MODIFIED CONJUGATE GRADIENT METHOD FOR TRAINING NEURAL NETWORKS BASED ON LOGISTIC MAPPING

ALAA LUQMAN IBRAHIM* and SALAH GAZI SHAREEF**

*Dept. of Mathematics, College of Science, University of Duhok, Duhok, Kurdistan Region-Iraq.
**Dept. of Mathematics, Faculty of Science, University of Zakho, Zakho, Kurdistan Region-Iraq.

(Received: October 17, 2018; Accepted for Publication: March 25, 2019)

ABSTRACT

In this paper, we suggested a modified conjugate gradient method for training neural network which assures the descent and the sufficient descent conditions. The global convergence of our proposed method has been studied. Finally, the test results present that, in general, the modified method is more superior and efficient when compared to other standard conjugate gradient methods.

KEYWORDS: artificial neural networks, conjugate gradient, global convergence, descent and sufficient descent conditions.

1. INTRODUCTION

Artificial neural networks (ANNs) are parallel computational samples consist of processing’s units and interconnected densely discriminated by an inherent propensity for learning from test and also discovering new knowledge. Because of their excellent ability of self-learning and self-adapting, they have been successfully applied in many aspects of artificial intelligence [2,6,7]. They are often found to be more active and precise than other classification techniques [3]. Although several different ways have been suggested, the feed forward neural networks (FNNs) are the most familiar and widely used in different kinds of applications.

Training of neural networks (NNs) can be formulated as a problem of nonlinear unconstrained optimization. Therefore, the training procedure can be achieved by minimizing the error function $E(\mathbf{w})$, defined by the sum of square differences between the actual output of the FNN, pointed by $\mathbf{o}^h_j$, and the wanted output, pointed by $\mathbf{t}^h_j$, relative to the appeared output, namely,

$$E(\mathbf{w}) = \frac{1}{2} \sum_{h=1}^{N} \sum_{j=1}^{h} (\mathbf{o}^h_j - \mathbf{t}^h_j)^2 = \sum_{h=1}^{N} E_h$$

(1.1)

where $\mathbf{w} \in \mathbb{R}^n$ is the vector network weights and the number of patterns used in the training set represented by $\mathbf{h}$. [8]

one of the most important iterative methods for efficiently training neural networks in scientific and engineering computation is called conjugate gradient method (CG) because of their simplicity and their very low memory requirements [4,5,12,14,17]. The conjugate gradient method produce a sequence of weights $\{\mathbf{w}_i\}$, is given by:

$$\mathbf{w}_{i+1} = \mathbf{w}_i + \lambda_i \mathbf{p}_i$$

(1.2)

where $i$ is the number of iteration generally called epoch, $\lambda_i > 0$ is the learning rate and the search direction $\mathbf{p}_i$ which is computed by:

$$\mathbf{p}_0 = -\nabla f_0 \quad \text{and} \quad \mathbf{p}_{i+1} = -\nabla f_{i+1} + \beta_i \mathbf{p}_i \quad \text{for} \quad i \geq 1,$$

(1.3)

where $\nabla f_i$ pointed to the gradient of $E(\mathbf{w})$ at the point $\mathbf{w}_i$ and the scalar $\beta_i$ is a known as the coefficient of (CG). The parameter $\beta_i$ of the classical formula are determined as follows:

$$\beta_{i}^{PR} = \frac{\mathbf{g}_{i+1}^T \mathbf{g}_i}{\mathbf{g}_i^T \mathbf{g}_i}, \quad \text{Polak and Ribiere (PR)}$$

(1.4)

$$\beta_{i}^{HS} = \frac{\mathbf{g}_{i+1}^T \mathbf{y}_i}{\mathbf{p}_i^T \mathbf{y}_i}, \quad \text{Hestenes and Steifel (HS)}$$

(1.5)

$$\beta_{i}^{FR} = \frac{\mathbf{g}_{i+1}^T \mathbf{g}_i}{\mathbf{g}_i^T \mathbf{g}_i}, \quad \text{Fletcher and Reeves (FR)}$$

(1.6)

$$\beta_{i}^{CD} = \frac{\mathbf{g}_{i+1}^T \mathbf{g}_i}{-\mathbf{g}_i^T \mathbf{p}_i}, \quad \text{Conjugate Descent (CD)}$$

(1.7)

$$\beta_{i}^{DY} = \frac{\mathbf{g}_{i+1}^T \mathbf{g}_i}{\mathbf{p}_i^T \mathbf{y}_i}, \quad \text{Dai and Yuan (DY)}$$

(1.8)

$$\beta_{i}^{LS} = \frac{\mathbf{g}_{i+1}^T \mathbf{g}_i}{-\mathbf{g}_i^T \mathbf{p}_i}, \quad \text{Liu and Storey (LS)}$$

(1.9)

for the above equation see [9,18,19,20,21,22].

The globally convergence of the above conjugate gradient methods has been studied by many authors with under some different line
searches [1,10,13]. To prove the convergence condition of the nonlinear CG methods, it is usually need that the step size $\lambda_i$ should achieve the following standard strong Wolfe conditions:

$$E(w_i + \lambda_i p_i) \leq E(w_i) + \rho \lambda_i g_i^T p_i \quad \text{(1.10)}$$

$$|g(w_i + \lambda_i p_i)^T p_i| \leq -\sigma g_i^T p_i \quad \text{(1.11)}$$

however, the standard Wolfe condition (1.10) and

$$g(w_i + \lambda_i p_i)^T p_i \geq \sigma g_i^T p_i \quad \text{(1.12)}$$

where $0 < \rho < \sigma < 1$

is used to prove the convergence of many other numerical methods such as (quasi-Newton method).

In this paper, will present our modified CG training algorithm in section 2. The descent and sufficient descent conditions of our modified method are proved in section 3. The global convergences of the proposed algorithm are discussed in section 4. Some numerical results are contained in section 5. Finally, conclusions are given in the last section.

2. MODIFIED CONJUGATE GRADIENT TRAINING ALGORITHM

In this section, suggested a modified CG training algorithm by using conjugate gradient coefficient of (Fletcher and Reeves) method and logistic mapping which is used extensively [16].

From the logistic mapping and (1.6), we have

$$\beta_i = \beta_i^{FR} (1 - \beta_i^{FR}) \quad \text{(2.1)}$$

where $0 \leq \beta_i \leq 1$.

To achieve balance, we will multiply first term of (2.1) by scalar $\gamma$, we get

$$\beta_i^{New} = \mu \beta_i^{FR} (y - \beta_i^{FR}) \quad \text{(2.2)}$$

and implies that

$$\beta_i^{New} = \frac{\beta_i^{New} g_i}{y_i} - \mu \beta_i^{New} (\beta_i^{New} g_i)^2 \quad \text{(2.3)}$$

or $\beta_i^{New} = \beta_i^{DY} - \mu (\beta_i^{FR})^2$.

Algorithm 1. (The modified CG algorithm)

Step(1): Initiate $w_0, g_0 = E_G$ and $t_{max}$ (maximum number of epochs), set $i = 0$.

Step(2): Compute $E_i$ and $g_i = \nabla E(w_i)$.

Step(3): If $E_i < E_G$, or $\|g_i\| \leq \epsilon$, return $w^* = w_i$ and $E^* = E_i$ then stop

else Evaluate $s_i = w_{i+1} - w_i$ and $y_i = g_{i+1} - g_i$.

Step(4): Determine the descent direction using (1.3) and (2.3).

Step(5): Compute the learning rate $\lambda_i$ to minimize

$$f(w_i + \lambda_i d_i)$$

Step(6): Updating new point of the weights based on Equation (1.2) and set $i = i + 1$

Step(7): If $i > t_{max}$ return “Error Goal not met” else go to step 2.

3. THE DESCENT AND THE SUFFICIENT DESCENT CONDITIONS OF THE MODIFIED CG ALGORITHM

This section, show that the modified CG algorithm satisfies the descent and sufficient descent conditions as stated in the following theorems:

Theorem 3.1. Suppose that the sequence $\{w_i\}$ is created by (1.2). Then the search direction given by equations (1.3) and (2.3) satisfies the descent condition, i.e. $p_i^T g_{i+1} \leq 0$.

Proof: From (1.3), we have $i = 0$

$$p_i^T g_0 = -\|g_0\|^2 \leq 0$$

suppose that $p_i^T g_k \leq 0 \forall k = 1, 2, ..., i$.

Now, we prove the present search direction is descent direction at the iteration $(i + 1)$.

$$p_{i+1} = -g_{i+1} + \beta_{i}^{DY} p_i \quad \text{(3.1)}$$

implies that

$$p_{i+1} = -g_{i+1} \left(\beta_{i}^{DY} - \mu (\beta_i^{FR})^2\right) p_i \quad \text{(3.2)}$$

By multiplying equation (3.2) by $g_i^T$, we have $p_i^T g_{i+1} = -\|g_{i+1}\|^2 + \beta_{i}^{DY} g_i^T p_{i+1} - \mu (\beta_i^{FR})^2 g_i^T p_i \quad (3.3)$

If $p_i^T g_{i+1} = 0$, then the equation (3.3) is achieve the descent condition i.e.

$$g_i^T p_{i+1} = -\|g_{i+1}\|^2 \leq 0$$

However, if $p_i^T g_{i+1} \neq 0$. We conclude $\|g_{i+1}\|^2 + \beta_{i}^{DY} g_i^T p_{i+1} \leq 0$, (3.4)

because the DY method satisfies the descent condition.

Since $g_{i+1}^T p_i \leq p_i^T y_i$ and clearly $p_i^T y_i > 0$,

$$\mu \in (0,1]$$

and $(\beta_i^{FR})^2 \geq 0$

so, the third term of equation (3.3) can be written as

$$-\mu (\beta_i^{FR})^2 g_i^T d_i \leq -\mu (\beta_i^{FR})^2 p_i^T y_i \leq 0$$

Finally, we have

$$g_i^T p_{i+1} = -\|g_{i+1}\|^2 + \beta_{i}^{DY} g_i^T p_{i+1} - \mu (\beta_i^{FR})^2 g_i^T p_i \leq 0$$

Theorem 3.2. Suppose that $p_{i+1}$ is produced by equations (1.3) and (2.3), and $\lambda_i$ is obtained from equations (1.10) and (1.11), then the sufficient descent condition is satisfied, i.e.

$$g_i^T p_{i+1} \leq -c\|g_{i+1}\|^2$$
Proof: From equation (3.4). Therefore, the equation (3.3) can be written as follows:
\[ g_{i+1}^T p_{i+1} \leq -\mu \left( \frac{g_{i+1}^T g_{i+1}}{g_{i}^T} \right)^2 g_{i+1}^T p_i \]  
(3.5)
Since \( g_{i+1}^T p_i \leq p_i^T y_i \), will be in the form
\[ g_{i+1}^T p_{i+1} + c \leq -\mu \left( \frac{g_{i+1}^T g_{i+1}}{g_{i}^T} \right)^2 p_i^T y_i \| g_{i+1}^i \|^2 \]  
(3.6)
we obtained \( g_{i+1}^T p_{i+1} + c \leq -\mu \| g_{i+1}^i \|^2 \),
where \( c = \mu \left( \frac{g_{i+1}^T g_{i+1}}{g_{i}^T} \right)^2 p_i^T y_i \). ■

4. THE GLOBAL CONVERGENCE OF THE MODIFIED CG ALGORITHM

To prove the global convergence result of the modified CG method, we need the following assumptions. [11]
Assumption 1. The level set \( S = \{ w : w \in \mathbb{R}^n, E(w) \leq E(w_0) \} \) is bounded. i.e. \( \exists \ B > 0 \) such that
\[ ||w|| \leq B, \forall \ w \in S \]  
(4.1)
Assumption 2. In a neighborhood \( \Omega \in S \), \( E \) is differentiable and its gradient \( g \) is Lipschitz continuous, i.e. \( \exists L > 0 \) such that
\[ ||g(w) - g(w_i)|| \leq L||w - w_i||, \forall \ w, w_i \in \Omega \]  
(4.2)
From Assumptions 1 and 2, \( \exists \ M > 0 \) such that
\[ ||g(w)|| \leq M, \forall \ w \in S \]  
(4.3)
Lemma 4.1 [15]. Assume that the Assumptions 1 and 2 holds and the sequence \( \{ w_i \} \) is created by the equations (1.2) and (1.3), where \( p_i \) satisfy the descent condition and \( \lambda_i \) is determined by (1.10) and (1.11). If
\[ \sum_{i \geq 1} \frac{1}{||p_i||^2} = \infty, \]  
(4.4)
Then
\[ \lim_{i \to \infty} \inf ||g_i|| = 0. \]  
(4.5)
If \( E \) is a uniformly convex function, \( \exists \ \theta > 0 \) such that:
\[ (g(x) - g(y))^T (x - y) \geq \theta \|x - y\|^2 \in \Omega. \]  
(4.6)
We can rewrite (4.6) in the following manner:
\[ \gamma(s_i) \geq \theta \|s_i\|^2. \]  
(4.7)
Theorem 4.1. Assume that Assumptions 1 and 2 holds. If any of the iterations of the (1.2) and (1.3), where \( \beta_i^{\text{new}} \) is defined by equation (2.3) and \( \lambda_i \) satisfies the strong Wolfe line search conditions (1.10) and (1.11), then
\[ \lim_{i \to \infty} \inf ||g_{i+1}|| = 0. \]  
(4.8)
Proof: By using contradiction, we assume their exist apposite constant such that
\[ ||g_i|| \geq \omega, \forall \ i \geq 0. \]  
(4.8)
Then, from (1.3) and (2.3), it follows that:
\[ p_{i+1} = -g_{i+1} + \beta_i^{\text{new}} p_i \]
which is can be written as
\[ ||p_{i+1}|| \leq ||g_{i+1}|| + \beta_i^{\text{new}} ||p_i||, \]  
(4.9)
and \( \beta_i^{\text{new}} = \frac{g_{i+1}^T y_i^*}{p_i^T y_i} - \mu \left( \frac{g_{i+1}^T g_{i+1}}{g_{i}^T} \right)^2 \) using equation (4.7), we obtained that
\[ ||p_i^{\text{new}}|| \leq \frac{\gamma_i \|g_{i+1}\|^2}{\theta \|s_i\|^2} + \mu \left( \frac{g_{i+1}^T g_{i+1}}{g_{i}^T} \right)^2 \]  
(4.10)
Then
\[ ||p_i^{\text{new}}|| \leq \frac{\lambda_i M^2}{\theta \|s_i\|^2} + \mu M^2. \]  
(4.11)
By combining the equations (4.9) and (4.11), we have
\[ ||p_{i+1}|| \leq M + \left( \frac{\lambda_i M^2}{\theta \|s_i\|^2} + \mu M^2 \right) ||p_i||. \]  
(4.12)
Implies that
\[ ||p_{i+1}|| \leq M + \left( \frac{M^2}{\theta \|s_i\|^2} + \mu M^2 \right) ||s_i||. \]  
(4.13)
Since, \( ||s_i|| = ||w - w_i|| \),
\( D = \max \{ ||w - w_i|| \}, \forall \ w, w_i \in \mathbb{R} \).
Hence (4.13) becomes
\[ ||p_{i+1}|| \leq M + \left( \frac{M^2}{\theta D} + \mu M^2 \right) = \varphi, \]  
leading to (4.4). So, from Lemma 4.1. Hence (4.5) holds and contradicting (4.8).

EXPERIMENTAL RESULTS

In this section, we examine the implementation of the modified method. The comparative tests include familiar nonlinear problems with various dimensions 4 ≤ n ≤ 5000. Our algorithms has converged as shown so as ||g_{i+1}|| ≤ 10^{-5} and Powell condition ||g_{i+1}|| ≥ 0.2 ||g_{i}||^2 is used to restart. All algorithms implemented with a cubic interpolation which uses function and gradient values. The algorithms are written in FORTRAN 95 language. Table (1) shows that the numerical results of the modified (CG) method is more effective than standard (DY) method with respect to the number of iterations (NI) and the number of functions evaluation (NF).

In addition to that, we will offer experimental numerical results in order to study and assess the performance of the modified (CG) method in classical artificial intelligence problems (Continuous Function Approximation).

In particular, we investigate the performance of DY method compare with our modified method during five times of the implementation the program. The implementation has been carried out...
by using MATLAB (2013a) and the MATLAB Neural Network Toolbox version 8.1 for conjugate gradient.

5.1 Problem: (Continuous Function Approximation)
Consider the approximation of the continuous trigonometric function as:

\[ f(x) = \sin(x) + \cos(3x), \text{WHERE } x \in [0, \pi]. \]

The network is trained to approximate the function and the network is trained until the mean squares of the errors becomes less than the error goal 1e-10 within the limit of 1000 epochs.

Tables 3: offer the performance comparison of the methods DY and modified (CG) for the continuous function approximation problem. All algorithms display excellent likelihood (100%) of successful training for network using the same initial weights. Thus, computational cost is possibly the most appropriate indicator for measuring the efficiency of the methods. The performs of modified (CG) method is better than the DY method in terms of the number of epochs, time, Gradient and Step size.

### Table (1): Comparison between the (modified and DY) methods

<table>
<thead>
<tr>
<th>Test Function</th>
<th>N</th>
<th>CG (DY)</th>
<th>Modified (CG)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>NI</td>
<td>NF</td>
</tr>
<tr>
<td>Miele</td>
<td>4</td>
<td>36</td>
<td>115</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>45</td>
<td>156</td>
</tr>
<tr>
<td></td>
<td>500</td>
<td>53</td>
<td>188</td>
</tr>
<tr>
<td></td>
<td>1000</td>
<td>60</td>
<td>222</td>
</tr>
<tr>
<td></td>
<td>5000</td>
<td>66</td>
<td>257</td>
</tr>
<tr>
<td>Non-Diagonal</td>
<td>4</td>
<td>24</td>
<td>63</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>29</td>
<td>79</td>
</tr>
<tr>
<td></td>
<td>500</td>
<td>29</td>
<td>214</td>
</tr>
<tr>
<td></td>
<td>1000</td>
<td>29</td>
<td>79</td>
</tr>
<tr>
<td></td>
<td>5000</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>Fred</td>
<td>4</td>
<td>8</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>8</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td>500</td>
<td>8</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td>1000</td>
<td>8</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td>5000</td>
<td>8</td>
<td>22</td>
</tr>
<tr>
<td>Beal</td>
<td>4</td>
<td>11</td>
<td>28</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>12</td>
<td>30</td>
</tr>
<tr>
<td></td>
<td>500</td>
<td>12</td>
<td>30</td>
</tr>
<tr>
<td></td>
<td>1000</td>
<td>12</td>
<td>30</td>
</tr>
<tr>
<td></td>
<td>5000</td>
<td>12</td>
<td>30</td>
</tr>
<tr>
<td>Central</td>
<td>4</td>
<td>18</td>
<td>127</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>20</td>
<td>153</td>
</tr>
<tr>
<td></td>
<td>500</td>
<td>23</td>
<td>192</td>
</tr>
<tr>
<td></td>
<td>1000</td>
<td>23</td>
<td>192</td>
</tr>
<tr>
<td></td>
<td>5000</td>
<td>24</td>
<td>205</td>
</tr>
<tr>
<td>Sum</td>
<td>4</td>
<td>5</td>
<td>27</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>14</td>
<td>80</td>
</tr>
<tr>
<td></td>
<td>500</td>
<td>20</td>
<td>100</td>
</tr>
<tr>
<td></td>
<td>1000</td>
<td>27</td>
<td>132</td>
</tr>
<tr>
<td></td>
<td>5000</td>
<td>32</td>
<td>151</td>
</tr>
<tr>
<td>Osp.</td>
<td>4</td>
<td>8</td>
<td>44</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>52</td>
<td>180</td>
</tr>
<tr>
<td></td>
<td>500</td>
<td>138</td>
<td>439</td>
</tr>
<tr>
<td></td>
<td>1000</td>
<td>196</td>
<td>607</td>
</tr>
<tr>
<td></td>
<td>5000</td>
<td>555</td>
<td>1857</td>
</tr>
</tbody>
</table>
Note: The fail result in standard CG is considered a twice value of modified (CG) results.

Table (2): Percentage of improving the modified method

<table>
<thead>
<tr>
<th>Tools</th>
<th>CG (PR)</th>
<th>Modified (CG)</th>
</tr>
</thead>
<tbody>
<tr>
<td>NI</td>
<td>100%</td>
<td>88.8277%</td>
</tr>
<tr>
<td>NF</td>
<td>100%</td>
<td>89.6225%</td>
</tr>
</tbody>
</table>

As we observe from Table 2 the NI and NF of the DY method are about 100%. That means, the modified method has improvement of 11.1722% and 10.3775% compared with standard method in NI and NF respectively. Generally, the modified (CG) method was improved by 10.77485% compared with DY method.

Table (3): Comparing the Performance of modified method with Standard DY method for training neural network

<table>
<thead>
<tr>
<th>Methods</th>
<th>No. Running</th>
<th>Epochs</th>
<th>CPU time(s)/Epoch</th>
<th>Gradient Step size</th>
</tr>
</thead>
<tbody>
<tr>
<td>DY</td>
<td>1</td>
<td>1000</td>
<td>00:04</td>
<td>0.00383</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>1000</td>
<td>00:03</td>
<td>0.00149</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>1000</td>
<td>00:03</td>
<td>0.00277</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>1000</td>
<td>00:03</td>
<td>0.00167</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>1000</td>
<td>00:03</td>
<td>0.00345</td>
</tr>
<tr>
<td>Modified</td>
<td>1</td>
<td>191</td>
<td>00:19</td>
<td>0.00695</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>464</td>
<td>00:01</td>
<td>0.00152</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>904</td>
<td>00:03</td>
<td>0.00211</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>761</td>
<td>00:02</td>
<td>0.00565</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>273</td>
<td>00:01</td>
<td>0.0115</td>
</tr>
</tbody>
</table>

Fig (1): Performance of DY method for training neural networks
CONCLUSION

This paper proposed a modified (CG) method which consists of (Fletcher and Reeves) method and by using logistic mapping. The search direction \( p_t \) produced by our proposed method satisfies both (the descent and sufficient descent) conditions. The global convergence of the modified (CG) method has been proved. Furthermore, we used the modified (CG) method for training neural networks. Depend on the numerical experiments, we found that modified method is more effective than the classical CG method, leading to a stable and faster convergence.

REFERENCE


