to $o_{s,i}^m$; and $O_{s,i}^{m-1}$ is a vector formed by all the outputs of layer $m-1$ (including a unity output, which is used as a reference to bias the next layer) and is given by

$$O_{s,i}^{m-1} = \begin{cases} [\,1 \quad O_{s,1}^{m-1} \cdots O_{s,Z_m}^{m-1}\,]^{\mathcal{T}} & \text{for } m > 1, \\ [\,1 \quad I_s^{\mathcal{T}}\,]^{\mathcal{T}} & \text{for } m = 1 \end{cases} \tag{4}$$

III. Proportional Factor Term

The BP algorithm given by (2) is modified by adding an extra term in order to increase the BP learning speed [1]. This term is proportional to $e(W(k))$ which represents the difference between the output and the target at each iteration. For batch learning [1], $e(W(k)) = [\,e_s \quad e_s \quad \ldots \quad e_s\,]^{\mathcal{T}}$, where the vector e is of appropriate dimension and $e_s = \sum_{s=1}^{n} [T_s - O_s^M]$.

[1]For on-line learning, $e(W(k)) = [T_s - O_s^M]$ for the output layer weights, and $e_s = \sum (T_s - O_s^M)$ for the hidden layer weights. See [1] for further details.

The modified BP algorithm is hence

$$\Delta W(k) = \alpha(-\nabla E(W(k))) + \beta \Delta W(k-1) + \gamma e(W(k)), \tag{5}$$

where γ is the proportional factor(PF). It is noted that the BP algorithm given by Equation (5) has three terms, one proportional to the derivative of $E(W(k))$, another proportional to the previous value of the incremental change of the weights and a third term proportional to $e(W(k))$. These three terms can be viewed as being analogous to the three terms in a PID controller, commonly used in control applications.

IV. Estimation of Optimal LR, MF and PF Terms

The optimization of α, β, and γ, such that $W(k+1)$ minimizes E is required. The objective function E can be treated as a function with three independent variables $E(\alpha, \beta, \gamma)$. From Equation (5)

$$W(k+1) = W(k) + \alpha P(k) + \beta \Delta W(k-1) + \gamma e(k), \tag{6}$$

where $P(k) = -\nabla E(k)$ is a descent directional vector. Substituting Equation (6) into Equation (3) gives

$$\begin{aligned} o_{s,i}^m \;=\; & f([W_i^m(k) + \alpha P_i^m(k) + \beta \Delta W_i^m(k-1) \\ & + \gamma e_i^m(k)]^{\mathcal{T}} O_s^{m-1}). \end{aligned} \tag{7}$$

Computation of the first and second derivatives of E with respect to α, β and γ yields:

$$g(\alpha, \beta, \gamma) = \begin{bmatrix} \frac{\partial E(\alpha,\beta,\gamma)}{\partial \alpha} \\ \frac{\partial E(\alpha,\beta,\gamma)}{\partial \beta} \\ \frac{\partial E(\alpha,\beta,\gamma)}{\partial \gamma} \end{bmatrix} \tag{8}$$

where

$$\frac{\partial E(\alpha,\beta,\gamma)}{\partial \alpha} = \frac{-2}{nZ_M} \sum_{s=1}^{n} \left[T_s - O_s^M\right]^{\mathcal{T}} \frac{\partial O_s^M}{\partial \alpha} \tag{9}$$

$$\frac{\partial E(\alpha,\beta,\gamma)}{\partial \beta} = \frac{-2}{nZ_M} \sum_{s=1}^{n} \left[T_s - O_s^M\right]^{\mathcal{T}} \frac{\partial O_s^M}{\partial \beta} \tag{10}$$

$$\frac{\partial E(\alpha,\beta,\gamma)}{\partial \gamma} = \frac{-2}{nZ_M} \sum_{s=1}^{n} \left[T_s - O_s^M\right]^{\mathcal{T}} \frac{\partial O_s^M}{\partial \gamma} \tag{11}$$

The Hessian matrix of E is given by

$$H(\alpha, \beta, \gamma) = \begin{bmatrix} \frac{\partial^2 E(\alpha,\beta,\gamma)}{\partial \alpha^2} & \frac{\partial^2 E(\alpha,\beta,\gamma)}{\partial \alpha \partial \beta} & \frac{\partial^2 E(\alpha,\beta,\gamma)}{\partial \alpha \partial \gamma} \\ \frac{\partial^2 E(\alpha,\beta,\gamma)}{\partial \beta \partial \alpha} & \frac{\partial^2 E(\alpha,\beta,\gamma)}{\partial \beta^2} & \frac{\partial^2 E(\alpha,\beta,\gamma)}{\partial \beta \partial \gamma} \\ \frac{\partial^2 E(\alpha,\beta,\gamma)}{\partial \gamma \partial \alpha} & \frac{\partial^2 E(\alpha,\beta,\gamma)}{\partial \gamma \partial \beta} & \frac{\partial^2 E(\alpha,\beta,\gamma)}{\partial \gamma^2} \end{bmatrix} \tag{12}$$

which can be similarly computed.

To complete the computation of the gradient vector Equation (8) and the Hessian matrix Equation (12), the derivatives of O_s^M at $(\alpha_0, \beta_0, \gamma_0)$ can be computed from Equation (7). Thus the derivatives of the objective function $E(\mathbf{X})$ can be found, where

$$\mathbf{X} = [\,\alpha \quad \beta \quad \gamma\,]^{\mathcal{T}}. \tag{13}$$

A. Error Quadratic Approximation Approach

A second order Taylor polynomial of degree 2 can be used to approximate $E(\mathbf{X})$ for $\mathbf{X}$ near $(\alpha_0, \beta_0, \gamma_0)$. Since $E(\mathbf{X})$ has continuous second-order partial derivative, this gives

$$\begin{aligned} E(\mathbf{X}) &\approx E(\alpha_0,\beta_0,\gamma_0) + (\alpha-\alpha_0)\frac{\partial E}{\partial \alpha} + (\beta-\beta_0)\frac{\partial E}{\partial \beta} \\ &+(\gamma-\gamma_0)\frac{\partial E}{\partial \gamma} + \frac{1}{2}(\alpha-\alpha_0)^2\frac{\partial^2 E}{\partial \alpha^2} \\ &+\frac{1}{2}(\beta-\beta_0)^2\frac{\partial^2 E}{\partial \beta^2} + \frac{1}{2}(\gamma-\gamma_0)^2\frac{\partial^2 E}{\partial \gamma^2} \\ &+(\alpha-\alpha_0)(\beta-\beta_0)\frac{\partial^2 E}{\partial \alpha \partial \beta} + (\beta-\beta_0)(\gamma-\gamma_0) \\ &\frac{\partial^2 E}{\partial \beta \partial \gamma} + (\gamma-\gamma_0)(\alpha-\alpha_0)\frac{\partial^2 E}{\partial \gamma \partial \alpha} \\ &= \frac{1}{2}\Gamma^{\mathcal{T}} H_e \Gamma + \Gamma^{\mathcal{T}} g_e + a_e \end{aligned} \quad (14)$$

where $\Gamma = [\,\alpha-\alpha_0 \quad \beta-\beta_0 \quad \gamma-\gamma_0\,]^{\mathcal{T}}$, $a_e = E(\alpha_0,\beta_0,\gamma_0)$, and the gradient vector g is given by Equation (8) and the Hessian matrix H by Equation (12) .

1) Cases: Four separate cases are considered for computing optimal values for the learning parameters.

Case I: From [15], $E(\mathbf{X})$ has continuous second partial derivatives in a convex set $\mathbf{C}$ and let the Hessian matrix $H(\mathbf{X})$ at $\mathbf{X}$ be positive definite for all $\mathbf{X}$ in $\mathbf{C}$. Also let $\mathbf{y}$ be a critical point of $E(\mathbf{X})$ in $\mathbf{C}$. Then $E(\mathbf{X})$ is strictly convex in $\mathbf{C}$ and so $\mathbf{y}$ is strong global minimizer of $E(\mathbf{X})$ over $\mathbf{C}$. Suppose that E is a function with $E(0,0,0)=0$ and gradient $E(0,0,0)=0$. From Equation (14), the quadratic polynomial simplifies to

$$\begin{aligned} E(\mathbf{X}) &\approx \frac{1}{2}\alpha^2 E_{\alpha\alpha} + \frac{1}{2}\beta^2 E_{\beta\beta} + \frac{1}{2}\gamma^2 E_{\gamma\gamma} + \alpha\beta E_{\alpha\beta} \\ &+\beta\gamma E_{\beta\gamma} + \gamma\alpha E_{\gamma\alpha}. \end{aligned} \quad (15)$$

The discriminants are

$$D_1 = 4(\frac{1}{4}E_{\alpha\alpha})(\frac{1}{4}E_{\beta\beta}) - (E_{\alpha\beta})^2$$

$$D_2 = 4(\frac{1}{4}E_{\alpha\alpha})(\frac{1}{4}E_{\gamma\gamma}) - (E_{\alpha\gamma})^2$$

$$D_3 = 4(\frac{1}{4}E_{\beta\beta})(\frac{1}{4}E_{\gamma\gamma}) - (E_{\beta\gamma})^2$$

When H is a positive definite (i.e. $E_{\alpha\alpha} > 0$, $D_1 > 0$), symmetric, square matrix and ($D_2 > 0$, $D_3 > 0$), the optimal LR, MF and PF terms can be calculated as

$$\frac{dE}{d\Gamma} = H\Gamma + g = 0 \Rightarrow \Gamma = -H^{-1}g \quad (16)$$

It is noted that this procedure minimizes Equation (14).

Case II: When H is a positive definite matrix and at least one of D_2 or D_3 is negative, then $E(\alpha,\beta,\gamma)$ cannot be characterized as convex. However, $E(\alpha,\beta,0)$ is convex and optimal LR and MF can be calculated as in Case I by setting $\gamma = 0$.

Case III: When H is a non-positive definite matrix and $E_{\alpha\alpha} > 0$, the second order expansion of $E(\alpha,0,0)$ is convex along the descent direction of $P(k)$. Then the optimal LR can be calculated as in Case I by setting $\beta = \gamma = 0$.

Case IV: When H is a non-positive definite matrix and $E_{\alpha\alpha} < 0$, the optimization target behaves in an accelerated decreased manner along the descent direction $P(k)$ because both E_α and $E_{\alpha\alpha}$ take negative values. Then the optimal LR can be estimated by the line search method proposed by Yu et al. [7], which has been shown to be capable of providing an effective descent to the optimization target.

B. Approximation of the Sigmoidal nonlinearity function

Let

$$([W_i^M(k)+\alpha P_i^M(k)+\beta\Delta W_i^M(k-1)+\gamma e_i^M(k)]^{\mathcal{T}} O_s^{M-1}) = x$$

Then the sigmoidal nonlinear function of the output layer can be approximated by a set of linear functions:

$$f(x) = \begin{cases} m_1 x + b_1 & \text{for } x_1 \le x \le x_2, \\ m_2 x + b_2 & \text{for } x_1 \ge x, \\ m_2 x + (2b_1 - b_2) & \text{for } x_2 \le x \end{cases} \quad (17)$$

$$\begin{aligned} O_s^M &= f([W_i^M(k) + \alpha P_i^M(k) + \beta\Delta W_i^M(k-1) \\ &+\gamma e_i^M(k)]^{\mathcal{T}} O_s^{M-1}) \end{aligned} \quad (18)$$

By substituting Equation (18) into Equations (9)-(11) and equating e_α, e_β, and e_γ to zero yields:

$$\begin{aligned} &\alpha m_j P_i^M \sum_{s=1}^{n}\left[\frac{\partial O_s^M}{\partial \alpha}\right]^{\mathcal{T}} O_s^{M-1} + \beta m_j \Delta W_i^M(k-1) \\ &\times \sum_{s=1}^{n}\left[\frac{\partial O_s^M}{\partial \alpha}\right]^{\mathcal{T}} O_s^{M-1} + \gamma m_j e_i^M \sum_{s=1}^{n}\left[\frac{\partial O_s^M}{\partial \alpha}\right]^{\mathcal{T}} O_s^{M-1} \\ &= \sum_{s=1}^{n}\left[\frac{\partial O_s^M}{\partial \alpha}\right]^{\mathcal{T}} (T_s - m_j W_i^M(k) O_s^{M-1} - b_j) \end{aligned} \quad (19)$$

$$\begin{aligned} &\alpha m_j P_i^M \sum_{s=1}^{n}\left[\frac{\partial O_s^M}{\partial \beta}\right]^{\mathcal{T}} O_s^{M-1} + \beta m_j \Delta W_i^M(k-1) \\ &\times \sum_{s=1}^{n}\left[\frac{\partial O_s^M}{\partial \beta}\right]^{\mathcal{T}} O_s^{M-1} + \gamma m_j e_i^M \sum_{s=1}^{n}\left[\frac{\partial O_s^M}{\partial \beta}\right]^{\mathcal{T}} O_s^{M-1} \\ &= \sum_{s=1}^{n}\left[\frac{\partial O_s^M}{\partial \beta}\right]^{\mathcal{T}} (T_s - m_j W_i^M(k) O_s^{M-1} - b_j) \end{aligned} \quad (20)$$

$$\begin{aligned} &\alpha m_j P_i^M \sum_{s=1}^{n}\left[\frac{\partial O_s^M}{\partial \gamma}\right]^{\mathcal{T}} O_s^{M-1} + \beta m_j \Delta W_i^M(k-1) \\ &\times \sum_{s=1}^{n}\left[\frac{\partial O_s^M}{\partial \gamma}\right]^{\mathcal{T}} O_s^{M-1} + \gamma m_j e_i^M \sum_{s=1}^{n}\left[\frac{\partial O_s^M}{\partial \gamma}\right]^{\mathcal{T}} O_s^{M-1} \\ &= \sum_{s=1}^{n}\left[\frac{\partial O_s^M}{\partial \gamma}\right]^{\mathcal{T}} (T_s - m_j W_i^M(k) O_s^{M-1} - b_j) \end{aligned} \quad (21)$$

Since the matrix A_2 (Equation (23)) is a nonsingular, the optimal α, β and γ can be calculated by solving Equations (19)-(21) simultaneously:

$$\Gamma = A_2^{-1} R_2 \quad (22)$$

$$A_2 = \begin{bmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{bmatrix} \quad (23)$$

where

$$A_{11} = m_j P_i^M \sum_{s=1}^{n} \left[\frac{\partial O_s^M}{\partial \alpha}\right]^{\mathcal{T}} O_s^{M-1},$$

$$A_{12} = m_j \Delta W_i^M(k-1) \sum_{s=1}^{n} \left[\frac{\partial O_s^M}{\partial \alpha}\right]^{\mathcal{T}} O_s^{M-1}$$

$$A_{13} = m_j e_i^M \sum_{s=1}^{n} \left[\frac{\partial O_s^M}{\partial \alpha}\right]^{\mathcal{T}} O_s^{M-1},$$

$$A_{21} = m_j P_i^M \sum_{s=1}^{n} \left[\frac{\partial O_s^M}{\partial \beta}\right]^{\mathcal{T}} O_s^{M-1}$$

$$A_{22} = m_j \Delta W_i^M(k-1) \sum_{s=1}^{n} \left[\frac{\partial O_s^M}{\partial \beta}\right]^{\mathcal{T}} O_s^{M-1},$$

$$A_{23} = m_j e_i^M \sum_{s=1}^{n} \left[\frac{\partial O_s^M}{\partial \beta}\right]^{\mathcal{T}} O_s^{M-1}$$

$$A_{31} = m_j P_i^M \sum_{s=1}^{n} \left[\frac{\partial O_s^M}{\partial \gamma}\right]^{\mathcal{T}} O_s^{M-1},$$

$$A_{32} = m_j \Delta W_i^M(k-1) \sum_{s=1}^{n} \left[\frac{\partial O_s^M}{\partial \gamma}\right]^{\mathcal{T}} O_s^{M-1}$$

$$A_{33} = m_j e_i^M \sum_{s=1}^{n} \left[\frac{\partial O_s^M}{\partial \gamma}\right]^{\mathcal{T}} O_s^{M-1}$$

and

$$R_2 = \begin{bmatrix} \sum_{s=1}^{n} \left[\frac{\partial O_s^M}{\partial \alpha}\right]^{\mathcal{T}} (T_s - m_j W_i^M(k) O_s^{M-1} - b_j) \\ \sum_{s=1}^{n} \left[\frac{\partial O_s^M}{\partial \beta}\right]^{\mathcal{T}} (T_s - m_j W_i^M(k) O_s^{M-1} - b_j) \\ \sum_{s=1}^{n} \left[\frac{\partial O_s^M}{\partial \gamma}\right]^{\mathcal{T}} (T_s - m_j W_i^M(k) O_s^{M-1} - b_j) \end{bmatrix} \quad (24)$$

C. First-order Approximation Approach

In this approach, the convexity of $E(\alpha, \beta, \gamma)$ is obtained by substituting the first-order Taylor expansion of the output with respect to α, β, and γ in the objective function E. The first-order Taylor series approximation of the output is given by

$$O_s^M(\mathbf{X}) \approx O_s^M(\alpha_0, \beta_0, \gamma_0) + (\alpha - \alpha_0)\frac{\partial O_s^M}{\partial \alpha} + (\beta - \beta_0)\frac{\partial O_s^M}{\partial \beta} + (\gamma - \gamma_0)\frac{\partial O_s^M}{\partial \gamma}. \quad (25)$$

Substituting Equation (25) into Equation (1), the objective function becomes

$$E \approx \frac{1}{nZ_M} \sum_{s=1}^{n} \Big(T_s - O_s^M(\alpha_0, \beta_0, \gamma_0) - (\alpha - \alpha_0)\frac{\partial O_s^M}{\partial \alpha} - (\beta - \beta_0)\frac{\partial O_s^M}{\partial \beta} - (\gamma - \gamma_0)\frac{\partial O_s^M}{\partial \gamma} \Big)^2. \quad (26)$$

From Equation (26), setting the partial derivatives of E with respect to α, β and γ equal to zero, yields three equations in three unknowns:

$$(\alpha - \alpha_0) \sum_{s=1}^{n} \left\| \frac{\partial O_s^M}{\partial \alpha} \right\|^2 + (\beta - \beta_0) \sum_{s=1}^{n} \left[\frac{\partial O_s^M}{\partial \alpha}\right]^{\mathcal{T}} \frac{\partial O_s^M}{\partial \beta} + (\gamma - \gamma_0) \sum_{s=1}^{n} \left[\frac{\partial O_s^M}{\partial \alpha}\right]^{\mathcal{T}} \frac{\partial O_s^M}{\partial \gamma} = \sum_{s=1}^{n} \left[\frac{\partial O_s^M}{\partial \alpha}\right]^{\mathcal{T}} \times (T_s - O_s^M(\alpha_0, \beta_0, \gamma_0)) \quad (27)$$

$$(\alpha - \alpha_0) \sum_{s=1}^{n} \left[\frac{\partial O_s^M}{\partial \beta}\right]^{\mathcal{T}} \frac{\partial O_s^M}{\partial \alpha} + (\beta - \beta_0) \sum_{s=1}^{n} \left\| \frac{\partial O_s^M}{\partial \beta} \right\|^2 + (\gamma - \gamma_0) \sum_{s=1}^{n} \left[\frac{\partial O_s^M}{\partial \beta}\right]^{\mathcal{T}} \frac{\partial O_s^M}{\partial \gamma} = \sum_{s=1}^{n} \left[\frac{\partial O_s^M}{\partial \beta}\right]^{\mathcal{T}} \times (T_s - O_s^M(\alpha_0, \beta_0, \gamma_0)) \quad (28)$$

$$(\alpha - \alpha_0) \sum_{s=1}^{n} \left[\frac{\partial O_s^M}{\partial \gamma}\right]^{\mathcal{T}} \frac{\partial O_s^M}{\partial \alpha} + (\beta - \beta_0) \sum_{s=1}^{n} \left[\frac{\partial O_s^M}{\partial \gamma}\right]^{\mathcal{T}} \times \frac{\partial O_s^M}{\partial \beta} + (\gamma - \gamma_0) \sum_{s=1}^{n} \left\| \frac{\partial O_s^M}{\partial \gamma} \right\|^2 = \sum_{s=1}^{n} \left[\frac{\partial O_s^M}{\partial \gamma}\right]^{\mathcal{T}} (T_s - O_s^M(\alpha_0, \beta_0, \gamma_0)) \quad (29)$$

The optimal α, β and γ values can be calculated by solving Equations (27)–(29) simultaneously.

V. Simulation Results

Computer simulations for the learning parameters of the three-term backpropagation algorithm using three optimization approaches have been carried out. XOR problem, which is a popular benchmark for neural network training is employed [16]. The network architecture used for this problem consisted of four input units, two hidden units and one output unit. The same initial weights as well as the same learning rate, momentum factor and proportional factor are used for the algorithm in each optimization approach. The convergence of the learning process is measured by taking the half-sum-of-squared error as the objective function. The initial values of the weights are drawn randomly between $[-10, 10]$. For the example, the learning parameters of the plain three-term BP algorithm is carefully chosen so as to make the convergence of the learning process as fast as possible. To make sense, the convergence performances versus running time were compared. The learning process terminates when the iterations are over a fixed number or the total squared error is less than

TABLE I
XOR EXPERIMENT DETAILS

Methods	Terminated Iteration Number	Squared Error	Running time (sec.)
Error Quadratic Approximation with proportional factor	50	5×10^{-8}	35
Error Quadratic Approximation without proportional factor	250	3×10^{-3}	132
Approximation of the Sigmoidal Function with proportional factor	132	5×10^{-6}	43
Approximation of the Sigmoidal Function without proportional factor	322	3×10^{-3}	144
First Order Approximation with proportional factor	750	2×10^{-4}	211
First Order Approximation without proportional factor	978	3×10^{-3}	276
The plain Three-term BP	1432	3×10^{-3}	243

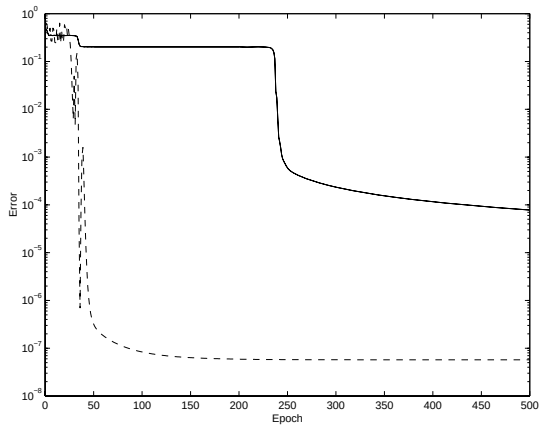

Fig. 1. Convergence performance comparison for XOR problem: Error Quadratic Approximation with Proportional Factor Approach (- - -) and Error Quadratic Approximation without Proportional Factor Approach (—): Evolution of error as a function of epoch number.

a small threshold. All the programs for the simulation were written in MATLAB and performed on a Dell PC-2.8GHz.

Theoretically speaking, all the present approaches can be explored to find the estimate of optimal learning parameters for the three-term BP algorithm. The Error Quadratic Approximation approach is preferred to the others, however, because in most cases it succeeds in getting a good estimate without requiring successive iterations. The others can be chosen as a substitution when the Error Quadratic Approximation approach fails to work.

The tests were performed to evaluate the convergence behaviour for the proposed optimization approaches. The terminated iteration number, averaged running time and error residuals for all the six methods are included in Table I. As can be seen, the learning rate optimization approaches own remarkable advantages in both fast convergence and time saving. The Error Quadratic Approximation approach with proportional factor behaves the best, as compared to the others. This is because it can provide with both much more effective descent direction and relatively accurate estimate of the optimal learning rate at the cost of a moderate increase in computational complexity. As to the First Order Approximation approach, though it exhibits better performance than the plain three-term BP. It behaves much inferior to the other optimization approaches. This degradation is due to the convex approximation of E that is often too crude in the extremely steep regions where the second derivative usually takes a negative value.

Figure 1 presents sample simulation results of the XOR example and the corresponding optimal learning rates versus different iteration number for the Error Quadratic Approximation with Proportional Factor Approach and Error Quadratic Approximation without Proportional Factor Approach. Note that the optimal learning rate sometimes varies from iteration to iteration. This give a sound support in necessity of using dynamic optimization for the learning parameters.

VI. CONCLUSIONS

In this paper a set of optimization approaches are developed and introduced to find the optimal learning parameters to improve the learning rate for the three-term PB algorithm. The

optimization approaches presented in this paper are based on simple manipulations of the first two derivative information, only a limited increase in computational complexity that is comparable to that of the plain three-term BP algorithm is required. Nevertheless, the benefit resultant from the learning parameters optimization is rather considerable. The convergence of the learning process is significantly accelerated and the overall running time for the learning procedure is consequently reduced to a great extent (by a factor up to 7). Quadratic Approximation with Proportional Factor approach is recommended for practical uses since it can provide significantly accelerated convergence at the cost of moderate increase in computational complexity, as compared to the plain theeterm BP algorithm.

ACKNOWLEDGMENTS

The author would like to thank Professor J.G. Taylor for his advice.

REFERENCES

[1] Zweiri, Y.H., Whidborne, J.F., & Seneviratne, L.D. Three-term backpropagation algorithm. *Neurocomputing*, 50:305–318, 2003.
[2] Rumelhart, D.E. & McClelland, J.L. *Parallel Distributed Proccessing: Explorations in the Microstructure of Cognition*, volume I. MIT Press, MA, 1986.
[3] Jacobs, R.A. Increasing rate of convergence through learning rate adaptation. *Neural Networks*, 1(4):295–307, 1988.
[4] Ooyen, A.O., & Neinhuis, B. Improving the convergence of the backpropagation algorithm. *Neural Networks*, 5:465–471, 1992.
[5] Rigler, A., Irvine, J., & Vodel, T. Rescaling of the variables in backpropagation learning. *Neural Networks*, 4:225–229, 1991.
[6] Yu, X.H., & Chen, G.A. Efficient backpropagation learning using optimal learning rate and momentum. *Neural Networks*, 10(3):517–527, 1997.
[7] Yu, X.H., Chen, G.A., & Cheng, S.X. Dynamic learning rate optimization of the backpropagation algorithm. *IEEE Transactions on Neural Networks*, 6(3):669–677, 1995.
[8] Salomon, R., & Hemmen, J.L. Accelerating backpropagation through dynamic self-adaptation. *Neural Networks*, 9(4):589–601, 1996.
[9] Fu, L.M., Hsu, H.H., & Principe, C.J. Incremental backpropagation learning networks. *IEEE Transactions on Neural Networks*, 7(3):757–761, 1996.
[10] Kuan, C.M., & Hornik, K. Convergence of learning algorithm with constant learning rates. *IEEE Transactions on Neural Networks*, 2(5):484–489, 1991.
[11] Gori, M., & Maggini, M. Optimal convergence of on-line backpropagation. *IEEE Transactions on Neural Networks*, 7:251–254, 1996.
[12] Karras, D.A., & Perantonis, S.J. An efficient constrained training algorithm for feedforward networks. *IEEE Transactions on Neural Networks*, 6:1420–1434, 1995.
[13] Ellacott, S.W. Techniques for the mathematical-analysis of neural networks. *Journal Of Computational And Applied Mathematics*, 50(1-3):283–297, 1994.
[14] Zweiri, Y.H., Seneviratne, L.D., and Althoefer, K. Stability analysis of a three-term backpropagation algorithm. *Neural Networks*, 18(10):1341–1347, 2005.
[15] Wolfe, M.A. *Numerical Methods for Unconstrained Optimization*. VNR, Wokingham, U.K., 1978.
[16] Ampazis, N., Perantonis, S.J., & Taylor, J.G. Dynamics of multilayer networks in the vicinity of temporary minima. *Neural Networks*, 12:43–58, 1999.