Improved Three-term Conjugate Gradient Algorithm For Training Neural Network

Assist. Prof. Dr. AbbasH.Taqi
Department of Mathematics, College of Science
University of Kirkuk
e-mail: dr.altaqi@yahoo.com

Abstract

A new three-term conjugate gradient algorithm for training feed-forward neural networks is developed. It is a vector based training algorithm derived from DFP quasi-Newton and has only O(n) memory. The global convergence to the proposed algorithm has been established for convex function under Wolfe condition. The results of numerical experiments are included and compared with other well known training algorithms in this field.

Keywords: Feed Forward Neural Network, Training Algorithms, Optimization,Three Terms Conjugate Gradient algorithms Neural Network.

1. Introduction

Learning system, such as multi-Layer feed forward neural networks (FFNN) are parallel computation models comprised of densely interconnected adaptive processing units, characterized by an inherent propensity for learning from experience also discovering new knowledge. Due to their excellent capability of self-learning and self-adapting. They have been successfully applied in many areas of artificial intelligence and are often found to be more efficient and accurate than more classification techniques [1]. The operation of a FFNN is usually based on the following equations [2]:

$$net_j^l = \sum_{i=1}^{N_{i-1}} w_{ij}^{l-1,l} y_i^{l-1} + b_j^l \quad (1)$$

$$y_j^l = f\left(net_j^l\right) \quad (2)$$

where net_j^l is the sum of its weighted inputs for the j-th node in the l-th layer$(j = 1,2,\dots,N_l)$, $w_{ij}^{l-1,l}$ are the weights from the i-th neuron at the $(l-1)$ layer to the j-th neuron at the l-th layer, b_j^l is the bias of the j-th neuron at the l-th layer, $y_j^l$is the output of the j-th neuron that belongs to the l-th layer, and $f\left(net_j^l\right)$ is the neuron activation function[2].

The problem of training a neural network is to iteratively adjust its weights in order to globally minimize a measure of difference between the actual output of the network and desired output for all examples of the training see [3] . More mathematically, the training process can be formulated as the minimization of the error function$E(w)$, defined by the sum of square differences between the actual outputs of the FNN denoted by $y_{j,p}^l$ and the desired output denoted by $t_{j,p}$, relative to the appeared output namely

$$E(w) = \frac{1}{2}\sum_{p=1}^{P} \sum_{j=1}^{N_L} (y_{j.p}^l - t_{j.p})^2 \quad (3)$$

where$w \in R^n$the vector network weights and p represents the number of patterns used in the training set.

This paper is organized as follows. Section 2 gives an overview of conjugate gradient methods and Quasi- Newton methods. Section 3 describes a new three-term conjugate (NTCG) gradient method. Section 4 presents the global convergence analysis of the proposed algorithm. Section 5 evaluates NTCG performance in comparison with other major learning algorithms.

2. Optimization Routines

Methods to speed up the learning phase and to optimize the learning process in (FFNN) have been recently studied and several new learning algorithms have been discovered see [4, 5].

Conjugate gradient methods are probably the most famous iterative methods for efficiently training neural networks due to their simplicity,numerical efficiency and their very low memory requirements. These methods generate a sequence of weights $\{w_k\}$ using the iterative formula

$$w_{k+1} = w_k + \alpha_k d_k \qquad (4)$$

where k is the current iteration usually called epoch. $w_1 \in R^n$ is a given initial point, $\alpha_k > 0$ is the learning rate and d_k is a descent (by descent we mean$d_k^T g_k < 0$) search direction defined by $(d_1 = -g_1)$ and

$$d_{k+1} = -g_{k+1} + \beta_{k+1} d_k \qquad (5)$$

whereg_k is the gradient of E at w_k and β_k is a scalar parameter. In the literature, there have been proposed several choices for β_k which give rise to distinct conjugate gradient methods [4]. The most well-known conjugate gradient methods include the Fletcher- Reeves (FR) method [6]. The update parameter of this method is specified as follow:

$$\beta^{FR} = \frac{\|g_{k+1}\|^2}{\|g_k\|^2}$$

where$\|\,.\,\|$ denotes to the Euclidean norm.

The formal steps for Fletcher-Reeves (FR) algorithm are given as follows:

FR Algorithm

Step1. Initiate w_1, $0 < \rho < \sigma < 1$, E_r and K_{max}: set k=1

Step2. Calculate the error function E_k and its gradient g_k

Step3. If $(E_k < E_r)$ return $w_* = w_k$ and $E_* = E_k$

Step4. If $(\|g_k\|^2 = 0)$ return error goal not met

Step5. Compute the search direction:

$$d_{k+1} = -g_{k+1} + \beta^{FR} d_k$$

Step6. Compute learning rate α_k using Wolfe line search conditions

(15) and (16)

Step7. Update the weights: $w_{k+1} = w_k + \alpha_k d_k$ and set $k = k + 1$

Step8. If $(k > k_{MAX})$ return goal not met else go to step 2.

A major difficulty with the conjugate gradient methods that is for a general function, the obtained directions are not necessarily descent directions and numerical instability can be resulted [7].

Battiti in [5] suggested One-Step-Secant (OSS) method for training FFNN, he defined the Search direction as follows:

$$d_{k+1} = -g_{k+1} + A_k s_k + B_k y_k$$

where $s_k = w_{k+1} - w_k$ and$y_k = g_{k+1} - g_k$, the two scalar parameters A_k and B_k are the following combination of scalar products of the previously defined vectors s_k, g_{k+1}, and y_k:

$$A_k = -\left(1 + \frac{y_k^T y_k}{s_k^T y_k}\right)\frac{s_k^T g_{k+1}}{s_k^T y_k} + \frac{y_k^T g_{k+1}}{s_k^T y_k}$$

$$B_k = \frac{s_k^T g_{k+1}}{s_k^T y_k}$$

Parker in [8] derived a second order method which approximates Newton's method. However, the scheme requires the evaluation of a square matrix with components of second partial derivatives which are not available for a general neural network. Reference [5] evaluated the back-propagation, steepest descent with line search, momentum method and classical Quasi-Newton (QN) method and concluded that the use of QN methods is mandated by their excellent convergence properties. In the QN methods, the first search direction is$d_1 = -g_1$, then the sequence$\{w_k\}$ of approximations to the minimizer generated using the iteration formula given in equation (4)

where

$$d_{k+1} = -H_{K+1}g_{k+1} \quad (6)$$

whereH_{k+1} is an approximation to the inverse of theHessian matrix for the error function E(W).There are different formulas for H_{k+1} see [9]. In this paper we consider the David, Fletcher and Powell (DFP) update [9] defined by

$$H_{k+1}^{DFP} = H_k + \frac{s_k s_k^T}{s_k^T y_k} - \frac{H_y y_k^T H_k}{y_k^T H_k y_k} \quad (7)$$

whereH_1 is the identity matrix.

Learning algorithms based on QN methods have some drawbacks such as matrix storage and burden of step length (learning rate) computation. A line search to calculate a reasonably accurate learning rate is indispensable for theses algorithms. In order to provide desirable performance, an efficient and reasonable accurate learning rate is needed [8].

3. New three-term conjugate gradient method (NTCG)

One characteristic of classical Quasi-Newton method is that the search directions provided them change if the objective function is scaled [9],this is obviously undesirable, the other drawbacks for the QN methods they requires storage matrix of size $n \times n$ and a number of calculations of order$Q(n^2)$. On the other hand the search directions generated by the conjugate gradient methods generally not descent. To overcome to the above mentioned drawbacks, we suggest a new three term CG method (NTCG) as follows. Consider the DFP update formula defined in equation (7), if we assume H_k is identity matrix at iteration k, then the scaled form of H_{k+1} can be written as

$$H_{k+1} = \theta_k \left(I + \frac{s_k s_k^T}{y_k^T y_k} \right) - \frac{y_k y_k^T}{y_k^T y_k} \quad (8)$$

whereθ_k is scalar then:

$$d_{k+1} = -H_{k+1}g_{k+1}$$

$$= -\theta_k g_{k+1} - \theta_k \frac{s_k^T g_{k+1}}{s_k^T y_k} s_k + \frac{y_k^T g_{k+1}}{y_k^T y_k} y_k \quad (9)$$

To find the value of θ_k assume that

$$d_{k+1}^T g_{k+1} = -\|g_{k+1}\|^2 \quad (10)$$

Then from (9) and (10) we get

$$(1-\theta_k)\|g_{k+1}\|^2 - \theta \frac{(s_k^T g_{k+1})^2}{s_k^T y_k} + \frac{(y_k^T g_{k+1})^2}{\|y_k\|^2} = 0 \quad (11)$$

Using Coushy-Shwartz Inequality for $y_k^T g_{k+1}$ i.e

$$(y_k^T g_{k+1})^2 \le \|y_k\|^2 \|g_{k+1}\|^2 \quad (12)$$

From (11) and (12) with simple algebra we get

$$\theta = \frac{2 s_k^T y_k \|g_{k+1}\|^2}{s_k^T y_k \|g_{k+1}\|^2 + (s_k^T g_{k+1})^2} \quad (13)$$

Hence

$$d_{k+1} = -\theta_k g_{k+1} - \theta_k \frac{s_k^T g_{k+1}}{s_k^T y_k} s_k + \frac{y_k^T g_{k+1}}{y_k^T y_k} y_k \quad (14)$$

In the implementations of CG methods, one often, requires the inexact line search such as the standard Wolfe line search requires the learning rate α_k satisfying the following equations:

$$E(w_k + \alpha_k d_k) \le E(w_k) + \rho \alpha_k d_k^T g_k \quad (15)$$

$$g(w_k + \alpha_k d_k)^T d_k \ge \sigma d_k^T g_k \quad (16)$$

where $0 < \rho < \sigma < 1$. Note that if d_k is descent direction then second Wolfe condition (16) gives,$s_k^T y_k > 0$ Therefore$\theta > 0$.

At this point, we present a high level description of the proposed algorithm called new three-term conjugate gradient algorithm (NTCG).

NTCG Algorithm:

Step1. Initiate w_1, $0 < \rho < \sigma < 1$, E_r and K_{max}: set k=1

Step2. Calculate the error function E_k and its gradient g_k

Step3. If $(E_k < E_r)$ return $w_* = w_k$ and $E_* = E_k$

Step4. If $(\|g_k\|^2 = 0)$ return error goal not met

Step5. Compute the search direction d_k using equations (13) and (14)

Step6. Compute learning rate α_k using Wolfe line search conditions
(15) and (16)

Step7. Update the weights:$w_{k+1} = w_k + \alpha_k d_k$and set$k = k + 1$

Step8. If $(k > k_{MAX})$ return goal not met else go to step 2.

4. Convergence analysis

In order to establish the global convergence result for our proposed method, we will impose the following assumptions on the Error function E

A.1: The level set $\pounds = \{w \in R^n : E(w) \leq E(w_1)\}$ is bounded i.e. there exist a positive constant $B > 0$ such that

$$\|w\| \leq B \ \forall \ w \epsilon \pounds \qquad (17)$$

A.2: In some neighborhood $Nof\pounds$ the error function E is continuously differentiable and its gradient g is Lipchitz continuous i.e. there exist a constant $L > 0$ such that

$$\|g_{(w)} - g_{(\bar{w})}\| \leq L\|w - \bar{w}\| \forall w, \bar{w} \epsilon \pounds \qquad (18)$$

Notice that since E is bounded below in R^n by zero, it is differentiable and its gradient is Lipchitz continuous, assumptions A.1 and A.2 always holds [10]. Furthermore under these assumptions on E there exist a constant $\boldsymbol{r} > 0$such that

$$\|g_{(w)}\| \leq \boldsymbol{r} \forall w \epsilon \pounds \qquad (19)$$

Proposition (4.1)

Suppose that the learning rate α_k satisfies the Wolfe conditions (15) and (16), then d_{k+1} given by equations (13) and (14) is a descent direction. i.e. $d_{k+1}^T g_{k+1} < 0$

Proof

The proof is consequence from Wolfe conditions and definition of θ.

Proposition (4.2)

If the learning rate α_k satisfies Wolfe conditions then θ defined by equations (13) bounded by. i.e.

$$1 \leq \theta \leq 2$$

Proof

$$\theta = \frac{2 s_k^T y_k \|g_{k+1}\|^2}{s_k^T y_k \|g_{k+1}\|^2 + (s_k^T g_{k+1})^2} \leq \frac{2 s_k^T y_k \|g_{k+1}\|^2}{s_k^T y_k \|g_{k+1}\|^2} = 2$$

On the other hand from second Wolfe condition (15) and proposition (4.1) we have $s_k^T y_k > 0$ and $s_k^T g_{k+1} \leq s_k^T y_k$,then

$$\theta = \frac{2 s_k^T y_k \|g_{k+1}\|^2}{s_k^T y_k \|g_{k+1}\|^2 + (s_k^T g_{k+1})^2} \geq \frac{s_k^T y_k \|g_{k+1}\|^2 + (s_k^T g_{k+1})^2}{s_k^T y_k \|g_{k+1}\|^2 + (s_k^T g_{k+1})^2} = 1$$

Proposition (4.3)

Suppose that d_k is a descent direction and that g_k satisfies the Lipschitz condition (18) for all w on the line segment connecting w_k and w_{k+1} , if the learning rate α_k satisfies Wolfe conditions (15) and (16) then

$$\alpha_k \geq \frac{(1-\sigma)}{L} \frac{g_k^T d_k}{\|d_k\|^2} \qquad (20)$$

Proof

The proof is exactly the same as that of proposition (4.1) in [11], thus we omit it.

To prove the global convergence of non-linear conjugate gradient algorithm, often the following zoutendijk condition is used.

Zoutendijk condition [12]

Suppose that assumptions A.1 and A.2 hold consider any algorithm of the form

$w_k + \alpha_k d_k$

where d_k is a descent direction and α_k satisfies Wolfe condition (15) and (16) then

$$\sum_{k=1}^{\infty} cos^2 \emptyset \|g_k\|^2 < \infty \quad (21)$$

where $\emptyset$ is the angle between d_k and $-g_k$ define by

$$\cos \emptyset = -\frac{g_k^T d_k}{\|d_k\|\|g_k\|} \quad (22)$$

The following proposition shows that our three-term conjugate gradient method satisfies Zoutendijk condition (21).

Proposition (4.4)

Suppose that assumptions A.1 and A.2 hold consider the NTCG algorithm where $\square_{\square}$ is descent direction and $\square_{\square}$ satisfies Wolfe conditions (15) and (16) then $\Sigma_{k=1}^{\infty} \frac{(d_k^T g_k)^2}{\|d_k\|^2} < +\infty$

Proof

From proposition (4.1) and (4.2) we have

$$\alpha_k \geq \frac{(1-\sigma)}{L} \frac{g_k^T d_k}{\|d_k\|^2} \geq -\frac{(\sigma-1)}{L\|d_k\|^2} g_k^T d_k \quad (23)$$

From first Wolfe condition (15) and (23) we get

$$E_k - E_{k+1} \geq -\rho\, \alpha_k g_k^T d_k$$

$$= \rho \frac{(1-\sigma)(g_k^T d_k)^2 \|g_k\|^2}{L\, \|d_k\|^2 \|g_k\|^2}$$

$$= c_{\circ} cos^2 \emptyset\, \|g_k\|^2$$

Where $c_{\circ} = \rho \frac{(1-\sigma)}{L}$. Since E is bounded below,

$$\sum cos^2 \emptyset \|g_k\| < +\infty \quad (24)$$

In conjugate gradient algorithms, the iteration can fail, in the since that $\|g_k\| \geq \gamma > 0$ for all k, only if $\|d_k\| \to \infty$ sufficiently rapidly [1, 13].More exactly, the sequence of gradient norms $\|g_k\|$ can be bounded a way from zero only if

$$\sum_{k=1}^{\infty} \frac{1}{\|d_k\|^2} < \infty \quad (25)$$

For any conjugate gradient method with Wolfe conditions, the following general result holds [13].

For any conjugate gradient method with Wolfe conditions, the following general result holds [13].

Lemma (4.1) [11]

Suppose that assumptions A.1 and A.2 hold and consider any conjugate gradient algorithm (4), where $\square_{\square}$ is a descent direction and $\square_{\square}$is obtained by Wolfe line search conditions .If

$\Sigma_{k=1}^{\infty} \frac{1}{\|d_k\|} = \infty$Then $lim_{k\to\infty} \inf \|g_k\|^2 = 0$

For uniformly convex function we can prove that the norm of the direction d_{k+1} generated by (13) and (14) is bounded above. Therefore by lemma (4.1), we can prove the following theorem.

Theorem (4.1)

Suppose that assumptions A.1 and A.2 hold and consider the algorithm NTCG. Where d_k is a descent direction and α_k computed by the Wolfe conditions, suppose that f is a uniformly convex functions,i.e. there exists a constant $\mu > 0$ such that

$$s_k^T y_k \geq \mu \|s_k\| \quad (26)$$

Then

$$lim_{k\to\infty} \|g_k\| = 0$$

Proof

From Lipchitz we have $\|y\| \leq L\|s\|$. On the other hand from uniformly convexity,$s_k^T y_k \geq \mu \|s_k\|^2$. Using the Cauchy inequality, assumption A.1 and equation (19) we have

$$\|d_{k+1}\| = \left\| -\theta\, g_{k+1} - \theta \frac{s_k^T g_{k+1}}{s_k^T y_k} s_k + \frac{y_k^T g_{u+1}}{y_k^T y_k} y_k \right\|$$

$$\leq 2\|g_{k+1}\| + 2\frac{\|s\|\|g_{k+1}\|}{\mu} + \|g_{k+1}\|$$

$$\leq 2\,\text{г} + \frac{2\,\text{г}B}{\mu} + \text{г} = \text{г}\left(3 + \frac{2B}{\mu}\right)$$

Therefore $\|d_{k+1}\|$ is bounded above and so

$$\sum_{u=1}^{\infty} \frac{1}{\|d_k\|} = \infty$$

then by lemma (4.1) $lim_{u\to\infty}\|g_k\| = 0$.

5. Experimental Results

In this section, we will present experimental results in order to evaluate the performance of our proposed algorithm (NTCG) in three test problems [3,10].The suggested (NTCG) algorithm was compared with three other training algorithms:One step secant algorithm (OSS), DFP Quasi-Newton and FR conjugate gradient algorithm.The implementation code was written in Matlab7.5. The methods DFP, FR, OSS and NTCG are implemented with the line search proposed in CONMIN [14]. The heuristic parameters were set as □ $= 10^{-4}$ and □ $= 0.5$,as in [11].All networks have received the same sequence of input patterns and the initial weights generated using the Nguyen-Widrow method [13]. The results have been averaged over 100 simulations.

For each of the test problem a table and figure summarize the result of all algorithms for simulations that reach solution is presented. The reported parameters are P_{mean} the mean value of epochs, T_{av} the average of total time and S_c the succeeded simulations out of 100 trails within error function evolutions limits.

5.1The continuous function approximation problem

The first problem we have considered is the approximation of the continuous function,f(x)=sin(2πx)+0.1*rand(size(x))

where x=-1:0.05:1. This problem maps one real input to a single real output. The selected architecture of the FFNN is 1-10-1 with logisticneurons with biases in the hidden layer and with a linear output neuron with bias. The error goal hasbeen set to 0.001 and the maximum epochs to 1000. The results of the simulations are presented in Table(1) and figure-1.

Table (1): The Con. Function Approx. Problem.

Algorithms	**P_{mean}**	**T_{av}**	**S_c**
DFP	26.78	25.19 s	94%
OSS	38.9	29.80 s	100%
FR	60.86	35.55 s	88%
NTCG	38.04	28.71 s	100%

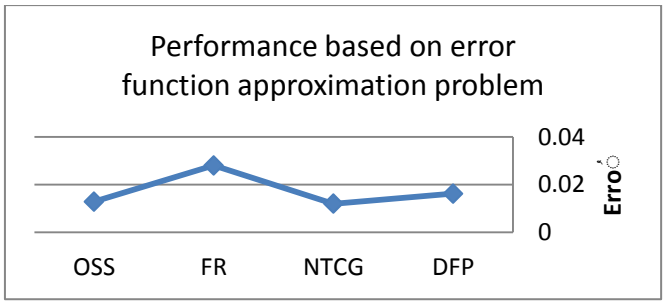

Figure 1: Comparison of algorithms based on error

5.2 Iris Classification Problem

This benchmark is perhaps the most best known to be found in the pattern recognition literature [8].The data set contains 3 classes of 50 instances each, where each class refers to a type of iris plant. Thenetwork architectures constitute of 1 hidden layer with 7 neurons and an output layer of 3 neurons. Thetraining goal was set to $E_{rr} \leq 0.01$ within the limit of 1000 epochs, and all networks were tested using 10-foldcross-validation [10].

Table 2: The Iris Classification Problem.

Algorithms	**P_{mean}**	**T_{av}**	**S_c**
DFP	69.12	1.9456 s	90%
OSS	65	1.4328 s	100%
FR	65.88	1.4463 s	86%
NTCG	58.04	1.3897 s	100%

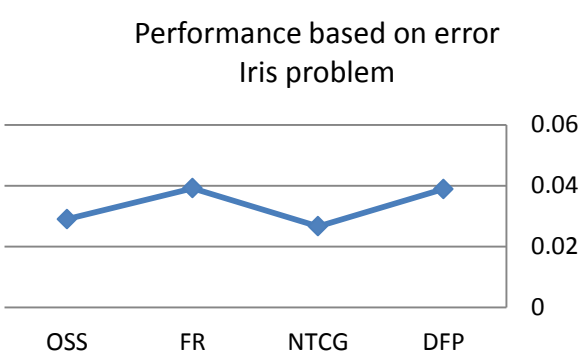

Figure 2: Comparison of algorithms based on error

5.3 SPECT Heart Problem.

This dataset contains data instances derived from cardiac Single Proton Emission Computed Tomography(SPECT) images from the University of Colorado. This is also a binary classification task, where patientsheart images are classified as normal or abnormal. The class distribution has 55 instances of the abnormalclass (20.6%) and 212 instances of the normal class (79.4%). From them there have been selected 80 instancesfor the training process and the remainder 187 for testing the neural networks generalization capability.The network architecture for this medical classification problem constitute of 1 hidden layer with 6 neuronsand an output layer of 2 neurons. The termination criterion is set to $E_{rr} \leq 0.1$ within the limit of 1000epochs [8].

Table 3: The Spect-Heart. Problem

Algorithms	$\mathbf{P_{mean}}$	$\mathbf{T_{av}}$	$\mathbf{S_c}$
DFP	12.64	0.80204	100%
OSS	11.0	0.68324	100%
FR	8.76	0.6384	100%
NTCG	9.56	0.6661	100%

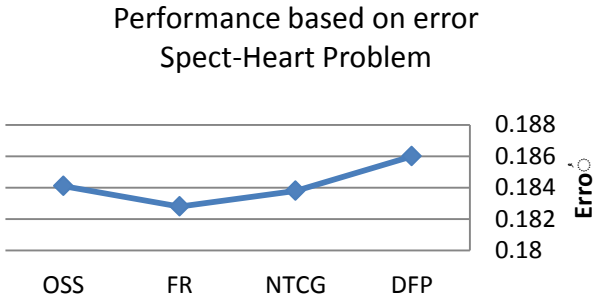

Figure 3: Comparison of algorithms based on error

6. Conclusion

In this paper, a three conjugate gradient algorithm has been proposed based on DFP quasi- Newton method, since the DFP algorithm generates conjugate search directions independent of line search, while the BFGS quasi-Newton not poses this property. An attractive property of the proposed method is that, it ensures sufficient descent, avoiding thereby the usually inefficient restarts. Under mild conditions, the global convergent property was established.Based on the numerical experiments, the proposed method outperforms FR and OSS algorithms.

Reference

[1] Haykin S(1994). 'Neural Network A comprehensive function: MacmillanCollege Publishing Company New york".

[2] Rumelhart D., Hinton G., and Williams R (1986).'Learning internal representation by error propagation' . In Rumelhart D and Mcllandeditors, parallel Distributed processing explorations in theMicrostructure of conjugate Cambridge.

[3] Livieris I. and Pintelas P.(2009).' Performance evaluation ofdescent CG methods for neural network training.In E.A. Lipitakis, editor, 9th Hellenic European Research on Computer Mathematic & its Applications Conference (HERCMA'09).

[4] Kalil A. and Hind M. (2014). 'Conjugate gradient algorithm based on Aiten'sProcess

الخلاصة

تم في هذا البحث اقتراح خوارزمية متجهات مترافقة وات الحدود الثلاثة لتعليم الشبكات العصبية الاصطناعية ذوات التغذية الأمامية. الخوارزمية المقترحة لا يحتاج إلى خزن مصفوفة إنما فقط بعض المتجهات. تم اشتقاق هذه الخوارزمية اعتمادا على صيغة ديفيد-فليجر-باول الشبيهة بالنيوتن. التقارب المطلقة لهذه الطريقة المقترحة تم إثباته نظرياً في حالة كون دالة الهدف محدبة وتحت شرط ولف. أختبر الخوارزمية في ثلاث مسائل للاختبارومقارنتها مع بعض الخوارزميات المعروفة في هذا المجال.

for training neural networks'. Raf. J. of Comp & Math. Vol.(11), No.(1).

[5] Battiti R. (1992). '1st and 2nd order method for learning betweensteepest descent and Newton method'. Neural Comp. 4(2).

[6] Fletcher R. and Reeves C. (1964).' function minimization byconjugate gradients'. Computer Journal. (7).

[7] Noccedal J.(1996).' conjugate gradient methods and nonlinearoptimization. In [Adams.J.L.Nazareth (EDS.) linear and nonlinear conjugate gradient related methods. SIAM.

[8] Parker D.(1987).'Optimal algorithms for adaptive Networks:second order back-propagation, second order Direct propagation,and second order Hebian learning.' Proceedings of the 1987 IEEE International Conference on Neural Networks, vol (2).

[9] Andreas A. and Wu S. (2007).'Practical Optimization Algorithms and Engineering Applications', Springer Science and Business Media, LLC New York.

[10] Livieris I., and pintelas P.(2011).'An advanced conjugate gradientraining algorithm based on a modified secant equation'. TechnicalReport No. TR11-03'. University of patras. Dep.of Math. GR-265 04 patras , Greece.

[11] Andrei N.(2013). 'A simple three-term conjugate gradient f unconstrained optimization'. J. of Computation and AppliedMathematics (241).

[12] Zoutendijk G. (1970). 'Non-linear programming computational Methods', in J. Abadi (Ed). Integer and Non-linear Programming,North Holland.

[13] Nguyen D. and Widrow B. (1990).' Improving the learning speedof 2-layer neural network by choosing initial values of adaptiveweights. Biological Cybernetics, 59.

[14] Shanno D. and Phua K.(1976).' Minimization of unconstrained multivariate functions'. ACM Transactions on Mathematical Software, 2.