A Preconditioned Unconstrained Optimization Method for Training Multilayer Feed-Forward Neural Network

Khalil K. Abbo[1], Zahra Abdlkareem[2]

[1]Department of Mathematics,
Mosul University, Mosul, Iraq
prof_khalil@uomosul.edu.iq

[2]Department of Mathematics,
Mosul University, Mosul, Iraq
Zozokareem26689@gmail.com

Received: 13.12.2018, Accepted: 24.02.2020, Published: 24.02.2020

Abstract — *Non-linear unconstrained optimization methods constitute excellent neural network training methods characterized by their simplicity and efficiency. In this paper, we propose a new preconditioned conjugate gradient neural network training algorithm which guarantees descent property with standard Wolfe condition. Encouraging numerical experiments verify that the proposed algorithm provides fast and stable convergence.*

Keywords: Unconstrained optimization, Neural network, Descent property, Wolfe condition.
Mathematics Subject Classification: 80C50, 30A40, 90C26.

1 Introduction

The batch training of the multi-layer feed-forward neural network (MFN) can be formulated as non-linear unconstrained minimization problem [4,11,15,16,17]. Namely

$$\min\ E(w_k),\ w_k \in R^n\ , \tag{1}$$

where w_k is the weight vector, E is the batch error measure defined as the sum of squared differences error functions over the entire training set, defined by

$$E(w) = \frac{1}{2}\sum_{p=1}^{P}\sum_{j=1}^{N_M}(O_{j,p}^{M} - T_{j,p})^2 \tag{2}$$

where $(O_{j,p}^{M} - T_{j,p})^2$ is the squared differences between the actual j-th output layer neuron for pattern P and the target output value. The scalar P is an index over input-output pairs, the general purpose of the training is to search an optimal set of connection weights w in the manner that the error of the network output can be minimized. The weight update equation for any training algorithm has the iterative form:

$$w_{k+1} = w_k + \alpha_k d_k \tag{3}$$

Where $w_1 \in R^n$ is a given starting point, α_k is a learning rate or(step size) with $\alpha_k \geq 0$, and d_k is a search direction which satisfies $g_k^T d_k < 0$.

The gradient $g_k = \nabla E(w_k)$ easily can be obtained by means of back-propagation of errors through the layers. Many minimization methods have been applied to the training MFN. Examples include back-propagation(BP) [9,10,21], conjugate gradient [7,14,18] and quasi-Melton's method [1,3,4]. BP method minimizes the error function using the steepest descent, namely $d_k = -g_k$, with fixed (heuristically chosen) step-size [21]. The standard BP algorithm which often behaves very badly on large-scale problems and which success depends of the user dependent parameters learning rate [5,13].

Conjugate gradient methods and quasi-newton methods that are applicable to large-scale problems. The conjugate gradient methods are iterative methods of the form (3) with search direction defined by :

$$d_{k+1} = -g_{k+1} + \beta_k d_k \tag{4}$$

where $d_1 = -g_1$, β_k is a scalar, some famous formulae for β_k are:

$$\beta_{k+1}^{FR} = \frac{\|g_{k+1}\|^2}{\|g_k\|^2} \quad , \beta_{k+1}^{HS} = \frac{g_{k+1}^T y_k}{d_k^T y_k} \quad , \quad \beta_{k+1}^{PRP} = \frac{g_{k+1}^T y_k}{\|g_k\|^2}, \tag{5}$$

here $y_k = g_{k+1} - g_k$, (FR) Fletcher-Reeves method [8], (HS) denotes the Hestenes and Stiefel [12], (PR) denotes the Polak and Ribiere [20].

Recently, Birgin and Matinez [6] proposed a spectral conjugate gradient method by combining conjugate gradient method and spectral method in the following way:

$$d_{k+1}^{SCG} = -\theta_k g_{k+1} + \beta_k s_k, \tag{6}$$

where θ_k is parameter. In the Quasi-Newton methods [2] the search directions define by:

$$d_{k+1}^{QN} = -H_{k+1} g_{k+1} \tag{7}$$

where $H_1 = I_{n*n}$ is the identity matrix and H_k updated by rank-one for example symmetric rank-one is

$$H_{k+1}^{SR1} == H_k + \frac{(s_k - H_k y_k)(s_k - H_k y_k)^T}{y_k^T (s_k - H_k y_k)} \tag{8}$$

or rank-2 such as DFP method

$$H_{k+1}^{DFP} == H_k + \frac{s_k s_k^T}{y_k^T s_k} - \frac{H_k y_k y_k^T H_k}{y_k^T H_k y_k} \tag{9}$$

(Daved-Fletcher-Powell) see [2].

The learning rate parameter α_k must be determined by some one dimensional line search along the search direction d_k. In our method, the learning rate achieved by the line search technique, must satisfy the standard Wolfe (WC) [22] conditions given by

$$E(w_{k+1}) \le E(w_k) + \rho \alpha_k g_k^T d_k$$

$$g_{k+1}^T d_k \le \sigma g_k^T d_k$$

or strong Wolfe conditions (SWC)

$$E(w_{k+1}) \le E(w_k) + \rho \alpha_k g_k^T d_k$$

$$\left| g_{k+1}^T d_k \right| \ge \sigma g_k^T d_k .$$

This paper is organized as follows. In Section (2) new preconditioned training conjugate gradient method suggested. The sufficient descent condition are presented in section (3). In section (4) global convergence of the new preconditioned conjugate gradient method established. Numerical results are reported in section (5).

2 Derivation of new preconditioned conjugate gradient method

In the quasi-Newton methods, an approximation matrix H_k of the invers Hessian $\nabla^2 E(w_k)$ is updated such that the new matrix H_{k+1} satisfies the following quasi-Newton equation[2] :

$$H_{k+1} y_k = s_k \tag{10}$$

where $s_k = x_{k+1} - x_k$.

In order to determine the new search direction we use the following quadratic equation:

$$Q = \left\| d^{QN} - d^{SCG} \right\|^2 \tag{11}$$

Subject to $\theta_k > 0$ and H_k is symmetric and positive definite. With simple algebra we get

$$\frac{\partial Q}{\partial \theta_k} = -2 g_{k+1}^T H_{k+1} g_{k+1} + 2\theta_k \|g_{k+1}\|^2 - 2\beta_k g_{k+1}^T s_k = 0 \tag{12}$$

resulting in:

$$\theta^* = \frac{g_{k+1}^T H_{k+1} g_{k+1} + \beta_k g_{k+1}^T s_k}{\|g_{k+1}\|^2} \tag{13}$$

and

$$\frac{\partial Q}{\partial \beta_k} = 2 g_{k+1}^T H_{k+1} s_k - 2\theta^*{}_k g_{k+1}^T s_k + 2\beta_k \|s_k\|^2 = 0 \tag{14}$$

From equations (13) and (14) we get:

$$\beta^*_k = \frac{g_{k+1}^T H_{k+1} g_{k+1} (g_{k+1}^T s_k) - g_{k+1}^T H_{k+1} s_k (\|g_{k+1}\|^2)}{\|g_{k+1}\|^2 (\|s_k\|^2 - (g_{k+1}^T s_k)^2)} \tag{15}$$

Therefor the new (PKZ say) search direction can be defined as

$$d_{k+1}^{PKZ} = -\theta^* g_{k+1} + \beta_k^* s_k \tag{16}$$

We summarize the above method as the following algorithm:

2.1 Algorithm (The PKZ algorithm)

Step(0): Given an initial starting point $w_1 \in R^n$ and $\varepsilon = 10^{-6}$,

consider $d_1 = -g_1, \alpha_1 = \frac{1}{\|g_1\|}$, and $k = 1$.

Step(1): Test for convergence, If $\|g_k\| < \varepsilon$, stop w_k is optimal. Else go to step 2.

Step(2): Compute α_k satisfying the Wolfe line search and update the variable $w_{k+1} = w_k + \alpha_k d_k$ and compute f_{k+1} , g_{k+1} , y_k and s_k.

Step(3): Direction computation : compute θ^*, β_k^* and d_{k+1}^{PKZ} from (13) , (14) and (16). If Powell restart is satisfied then $d_{k+1} = -\theta_k^* g_{k+1}$, else $d_{k+1} = d^{PKZ}$, compute initial guess for $\alpha_{k+1} = \alpha_k \left(\frac{\|d_k\|}{\|d_{k-1}\|} \right)$ and set $k = k+1$, go to step 1.

3 The Descent Property and Descent Algorithm

Since the conjugate gradient methods belong to the descent methods for solving unconstrained optimization problems, the new method should be chosen such that $g_{k+1}^T d_{k+1} \le 0$ if a line search is used. Furthermore, the sufficient descent condition is

$$g_{k+1}^T d_{k+1} \le -c\|g_{k+1}\|^2 \quad \text{for} \quad k \ge 0 \text{ and } c > 0 \tag{17}$$

Theorem 3.1 Suppose θ^* is positive parameter and step size α_k satisfies (WC or SWC) Wolfe conditions, then the search directions generated by (16)are descent directions for all k.

Proof. By induction, for initial direction (k=0) we have :

$$d_1 = -g_1 \Rightarrow \quad d_1^T g_1 = -\|g_1\|^2 < 0 \tag{18}$$

Now let the theorem be true for all k, i.e.

$$d_k^T g_k < 0. \tag{19}$$

To complete the proof, we have to show that the theorem is true for all $k+1$. Multiplying (16) by g_{k+1}^T, we have

$$g_{k+1}^T d_{k+1}^{PKZ} = -\theta_{k+1}^* \|g_{k+1}\|^2 + \beta_k^* g_{k+1}^T s_k$$
$$= -\left(\frac{g_{k+1}^T H_{k+1} g_{k+1} + \beta_k^* g_{k+1}^T s_k}{\|g_{k+1}\|^2} \right) \|g_{k+1}\|^2 + \beta_k^* g_{k+1}^T s_k . \qquad (20)$$

Since H_{k+1} is a positive definite matrix, namely,

$$g_{k+1}^T H_{k+1} g_{k+1} > 0 \qquad (21)$$

from (20) and (21) we get :

$$g_{k+1}^T d_{k+1}^{PKZ} = - g_{k+1}^T H_{k+1} g_{k+1} < 0 . \qquad (22)$$

4 Global convergence property

In order to establish the global convergence of the proposed method. We assume that the following assumption always holds.

Assumption 1

i- The level set $S = \{ w \in R^n : E(w) \le E(w_1) \}$ is bounded, namely, there exists a constant $B > 0$ such that

$$\|w\| \le B \quad \text{for all} \quad w \in S . \qquad (23)$$

ii- In some neighbourhood N of S, E is continuously differentiable, and its gradient is Lipschitz continuous, namely, there exist $L > 0$ such that:

$$\|g(x) - g(y)\| \le L \|x - y\| \quad \forall \mathrm{x}, \mathrm{y} \in \mathrm{N} . \qquad (24)$$

Lemma 1 Suppose Assumption(1) holds. Consider any iteration of (3) and(4), where d_k satisfies $g_k^T d_k < 0$ for $k \in N^+$ and α_k satisfies the Wolf line search conditions (WC or SWC) . Then

$$\text{If} \quad \sum_{k \ge 1} \frac{1}{\|d_k\|^2} = \infty \quad \text{then } Lim \inf \|g_k\| = 0 \quad \text{as } k \to \infty \qquad (25)$$

More details can be found in [23].

Now, we give the following Theorem of global convergence for the PKZ conjugate gradient method.

Theorem 4.1 Suppose that Assumption 1. holds and E is strongly convex. If $\{w_k\}$ obtained by PKZ algorithm, then we have either $g_{k+1} = 0$ for some k or

$$\lim_{k\to\infty}\inf\|g_k\| = 0 \quad . \tag{26}$$

Proof. If $\|g_{k+1}\| \neq 0$ for all k, then there exists $\mu > 0$ such that $\|g_{k+1}\| > \mu$. Now by the equations(13), (15) and (16), we obtain :

$$\|d_{k+1}\| \le \frac{\|g_{k+1}\|^2}{\|g_{k+1}\|^2}\|g_{k+1}\| + \frac{|\beta_k|\|g_{k+1}\|\|s_k\|}{\|g_{k+1}\|^2}\|g_{k+1}\| + |\beta_k|\|s_k\|$$
$$\le \|g_{k+1}\| + 2|\beta_k|\|s_k\|.$$

From (15) we get :

$$|\beta_k| \;=\; \left|\frac{g_{k+1}^T H_{k+1} g_{k+1}(g_{k+1}^T s_k) - g_{k+1}^T H_{k+1} s_k(\|g_{k+1}\|^2)}{\|s_k\|^2\|g_{k+1}\|^2 - (g_{k+1}^T s_k)^2}\right|$$
$$\le \left|\frac{g_{k+1}^T H_{k+1} g_{k+1}(g_{k+1}^T s_k) - g_{k+1}^T H_{k+1} s_k(\|g_{k+1}\|^2)}{\|s_k\|^2\|g_{k+1}\|^2}\right|$$
$$\le \left|\frac{g_{k+1}^T H_{k+1} g_{k+1}(g_{k+1}^T s_k)}{\|s_k\|^2\|g_{k+1}\|^2}\right| + \left|\frac{g_{k+1}^T H_{k+1} s_k(\|g_{k+1}\|^2}{\|s_k\|^2\|g_{k+1}\|^2}\right|$$
$$\le \frac{\|g_{k+1}\|^2\;\|g_{k+1}\|\;\|s_k\|}{\|s_k\|^2\|g_{k+1}\|^2} + \frac{\|g_{k+1}\|\;\|y_k\|\;\|g_{k+1}\|^2}{\|s_k\|^2\|g_{k+1}\|^2}.$$

Utilizing Lipschtiz condition we have :

$$|\beta_k| \;\le\; \frac{\|g_{k+1}\|^2\;\|g_{k+1}\|\;\|s_k\|}{\|s_k\|^2\|g_{k+1}\|^2} + L\frac{\|g_{k+1}\|\;\|s_k\|\;\|g_{k+1}\|^2}{\|s_k\|^2\|g_{k+1}\|^2}$$
$$\le (1+L)\frac{\|g_{k+1}\|}{\|s_k\|}.$$

Therefore, we get :

$$\|d_{k+1}\| \le \|g_{k+1}\| + 2(1+L)\frac{\|g_{k+1}\|}{\|s_k\|}\|s_k\|$$
$$\le [1 + 2(1+L)]\;\|g_{k+1}\|$$
$$\sum_{k=1}^{\infty}\frac{1}{\|d_k\|} > \sum_{k=1}^{\infty}\frac{1}{(3+3L)*\mu} = \infty$$

Therefore by lemma 1 we get

$$Lim\inf\|g_k\| = 0$$

which completes the proof.

5 Experiments and Results

A computer simulation has been developed to study the performance of the learning algorithms. The simulations have been carried out using MATIAB(7.6) the performance of the PKZ has been evaluated and compared with the following batch training algorithms:

1-Clasical Backpropagation (CPB).

2-Daved-Fletcher-Powell (DFP) Quasi- Newton algorithm.

3-Polak-Ribier conjugate gradient(PRCG) method.

The algorithms were tested using the initial weights, initialized by the Nguyen –widrow method [19] and received the same sequence of input patterns. The weights of network are updated only after the entire set of patterns to be learned has been presented.

For each of the test problems, a table summarizing the performance of the algorithms for simulations that reached solution is presented. The reported parameters are min the minimum number of epochs for 100 simulation, mean the mean value of epochs for 100 simulation, Max the maximum number of epochs for 100 simulation, Tav the average of total time for 100 simulations and Such, the succeeded simulations out of 100 trails within error function evaluations limit. If an algorithm fails to converge within the above limit considered that it fails to train the MFN, but its epoch are not included in the statically analysis of the algorithm, one gradient and one error function evaluations are necessary at each epoch.

Problem 1 (Continuous Function Approximation)

The first test problem we consider is the approximation of the continuous trigonometric function: $f(x) = \sin(x) * \cos(3x)$. The network architecture for this problem is 1-15-1 FNN (thirty weights, sixteen biases) is trained to approximate the function f(x), where x∈[-π,π] and the network is trained until the sum of the squares of the errors becomes less than the error goal 0.0005, comparative results are shown in Table 1.

Table 1: Results of simulations for the function approximation problem

Algorithms	Min	Max	Mean	Tav	Succ
CBP	fail	--	--	--	0.0%
DFP	31	459	98.97	4.86	100%
PR	60	529	115.27	4.96	85%
PKZ	36	475	95.13	4.83	100%

Form Table 1, we conclude that the algorithm PKZ is the best algorithm with simulations accuracy and success, number of epochs and the time.

Problem 2 (XOR Problem)

The second problem we have been encountered is the XOR Boolean function problem, which is considered as a classical problem for the MFN training. The XOR function maps two binary inputs to a single binary output. As it is well known this function is not linearly separable. The network architectures for this binary classification problem consists of one hidden layer with 3 neurons and an output layer of one neuron. The termination criterion is set to $\varepsilon_2 \leq 0.0005$ within limit of 1000 epochs, Table 2 summarizes the result of all algorithms i.e for 100 simulations the minimum epoch for each algorithm are listed in the first column (Min), the maximum epoch for each algorithm are listed in the second column, third column contains (Mean) the mean value of epochs and (Tav) is the average of time for 100 simulations and last columns contains the percentage of succeeds of the algorithms in 100 simulation.

Table 2: Results of simulations for the XOR function

Algorithms	Min	Max	Mean	Tav	Succ
CBP	205	1000	573	4.8966	98%
DFP	5	28	7.44	2.318	100%
PR	7	31	9.61	2.702	100%
PKZ	7	30	8.22	2.411	100%

Form Table 2, we conclude that the algorithm PKZ is the best algorithm with simulations accuracy and success, number of epochs and the time.

References

[1] L. E. Achenie, Computational experience with a quasi Newton method based training of feed-forward neural networks, Proc. World Congress on Neural Networks, San Diego, 1994, III607-III612.

[2] A. Andreas and S. Wu, Practical optimization algorithms and engineering applications, Springer US, 2007, 1-26.

[3] R. Battiti, First-and second-order methods for learning: between steepest descent and Newton's method, Neural Comput., 4(2) 1992, 141-166.

[4] R. Battiti and F. Masulli, BFGS optimization for faster and automated supervised learning, In International neural network conference, Springer, Dordrecht, 1990 .

[5] C. M. Bishop, Neural networks for pattern recognition. Oxford university press, 1995.

[6] E. Birgin and J. Martinez, A spectral conjugate gradient method for unconstrained optimization, Appl. Math. Opt., 43(2) 2001, 117-128.

[7] C. Charalambous, Conjugate gradient algorithm for efficient training of artificial neural networks, “IEE Proceedings G (Circuits, Devices and Systems)”, 139(3) 1992, 301-310.

[8] R. Fletcher and C.M. Reeves, Function minimization by conjugate gradients, Comput. J., 7(2) 1964, 149-154.

[9] L. Gong, C. Liu, Y. Li and Y. Fuqing, Training feed-forward neural networks using the gradient descent method with the optimal stepsize. Journal of Computational Information Systems, 8(4) 2012, 1359-1371.

[10] S. Haykin, Neural networks: a comprehensive foundation, Prentice Hall PTR, 1994.

[11] J. Hertz, A. Krogh and RG. Palmer, Introduction to the theory of neural computation, Addison Wesley, Longman, 1991.

[12] M.R. Hestenes and E. Stiefel, Methods of conjugate gradients for solving linear systems, Washington, 1952.

[13] R.A. Jacobs, Increased rates of convergence through learning rate adaptation, Neural networks, 1(4) 1988, 295-307.

[14] K. Abbo and M. Hind, Improving the learning rate of the Back-propagation Algorithm Aitken process, Iraqi J. Of Statistical Sci., 2012.

[15] A. E. Kostopoulos, D. G. Sotiropoulos and T. N. Grapsa, A new efficient variable learning rate for Perry's spectral conjugate gradient training method, 2004.

[16] I. E. Livieris and P. Pintelas, An advanced conjugate gradient training algorithm based on a modified secant equation, ISRN Artificial Intelligence, 2011.

Khalil K. Abbo, ORCID: https://orcid.org/0000-0001-5858-625X
Zahra Abdlkareem, ORCID: https://orcid.org/0000-0001-9827-2634