ISSN: 1813 – 1662 (Print)
E-ISSN: 2415 – 1726 (On Line)

TJPS

Tikrit Journal of Pure Science

Journal Homepage: http://tjps.tu.edu.iq/index.php/j

Generalized Dai-Yuan conjugate gradient algorithm for training multi-layer feed-forward neural networks

Hind H. Mohammed

Department Of Math. College Of Computer Sciences And Mathematics, University Of Mosul , Mosul , Iraq

https://doi.org/10.25130/tjps.v24i1.341

ARTICLE INFO.

Article history:

-Received: 17 / 12 / 2017
-Accepted: 4 / 1 / 2018
-Available online: / / 2019

Keywords: Conjugate Gradient, Artificial Neural Networks, Back-Propagation, Dai-Yuan, Training, Optimization

Corresponding Author:
Name: Hind H. Mohammed
E-mail: hindmth80@gmail.com
Tel:

ABSTRACT

In this paper, we will present different type of CG algorithms depending on Peary conjugacy condition. The new conjugate gradient training (GDY) algorithm using to train MFNNs and prove it's descent property and global convergence for it and then we tested the behavior of this algorithm in the training of artificial neural networks and compared it with known algorithms in this field through two types of issues.

1. Introduction

Recently, the artificial neural networks (ANNs) plays an important area of the studies besides it has been utilized in several applications of AI (artificial intelligence). ANNs attracted the attentions of many searchers more than the scientific community because of the excellently capacity of self-adapting and self-learning. These days, ANNs are considered as a great implement using with pattern classification, it has been determined as powerful modules to many systems [1].

Clearly, there is a strong relation between the unconstrained optimization theory and the processing of learning in the ANN. Mathematically, the training process can be expressed as a minimization of the error function $E(w)$ using offline version which depends on the connection between the layer of the network (i.e. weights w). Therefor the training of neural network became as iteratively process to adjusting the weights of the network defined as the sum of squares of the errors in the outputs [2] that is:

$$E(w)=\frac{1}{2}\sum_{j=1}^{P}\sum_{i=1}^{M}(O_i^{(j)}-T_i^{(j)})^2 \quad (1)$$

Where O_i is the actual output and T_i is the target(desired) of the i-th neuron. The index j signifies the specific training pattern.

Where O_i and T_i are the desired (target) and the actual output of the i-th neuron, respectively. The index j denotes the particular learning pattern. The vector $w \in R^n$ is consist of all the weights of the network [3]. Let us define the gradient of the error function as $[g_k=\nabla E(w_k)]$. The batch back-propagation (BP) update is a form of gradient descent defined as

$$w_{k+1}=w_k-\alpha_0 g_k \qquad k=1,2,... \quad (2)$$

Where α_0 is the learning rate (step-size) which is constant at each iteration.

The back-propagation (BP) or gradient algorithms are one of the greatest used error minimization methods used to training multi-layer feed-forward neural networks (MFNN) [4]. Clearly, the gradient descent is a local optimization method which has backward error correction to the weights of the networks. Although, the common succeeding of the Back-Propagation in training the NNs but the deficiencies of this method are still required to solve. At the beginning, the Back-Propagation algorithm will trap in the local minima especially with non-linear separable problems[1]. Which may be lead to fail in the finding of a global optimal solution. As well as, the rate of convergence of Back-Propagation still very slow even if the training can be attained. Besides, the behavior of convergence to this algorithm depends largely on the choices of initial connection weights and other parameters in the algorithm such as momentum and the learning rate.

The improving of the training efficiency of a NNs run large area of papers and several research have been introduced in the literature[4].

Bishop in [1] recapitulated several optimization techniques were proposed to improve the activity of learning efficiency i.e. the error minimization manner. Among these optimization algorithms[5], the non-linear conjugate gradient (CG) method, which is generated sequence of weights $\{w_k\}_{k=1}^{\infty}$ with initial weights $w_1 \in R^n$. Firstly set $d_1 = -g_1$ and then a sequence w_k of the approximations to minimize is defined in the way:

$$w_{k+1} = w_k + \alpha_k d_k \qquad k = 1,2,... \quad (3)$$

$$d_{k+1} = -g_{k+1} + \beta_k.d_k \quad (4)$$

The conjugate gradient algorithms have various types according to the way of predefining the parameter β_k. As we known, there have been suggested several selections for β_k which belongs to different conjugate gradient ways.

$$\beta^{FR} = \frac{g_{k+1}^T\, g_{k+1}}{g_k^T\, g_k} \text{ [6]} \qquad \beta^{PR} = \frac{g_{k+1}^T\, y_k}{g_k^T\, g_k} \text{ [7]}$$

$$\beta^{DY} = \frac{g_{k+1}^T\, g_{k+1}}{d_k^T\, y_k} \text{ [8]} \qquad \beta^{HS} = \frac{g_{k+1}^T\, y_k}{d_k^T\, y_k} \text{ [9]}$$

where $y_k = g_{k+1} - g_k$. Often, to analysis the convergence of CG algorithms needs the inexact line search such as the standard Wolfe line search which defined by:

$$\mathrm{E}(\mathrm{w}_k + \alpha_k.d_k) \le \mathrm{E}(\mathrm{w}_k) + \rho.\alpha_k\, g_k^T d_k \quad (5)$$

$$\mathrm{g}(\mathrm{w}_k + \alpha_K d_k)^{\mathrm{T}} d_k \ge \sigma \mathrm{g}_k^{\mathrm{T}} \mathrm{d}_k \quad (6)$$

While the strong condition of Wolfe line search are:

$$\mathrm{E}(\mathrm{w}_k + \alpha_k.d_k) \le \mathrm{E}(\mathrm{w}_k) + \rho.\alpha_k\, g_k^T d_k \quad (7)$$

$$|g_{k+1}.d_k| \le -\sigma.g_k\, d_k \quad (8)$$

If $0 < \rho < \sigma < 1$.Other consequential subject in the CG methods is the search directions produced by equation (4) are conjugate directions if the line search is exact and the objective function is convex that is:

$$d_i^T G\, d_j = 0 \text{ , } if \ i \ne j \quad (9)$$

The variable G is devoted to the Hessian matrix of the objective function, it's clear that the equation (9) can be rewritten in the following form: $d_{k+1}^T\, y_k = 0$ (10)

which is called pure conjugacy condition. Perry in [10] generalized the conjugacy condition (10) for non-quadratic functions to the:

$$d_{k+1}^T\, y_k = -\, g_{k+1}^T\, s_k \quad (11)$$

where $s_k = w_{k+1} - w_k$. Dai and Liao in [11] showed that for general objective function with inexact line search the equation (11) can be written as follows:

$$d_{k+1}^T\, y_k = -\xi\, g_{k+1}^T\, s_k \text{ , } \xi > 0 \quad (12)$$

This paper is planned as follows: In section 2, the novel conjugate gradient teaching (GDY) algorithm for MFNNs presented. Section 3 presents the descent property and global convergence for the proposed algorithm. while section 4 include the experimental results to appraisal the performance of the proposition training method and paragons it with famous training methods.

2. Derivation of the new GDY method

An advantage of Dai-Yuan method(β^{DY}) is that it ensures the direction is descent and hence the global convergence is achieved when implemented with the conditions of Wolfe line search. In the other hand the β^{DY} method has drawback similar to the β^{FR} method, so it cannot get out the cycles of the teeny steps, to overcome to this drawback , I suggest a new formula β^{GDY} as follows: Consider the following search direction:

$$d_{k+1} = -g_{k+1} + [\frac{g_{k+1}^T\, g_{k+1}}{s_k^T\, y_k} - \gamma \frac{s_k^T\, g_{k+1}}{s_k^T\, y_k}] s_k \quad (13)$$

Now to find the value of γ , multiply both side of equation (13) by y^T

to get:

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

From the equation(14) and Perry conjugacy condition (11)we get:

$$-\, y_k^T g_{k+1} + g_{k+1}^T g_{k+1} - \gamma\, s_k^T g_{k+1} = -s_k^T g_{k+1}$$

$$\Rightarrow \gamma\, s_k^T g_{k+1} = g_{k+1}^T g_{k+1} - (g_{k+1}^T g_{k+1} - g_k^T g_{k+1}) + s_k^T g_{k+1}$$

Therefore the value of γ is:

$$\gamma = 1 + \frac{g_{k+1}^T\, g_k}{s_k^T\, g_{k+1}} \quad (15)$$

Substitute equation (15) in (13) to get:

$$\therefore d_{k+1} = -g_{k+1} + [\frac{\|g_{k+1}\|^2}{s_k^T\, y_k} - (\frac{s_k^T\, g_{k+1}}{s_k^T\, y_k} + \frac{g_{k+1}^T\, g_k}{s_k^T\, y_k})] s_k \quad (16)$$

Or $\therefore d_{k+1} = -g_{k+1} + \beta_k s_k$ (17)

At this point, we present a high level description of the proposed (GDY) algorithm:

The Algorithm(GDY)

Step0: Initiate w_1, and choose σ, ρ such that $0 < \rho < \sigma < 1$,
$E_G, \varepsilon \le 10^{-5}$ & $\mathrm{K}_{\max}$,set $\mathrm{k} = 1$.

Step 1: Compute E_k & g_k (i.e. the error function value and its gradient).

Step 2: IF $\|g_k\| < \varepsilon\, or (E_k < E_G)$, set $w^{goal} = w_k$ and $E^{goal} = E_k$, print the goal is meet then stop .

Step 3: Calculate the descent direction: $d_k = -g_k$ when $k = 1$ then go to

Step 5, Else $d_{k+1} = -g_{k+1} + \beta_{k+1}^{GDY} s_k$.

Step4: Calculate α_k (the learning rate) using the standard conditions of Wolfe (5) and (6).

Step 5: Updating weights: $w_{k+1} = w_k + \alpha_k d_k$

and put $k = k+1$.

Step 6: Stop if $k > k_{\max}$ and print the Error goal not reach, otherwise return to step 1.

3. Convergence analysis

The descent condition outplays an chief role in the global convergence analysis to many methods. The next theorem signifies that the algorithm(GDY) is produced a descent directions.

Theorem(1)

Consider the search directions generated by the equation(16) and assume that the learning rate α_k satisfies the Wolfe condition (5) and (6), then $d_k^T g_k < 0 \quad \forall k \ge 1$

Proof :

Clearly the theorem is true when $k=1$ and suppose the theorem is true at k ,we shall show it is true at $k = k+1$. Consider the search direction defined by equation (16)

$$d_{k+1} = -g_{k+1} + [\frac{\|g_{k+1}\|^2}{y_k^T s_k} - (\frac{g_{k+1}^T s_k}{y_k^T s_k} + \frac{g_{k+1}^T g_k}{y_k^T s_k})]s_k$$

Multiply both sides of the above equation by g_{k+1} get:

$$d_{k+1}^T g_{k+1} = -\|g_{k+1}\|^2 + [\frac{\|g_{k+1}\|^2}{y_k^T s_k} - (\frac{g_{k+1}^T s_k}{y_k^T s_k} + \frac{g_{k+1}^T g_k}{y_k^T s_k})]s_k^T g_{k+1}$$

$$= -\|g_{k+1}\|^2 + [g_{k+1}^T g_k - g_{k+1}^T s_k - g_{k+1}^T g_k)]\frac{s_k^T g_{k+1}}{y_k^T s_k}$$

$$= -\|g_{k+1}\|^2 + [g_{k+1}^T y_k - g_{k+1}^T s_k]\frac{s_k^T g_{k+1}}{y_k^T s_k}$$

use Lipschitz condition with $0 < L \le 1$ and $y_k^T s_k > 0$

To get

$$d_{k+1}^T g_{k+1} \le -\|g_{k+1}\|^2 + [L g_{k+1}^T s_k - g_{k+1}^T s_k]\frac{s_k^T g_{k+1}}{y_k^T s_k}$$

$$\therefore d_{k+1}^T g_{k+1} \le -\|g_{k+1}\| + (L-1)\frac{(s_k^T g_{k+1})^2}{s_k^T y_k} < 0 \blacksquare$$

Toward analyze the globally convergence property of the proposed algorithm(GDY), the next assumption are required. These assumptions have been used extensively in the literature for the global convergence analysis of conjugate gradient methods.

Assumption H

1- Let set $\psi = \{w \in R^n \mid E(w) \le E(w_1)\}$ then the level ψ is bounded, that is, $\exists$ a constant $B > 0$ satisfying:

$\|w\| \le B \qquad for\, all\ \mathrm{w} \in N$ (18)

2- Suppose that f is continuously differentiable function and its gradient is Lipschitz continuous in some neighborhood N of ψ ,i.e. $\exists$ a positive L such that $\|g(w_1) - g(w_2)\| \le L\|w_1 - w_2\|, \forall w_1, w_2 \in N$ (19)

Obviously, Assumption H denotes that $\exists$ a scaler γ such that

$\|g(w)\| \le \gamma \qquad \forall\ w \in N$ (20)

Now, we can obtain the next general result to any CG method satisfied strong Wolfe line search, which is attained in [1].

Lemma(1).

Assume that Assumption H satisfies. Consider a CG method in the form (3)-(17) such that α_k is computed from the strong Wolfe line search and d_k is a descent direction, if: $\sum_{k\ge1} \frac{1}{\|d_k\|^2} = \infty$ (21)

we have that $\lim_{k\to\infty} \inf(\|g_k\|) = 0$ (22)

For any uniformly convex function, we can verify that $\|d_k\|$ which generated by(17)is restricted from above. Therefore by using Lemma(1) we get the next result.

Theorem(2).

Assume that Assumption H is hold. Consider the direction search (16), where d_k is satisfied descent condition (i.e. $g_k^T d_k < 0$) and α_k is computed using strong Wolfe line search . If $\exists$ constant $\mu > 0$ satisfying:

$(\nabla f(w) - \nabla f(z))^T (w - z) \ge \mu\|w - z\|^2, \forall w, z \in \Omega$ (23)

We have $\lim_{k\to\infty} \|g_k\| = 0$ (24)

Proof:

We will demonstrate this theory using the method of contradiction, take the norm to the equation (17) to get:

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

Since $s_k^T y_k > 0$ by Wolfe condition then:

$$|\beta_k| \le \frac{\|g_{k+1}\|^2}{s_k^T y_k} + \frac{\|s_k\|\|g_{k+1}\|}{s_k^T y_k} + \frac{\|g_{k+1}\|\|g_k\|}{s_k^T y_k} \qquad (26)$$

Now, to the both sides of equation (16) take the norm to get:

$$\|d_{k+1}\| = \|-g_{k+1} + \beta_k s_k\| \Rightarrow \|d_{k+1}\| \le \|g_{k+1}\| + |\beta_k|\ \|s_k\| \qquad (27)$$

And offset the equation (26) in (27) get:

$$\|d_{k+1}\| \le \|g_{k+1}\| + \left[\frac{\|g_{k+1}\|^2}{s_k^T y_k} + \frac{\|s_k\|\|g_{k+1}\|}{s_k^T y_k} + \frac{\|g_{k+1}\|\|g_k\|}{s_k^T y_k} \right] \|s_k\| \qquad (28)$$

Substitute equation (20) in (28) to get:

$$\|d_{k+1}\| \le \gamma + \left[\frac{\gamma^2}{s_k^T y_k} + \frac{\|s_k\|\gamma}{s_k^T y_k} + \frac{\gamma^2}{s_k^T y_k} \right] \|s_k\| \qquad (29)$$

By using equation(23) for w_{k} and $w_{\mathrm{k+1}}$ we have:

$s_k^T y_k \le L\|s_k\|^2$ (30)

Therefore (29) become:

$$\|d_{k+1}\| \le \gamma + \left[\frac{\gamma^2}{L\|s_k\|^2} + \frac{\|s_k\|\gamma}{L\|s_k\|^2} + \frac{\gamma^2}{L\|s_k\|^2} \right] \|s_k\| \Rightarrow \|d_{k+1}\| \le (\gamma + \frac{\gamma}{L}) + \left[\frac{2\gamma^2}{L\|s_k\|} \right] \qquad (31)$$

Suppose that there exist constants η_1 , η_2 and η_3 such that $\eta_1 = (\gamma + \frac{\gamma}{L})$ and $\eta_2 = \frac{2\gamma^2}{L\|s_k\|}$ then (31) can be written as: $\|d_{k+1}\| \le \eta_1 + \eta_2 = \eta_3$:

ISSN: 1813 – 1662 (Print)
E-ISSN: 2415 – 1726 (On Line)

$$\therefore \frac{1}{\|d_{k+1}\|} \geq \frac{1}{\eta_3} \Rightarrow\Rightarrow \frac{1}{\|d_{k+1}\|^2} \geq \frac{1}{\eta_3^2} \Rightarrow \therefore \sum_k \frac{1}{\|d_{k+1}\|^2} \geq \sum_k \frac{1}{\eta^2_{_3 3}} = \infty$$

This is a contradiction lemma(1)

$$\therefore \sum_k \frac{1}{\|d_{k+1}\|^2} = \infty \Rightarrow \underset{k\to\infty}{Lim} \|g_k\| = 0 \blacksquare$$

4. The experiments Results:

In this part of the search , we will study the results of numerical experimental to show the activity of the proposed in three problems the Monk1-problem, Monk2-problem and continuous function approximation problem. A comparison was made between the performance of the proposed algorithm (**GDY**) with offline versions of the conjugate gradient method and each algorithm Fletcher_ Reeves update (FRCG) which is known as (traincgf) and Polack_ Ribiere method (PRCG) update which is denoted by (traincgp) (see appendix of MATLAB) as well as Dia-Yuan whose program code was written for not being available in the toolboxes of MATLAB.

The simulation was performed using MATIAB (2009a). The same random weights for each algorithm were used during the comparison process and generated them using the Nguyen-Widrow method [12]. It is worth noting that the default values of the parameters and the explanatory tools for the algorithms mentioned above were used unless otherwise stated. For all the three test questions mentioned above we was written the summarize of the performance to each algorithm for 100 simulations as table that reached a solution within a predetermined limit of time. The parameters used in all tables are: timemean the average of total time; epochmean average of total numbers of epochs, perfmean average of total perfom, gradient, the mean value of the gradient, and Succ is denoted to the succeeded simulations out of (100) marks within error function evaluations limit. All methods have been implemented with the Wolfe line search conditions (5) and (6).

Finally, It is worth mentioning that it has been considered that the algorithm has failed to train the artificial neural network if it does not reach the solution within the predetermined number of parameters and thus its epoch are not enter the statistical analysis of the algorithm.

4.1 Monk's Problems

Monk's problems are one of the first problems used to compare the performance of different learning algorithms of the neural network. This data consists of three binary classification problems based on the artificial robot domain, which are described by six different binary features. Symbolic rules There are three problems in Monk. The domains for all Monk problems are the same (432 patterns). The domain of each problem has been separated into a test and train set[12]:

4.1.1 MONK-1contains 124 examples chosen it randomly from the data group used it to train the network and the remaining data(308 examples) were applied to test the network (table(1)) and the figure (1) illustrates the behavior of these algorithms. We note from the table(1) that the proposed method approaches the behavior of the PRCG while there is a significant difference between it and the FRCG.

4.1.2 MONK-2 includes 169 patterns selected at randomly from the data group to train so the residual 263 patterns used to test the network (table(2)) while the figure (2) shows the path of convergence of these algorithms to the desired error value which is (1*e-10).

4.2 Continuous Function Approximation:

The approximation function of the continuous trigonometric function which can be written as:

$f(w) = \cos(3w) * \sin(w)$ is considered as one of the known issues for training neural networks.

The artificial neural network architecture used to solve this problem consists of a single neuron in both the input and output layer, as well as a hidden layer, which is the middle of the input and output layer consist of fifteen neurons equipped with 30 weights and 16 biases. The network is learned who to approximate the function $f(w), w \in [-\pi, \pi]$ and stopped when sum of squares of the errors come to be fewer than 0.001 (the goal). The hidden neurons of network used the logistic function as activation function and the linear activations function in the output layer. The convergence of these algorithms from the target is shown in Figure (3) and the comparative results are shown in table (3) Which shows us a close convergence in the behavior of the functions PRCG and the new algorithm DGY.

Results of Simulations for the MONK-1 Problem table(1)

Algorithms	**Timemean**	**Epochmean**	**Perfmean**	**gradient**	**Succ**
PRCG	2.3044	62.8	0.19451	0.11419	%100
FRCG	3.2604	123.56	0.19706	0.2452	%100
DYCG	2.3482	66.48	0.19533	0.16857	%100
GDY	2.3646	63.04	0.19362	0.11804	%100

Results of Simulations for the MONK-2 Problem table(2)

Algorithms	**Timemean**	**Epochmean**	**Perfmean**	**gradient**	**Succ**
PRCG	4.5054	104.33	0.18814	0.17839	%98
FRCG	6.913	253.8	0.19255	0.35646	%100
DYCG	6.8302	218.4	0.19409	0.18874	%100
GDY	4.4032	107.31	0.18747	0.17689	%98

Results of Simulations for the Continuous Fun. Approx.Problem table(3)

Algorithms	timemean	Epochmean	Perfmean	Gradient	Succ
PRCG	3.5358	119.24	0.0019505	0.020778	%100
FRCG	5.4828	216.84	0.0019717	0.038426	%100
DYCG	3.5594	113	0.0019549	0.02394	%100
GDY	3.6736	118.72	0.0019486	0.020128	%100

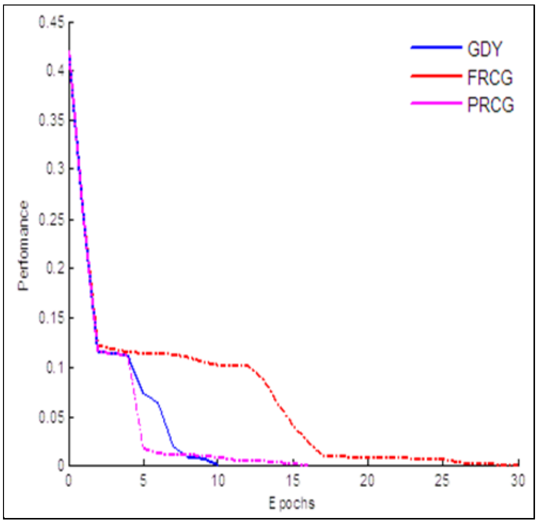

Figure (1): Monk1 Performance Comparison for Training a FFNN Using GDY, FRCG and PRCG

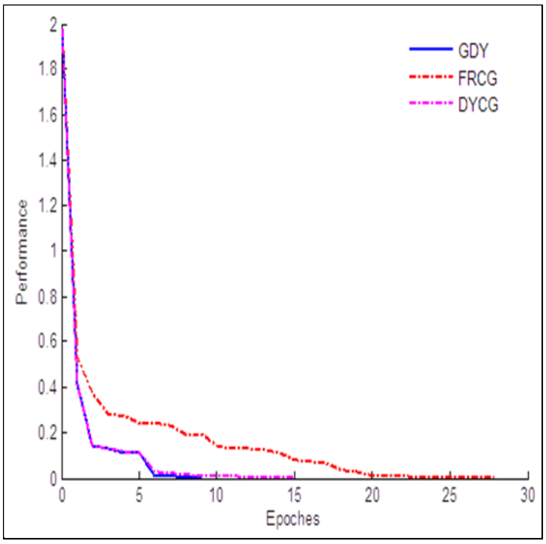

Figure (2): Monk2 Performance Comparison for Training a FFNN Using GDY, FRCG and DYCG

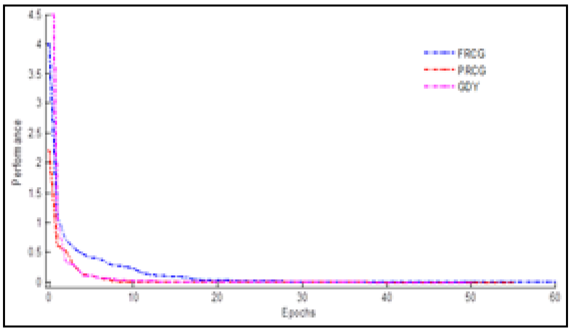

Figure (3): Function App. Performance Comparison for Training a FFNN Using NewCG, FRCG and DYCG

Conclusion

In this search, we proposed a new conjugate gradient method can be used it to train feed _forward neural networks with multilayer, the derivation of the value of β_k is based on Peary conjugacy condition. The proposed method holds descent and globally convergence under the standard Wolfe conditions. Practically, the numerical experiments showed that the convergent behavior of the proposed algorithm GDY with the Polack_Ribiere method and the large variation between it and the functions FRCG and DYCG in the other hand.

References

[1] Bishop, C. and Bishop, C. M. (1995). Neural networks for pattern recognition. Oxford university press.

[2] Rumelhart, D. E.; Hinton, G. E., and Williams, R. J. (1985). Learning internal representations by error propagation (No. ICS-8506). California Univ San Diego La Jolla Inst for Cognitive Science.

[3] Hertz, J.; Krogh, A., and Palmer, R. G. (1991). *Introduction to the theory of neural computation*. Addison-Wesley/Addison Wesley Longman.

[4] Nawi, N. M., Ransing, M. R., & Ransing, R. S. (2006). An Improved Learning Algorithm based on the Conjugate Gradient Method for Back Propagation Neural Networks. *Proc of World Academy of Science, Eng, and Technology, 14.*

[5] Beigi, H. S. and Li, C. J. (1993). Learning algorithms for neural networks based on quasi-Newton methods with self-scaling. *Journal of dynamic systems, measurement, and control*, **115(1)**: 38-43.

[6] Fletcher, R., and Reeves, C. M. (1964). Function minimization by conjugate gradients. *The computer journal*, **7(2)**: 149-154.

[7] Poliak, B.T. (1969) .The Conjugate Gradient Method in Extreme Problems, *URSS Comp. Math. Math. phys.*, **9(4)**: 94-112.

[8] Dai, Y. H., & Yuan, Y. (1999). A nonlinear conjugate gradient method with a strong global

convergence property. *SIAM Journal on optimization*, **10 (1)**: 177-182.

[9] Hestenes, M. R., and Stiefel, E. (1952). Methods of conjugate gradients for solving linear systems. *Journal of Research of the National Bureau of Standards*, **49(6)**: 409-436.

[10] Perry, A. (1978). A modified conjugate gradient algorithm, *Operation Research*, **26(6)**: 1073-1078.

[11] Dai, Y. H. and Liao, L. Z. (2001). New conjugacy conditions and related nonlinear conjugate gradient methods. *Applied Mathematics and Optimization*, ***43*(1)**: 87-101.

[12] Nguyen, D., and Widrow, B. (1990, June). Improving the learning speed of 2-layer neural networks by choosing initial values of the adaptive weights. In *Neural Networks, 1990., 1990 IJCNN International Joint Conference on* (pp. 21-26). IEEE.

تعميم خوارزمية التدرج المترافق Dai-Yuan لتدريب الشبكات العصبية متعددة الطبقات ذات التغذية الامامية

هند حسام الدين محمد

قسم الرياضيات ، كلية علوم الحاسوب والرياضيات ، جامعة الموصل ، الموصل ، العراق

الملخص

في هذا البحث نقدم نوع مختلف من خوارزميات التدرج المترافق اعتمادا على حالة شرط الترافق لـ Peary. خوارزمية التدرج المترافق الجديدة (GDY) استخدمت لتدريب الشبكات العصبية ذات التغذية الامامية وتم اثبات خاصية الانحدار والتقارب للخوارزمية المقترحة ومن ثم اختبار سلوك هذه الخوارزمية في تدريب الشبكات العصبية الاصطناعية ومقارنتها مع خوارزميات معروفة في هذا المجال من خلال نوعين من المسائل.