Convergence Analysis of the Quickprop Method

M.N. Vrahatis
University of Patras,
Department of Mathematics,
U.P. Artificial Intelligence
Research Center–UPAIRC,
GR-26500 Patras, Greece.

G.D. Magoulas
University of Athens,
Department of Informatics,
U.P. Artificial Intelligence
Research Center–UPAIRC,
GR-15771 Athens, Greece.

V.P. Plagianakos
University of Patras,
Department of Mathematics,
U.P. Artificial Intelligence
Research Center–UPAIRC,
GR-26500 Patras, Greece.

Abstract

A mathematical framework for the convergence analysis of the well known Quickprop method is described. The convergence of this method is analyzed. Furthermore, we present modifications of the algorithm that exhibit improved convergence speed and stability and at the same time, alleviate the use of heuristic learning parameters. Simulations are conducted to compare and evaluate the performance of a proposed modified Quickprop algorithms with various popular training algorithms. The results of the experiments indicate that the increased convergence rates, achieved by the proposed algorithm, affect by no means its generalization capability and stability.

Introduction

The Quickprop (Qprop) method, introduced by Fahlman in 1988 [6] is a very popular batch training algorithm for Feedforward Neural Networks (FNNs). It is well known that far from the neighborhood of a minimizer the morphology of the error surface, in certain cases, causes the Qprop algorithm to create inappropriate learning rates and the algorithm exhibits stability problems. Application-dependent heuristic learning parameters are used to alleviate this problem [6].

In this paper we analyze the Qprop method as a multivariable generalization of the secant method for nonlinear equations applied to the gradient of the batch error function. Furthermore, we present a modification of this algorithm that exhibit improved convergence speed and stability and at the same time, alleviate the use of heuristic learning parameters. Also, we prove a theorem for the convergence of the modified scheme.

Secant methods

Let $F = (f_1, f_2, \ldots, f_n) : \mathcal{D} \subset \mathbb{R}^n \to \mathbb{R}^n$ be a Fréchet-differentiable function and $x^* \in \mathcal{D}$ a solution of the nonlinear system of equations:

$$F(x) = \Theta^n \equiv (0, 0, \ldots, 0). \tag{1}$$

The best known method for approximating x^* numerically is Newton's method. Given an initial guess x^0, this method computes a sequence of points $\{x^k\}_{k=0}^{\infty}$, obtained by solving the following Newton's equation:

$$F'(x^k)\left(x^{k+1} - x^k\right) = -F(x^k). \tag{2}$$

If x^0 is sufficiently close to x^*, F is continuously differentiable in a neighborhood of x^* and the Jacobian $F'(x^*)$ is nonsingular, the iterates $\{x^k\}$ of Newton's method converge quadratically to x^*. Furthermore, under the same assumptions any sequence $\{y^k\}$ which converges to x^* superlinearly is closely related to Newton's method by the fact that the relative difference between $y^{k+1} - y^k$ and the Newton correction $-F'(y^k)^{-1}F(y^k)$ will tend to zero [3].

The quadratic convergence of Newton's method is attractive. However, the method depends on a good initial approximation and it requires in general $n^2 + n$ function evaluations per iteration besides the solution of an $n \times n$ linear system.

Quasi-Newton methods were developed to save computational effort of individual iterations while maintaining some convergence properties of Newton's method. They maintain approximations of x^* and the Jacobian at the solution $F'(x^*)$ as the iteration progresses. If x^k

0-7803-5529-6/99/$10.00

and B_k are the current approximate solution and Jacobian, then after the computation of x^{k+1}, B_k is *updated* to form B_{k+1}. The construction of B_{k+1} determines the quasi-Newton method. Given an initial guess x^0, this method computes a sequence of points $\{x^k\}_{k=0}^{\infty}$, obtained by solving the following *quasi–Newton* or *secant* equation [4]:

$$B_{k+1}\left(x^{k+1}-x^k\right)=F(x^{k+1})-F(x^k). \qquad (3)$$

The advantages of quasi-Newton methods is that they require only n function evaluations for each iteration. Hence, if a good preconditioner (initial approximation to $F'(x^*)$) can be found, these methods have an advantage in terms of function evaluation cost over Newton's method. In most quasi-Newton methods derivatives are not computed at every iteration and it is not necessary to solve completely a linear system like (2). On the other hand, the local rate of convergence turns to be superlinear instead of quadratic for most of these methods.

The most used approximation to the Jacobian proposed by Broyden [2] and his method is locally superlinear convergent, and hence is a very powerful alternative to Newton's method. Broyden's algorithm for solving (1) has the following general form (cf. [4]). Given an initial guess x^0 and a matrix B_0, this method computes a sequence of steps s_k obtained as follows:

for $k=0,1,\ldots$ until convergence **do**:

Solve $B_k s_k=-F(x^k)$ for s_k,
Set $x^{k+1}=x^k+s_k$,
Set $z^k=F(x^{k+1})-F(x^k)$,
Set $B_{k+1}=B_k+\dfrac{\left(z^k-B_k s_k\right)s_k^{\top}}{s_k^{\top}s_k}$.

Broyden's method is very popular in practice, for two main reasons. First, it generally requires fewer function evaluations than a finite difference Newton's method. Second, it can be implemented in ways that require only $O(n^2)$ arithmetic operations per iteration [5, pp.27–29].

The Quickprop method

In this section, we show that the Qprop method belongs to the family of secant methods. Of course Qprop is related to the minimization of the error function. It is well known that in minimization problems all the local minima w^* of a continuously differentiable error function E satisfy the necessary conditions:

$$\nabla E(w^*)=\Theta^n, \qquad (4)$$

where ∇E denotes the gradient of E. Eq. (4) represents a set of n nonlinear equations which must be solved to obtain w^*. Therefore, one approach to the minimization of E is to seek the solutions of the set of Eq. (4) by including a provision to ensure that the solution found does indeed correspond to a local minimizer. This is equivalent to solving the following system of equations:

$$\begin{aligned}\partial_1 E(w_1,w_2,\ldots,w_n)&=0,\\ \partial_2 E(w_1,w_2,\ldots,w_n)&=0,\\ &\vdots\\ \partial_n E(w_1,w_2,\ldots,w_n)&=0,\end{aligned} \qquad (5)$$

where $\partial_i E$ denotes the ith coordinate of ∇E.

The classical iterative scheme of the Qprop method for the ith weight is given by:

$$w_i^{k+1}=w_i^k-\left\{\frac{\partial_i E(w^k)-\partial_i E(w^{k-1})}{w_i^k-w_i^{k-1}}\right\}^{-1}\partial_i E(w^k).$$

Using matrix formulation the above scheme can by written as:

$$w^{k+1}=w^k-B_k^{-1}\nabla E(w^k),$$

where the matrix B_k is the diagonal matrix with elements $[b_{ii}^k]$, $i=1,2,\ldots,n$ given by:

$$b_{ii}^k=\frac{\partial_i E(w^k)-\partial_i E(w^{k-1})}{w_i^k-w_i^{k-1}}.$$

It is obvious that the matrix B_k satisfies the following secant equation:

$$B_k\left(w^k-w^{k-1}\right)=\nabla E(w^k)-\nabla E(w^{k-1}), \qquad (6)$$

and thus the Qprop method belongs to the class of quasi–Newton methods.

Using the above framework a straightforward modification of the Quickprop method is the following:

$$w_i^{k+1}=w_i^k-\eta_i\left\{\frac{\partial_i E(w^k)-\partial_i E(w^{k-1})}{w_i^k-w_i^{k-1}}\right\}^{-1}\partial_i E(w^k),$$

where η_i are arbitrary nonzero real numbers. This is so because the new $\eta_i b_{ii}^k$ satisfy the corresponding secant equation.

Based on the above analysis it is obvious that the Qprop as well as the above modification follows the convergence properties of the secant methods [4, 14, 16].

In general, the matrix B_k may contain non positive entries. This fact results in a non positive definite matrix, which in practice means that the method may take uphill or zero steps in the corresponding directions. To alleviate this problem in [6] a heuristic parameter, called "the maximum growth factor" has been introduced.

A modified algorithm

To avoid tuning the problem dependent heuristics of the Qprop method and to guarantee the desirable property of positive definiteness of B_k we propose the following modification:

$$w_i^{k+1} = w_i^k - \eta \left\{ \frac{|\partial_i E(w^k) - \partial_i E(w^{k-1})|}{|w_i^k - w_i^{k-1}|} \right\}^{-1} \partial_i E(w^k),$$

where the coefficient η can be properly tuned. To this end the following conditions due to Wolfe are used:

$$E(w^k + \eta^k \varphi^k) - E(w^k) \leq \sigma_1 \eta^k \langle \nabla E(w^k), \varphi^k \rangle, \quad (7)$$

$$\langle \nabla E(w^k + \eta^k \varphi^k), \varphi^k \rangle \geq \sigma_2 \langle \nabla E(w^k), \varphi^k \rangle, \quad (8)$$

where $0 < \sigma_1 < \sigma_2 < 1$. The first inequality ensures that the function is reduced sufficiently, and the second prevents the steps from being too small. It can be shown that if φ^k is a descent direction, if E is continuously differentiable and if E is bounded below along the ray $\{w^k + \eta\varphi^k \mid \lambda > 0\}$, then there always exist stepsize satisfying (7)–(8) [13, 20, 21].

The following theorem, due to Wolfe [4, 13, 20, 21], states that if E is bounded below, then the sequence $\{w^k\}_{k=0}^{\infty}$ generated by any algorithm that follows a descent direction φ^k whose angle θ_k with $-\nabla E(w^k)$ is such that:

$$\cos\theta_k = \frac{\langle -\nabla E(w^k), \varphi^k \rangle}{\|\nabla E(w^k)\| \, \|\varphi^k\|} \geq \delta > 0, \quad (9)$$

and satisfy the Wolfe's conditions, then it holds that $\lim_{k\to\infty} \nabla E(w^k) = 0$.

Theorem 1 [4, 13, 20, 21]. *Suppose that the error function $E : \mathbb{R}^n \to \mathbb{R}$ is continuously differentiable on $\mathbb{R}^n$ and assume that ∇E is Lipschitz continuous on $\mathbb{R}^n$. Then, given any $w^0 \in \mathbb{R}^n$, either E is unbounded below, or there exists a sequence $\{w^k\}_{k=0}^{\infty}$ obeying the Wolfe's conditions (7)–(8) and either:*

(i) $\langle \nabla E(w^k), (w^{k+1} - w^k) \rangle < 0$, *or*

(ii) $\nabla E(w^k) = 0$, and $w^{k+1} - w^k = 0$,

for each $k > 0$. Furthermore, for any such sequence, either:

(a) $\nabla E(w) \neq 0$ *for some* $k \geq 0$, *or*

(b) $\lim_{k\to\infty} E(w^k) = -\infty$, *or*

(c) $\lim_{k\to\infty} \langle \nabla E(w^k), (w^{k+1} - w^k) \rangle / \|w^{k+1} - w^k\| = 0$.

For a relative convergence result where the sequence $\{w^k\}_{k=0}^{\infty}$ converges q-superlinearly to a minimizer w^* see [4, p.123].

In practice, the condition (8) generally is not needed because the use of a backtracking strategy avoids very small steps. Also it can be proved (see [4]) that if (8) is replaced by:

$$E(w^k + \eta^k \varphi^k) - E(w^k) \geq \sigma_2 \eta^k \langle \nabla E(w^k), \varphi^k \rangle, \quad (10)$$

where $\sigma_2 \in (\sigma_1, 1)$, then Theorem 1 still holds.

A high level description of this modified Qprop (MQprop) algorithm is given below.

Algorithm 1 Modified Quickprop Algorithm – MQprop

1. Input $\{E; w^0; \eta^0; (\lambda_1^0, \lambda_2^0, \ldots, \lambda_n^0); MIT; \epsilon\}$.
2. Set $k = -1$.
3. If $k < MIT$, replace k by $k+1$, set $\eta = \eta^0$, and go to the next step; otherwise, go to Step 8.
4. If $k \geq 1$ and $\Lambda_i^k = |\partial_i E(w^k) - \partial_i E(w^{k-1})| / |w_i^k - w_i^{k-1}| \neq 0$, for all $i = 1, \ldots, n$, set $\lambda_i^k = 1/\Lambda_i^k$; otherwise set $\lambda_i^k = \lambda_i^0$.
5. Tune η by means of a tuning subprocedure.
6. Set $w^{k+1} = w^k - \eta \, \mathrm{diag}\{\lambda_1^k, \lambda_2^k, \ldots, \lambda_n^k\} \nabla E(w^k)$.
7. If $\|\nabla E(w^k)\| \leq \epsilon$ go to Step 8; otherwise go to Step 3.
8. Output $\{w^k; E(w^k); \nabla E(w^k)\}$.

Assume now that the tuning subprocedure of Step 5 of Algorithm 1 consists of the pair of relations (7)–(8). The following theorem states that if E is bounded below, then the sequence $\{w^k\}_{k=0}^{\infty}$ generated by Algorithm 1 converges to a point w^* for which $\nabla E(w^*) = 0$.

Theorem 2 *Suppose that the error function $E : \mathbb{R}^n \to \mathbb{R}$ is continuously differentiable and bounded below on $\mathbb{R}^n$ and assume that ∇E is Lipschitz continuous on $\mathbb{R}^n$. Then, given any point $w^0 \in \mathbb{R}^n$, for any sequence $\{w^k\}_{k=0}^{\infty}$, generated by Algorithm 1, satisfying the Wolfe's conditions (7)–(8) implies that* $\lim_{k\to\infty} \nabla E(w^k) = 0$.

Proof. The sequence $\{w^k\}_{k=0}^{\infty}$ follows the direction

$$\varphi^k(w^k) = -\mathrm{diag}\{1/\Lambda_1^k, \ldots, 1/\Lambda_n^k\} \nabla E(w^k),$$

which is a descent direction since

$$\langle \nabla E(w^k), \varphi^k(w^k) \rangle < 0.$$

Moreover, the restriction on the angle θ_k is fulfilled since, as it can be easily justified utilizing Relation (9), $\cos\theta_k > 0$. Thus, by Theorem 1 holds that $\lim_{k\to\infty} \nabla E(w^k) = 0$. Thus the theorem is proved.

Experimental study

Next, we give quantitative results on three neural network applications applying the following methods: (i) the batch Backpropagation with constant learning rate (BP) [17]; (ii) the Steepest Descent with Line Search for the learning rate proposed by Polak [16] (SDLS); (iii) the batch Backpropagation with constant stepsize and Momentum (BPM) [17]; (iv) the Adaptive Backpropagation with heuristics (ABP) [19]; (v) the Fletcher–Reeves (FR) method [7]; (vi) the Polak–Ribiere (PR) method [7]; (vii) the Polak–Ribiere (PR) method constrained by the FR method (PR-FR) [7]. In the implementation of FR, PR, and PR-FR, the Armijo line search of proposed by Polak [16] has been used. The results are exhibited in terms of the average number of iterations (μ_{IT}) required to obtain a local minimum, the average number of gradient and function evaluations (μ_{FE}) and the number of successful runs out of 1000 (Success).

The selection of initial points is very important in FNN training. Very small initial values lead to very small corrections of the variables so that practically no change takes place for some variables and more iterations are necessary to train the network [17]. In the worst case the learning stops in an undesired local minimum. On the other hand, very large initial values speed up the learning process but in many cases they can lead neurons to saturation and generate very small gradient values. In such cases, learning is considerably slow [11]. A well known initialization heuristic for FNNs is to select the points with uniform probability from an interval $(w_{\min}, w_{\max})$, where usually $w_{\min} = -w_{\max}$. A common choice is the interval $(-1, +1)$. Thus, 1000 initial starting points have been randomly selected from this interval to test the different methods.

Example 1 *The XOR problem* [10, 12, 17]. The classification of the four XOR patterns in two classes is an interesting problem because it is sensitive to initial points as well as to stepsize variations, and presents a multitude of local minima. The patterns are classified using an 2-2-1 FNN with 9 variables.

The termination condition for all algorithms tested is to find a local minimizer with batch error function value $E \leq 0.04$. The results are summarized in following table.

Table 1: *Results for the XOR problem.*

Algorithm	μ_{IT}	μ_{FE}	Success
BP	549	1098	810/1000
SDLS	64	435	810/1000
BPM	803	1606	810/1000
ABP	157	314	810/1000
FR	84	282	130/1000
PR	21	169	380/1000
PR-FR	22	171	410/1000
MQprop	52	234	810/1000

In this case the number of successful runs is related to the local minima problem. Thus FR, PR and PR-FR usually converge to an undesired local minimum, i.e. a minimizer with function value $E > 0.04$ which means that some of the patterns are not correctly classified. MQprop exhibits better performance than FR, PR and PR-FR as regards the number of successful runs. Concerning training speed, measured by the mean number of function and gradient evaluations needed to successfully classify the patterns MQprop outperforms BP, SDLS, BPM and FR. Note that PR and PR-FR require less function evaluations than MQprop but they reveal a smaller number of successful runs.

Example 2 *Texture classification problem* [12]. A total of 12 Brodatz texture images [1]: 3, 5, 9, 12, 15, 20, 51, 68, 77, 78, 79, 93 (see Figure 1 in [12]) of size 512×512 is acquired by a scanner at 150dpi. From each texture image 10 subimages of size 128×128 are randomly selected, and the co-occurrence method, introduced by Haralick [8] is applied. In the co-occurrence method, the relative frequencies of gray–level pairs of pixels at certain relative displacements are computed and stored in a matrix. The combination of the nearest neighbor pairs at orientations 0°, 45°, 90° and 135° are used in the experiment. A set of 10 sixteenth-dimensional training patterns are created from each image. An 16-8-12 FNN with 244 variables is trained to classify the patterns to the 12 texture types.

Detailed results regarding the training performance of the algorithms are presented in Table 2. The termination condition is a classification error $CE < 3\%$ [12]; that is the network classifies correctly 117 out of the 120 patterns.

The successfully trained FNNs are tested for their generalization capability, [9, 17], using test patterns from 20 subimages of the same size randomly selected from each image. To evaluate the generalization performance of the FNN the *max* rule is used, i.e. a test pattern is considered to be correctly classified if the corresponding output neuron has the greatest value among the output

neurons. The average percentage of success in classification for each algorithm is: BP=90.0%; SDLS=90.0%; BPM=90.0%; ABP=93.5%; FR=92.0%; PR=92.6%; PR-FR=93.5%; MQprop=94.0%.

Table 2: *Results for the texture classification problem.*

Algorithm	μ_{IT}	μ_{FE}	Success
BP	15839	31678	960/1000
SDLS	13256	26517	965/1000
BPM	12422	24844	940/1000
ABP	560	1120	1000/1000
FR	1624	12674	250/1000
PR	140	810	990/1000
PR-FR	145	1005	996/1000
MQprop	406	1228	1000/1000

Regarding the training phase, PR exhibits the best performance in terms of the average number of gradient and error function evaluations required. On the other hand ABP, and MQprop are more robust in the sense that they exhibit larger number of successes providing also good generalization capability.

Example 3 *The numeric font learning problem* [12, 18]. This experiment refers to the training of a multilayer FNN with 460 variables for recognizing 8×8 pixel machine printed numerals from 0 to 9. The network has 64 input neurons and 10 output neurons representing 0 through 9. Numerals are given in a finite sequence $C = (c_1, c_2, \ldots, c_p)$ of input–output pairs $c_p = (u_p, t_p)$ where u_p are the binary input vectors in $\mathbb{R}^{64}$ determining the 8×8 binary pixel and t_p are binary output vectors in $\mathbb{R}^{10}$, for $p = 1, \ldots, 10$, determining the corresponding numerals.

Table 3: *Results for the numeric font problem.*

Algorithm	μ_{IT}	μ_{FE}	Success
BP	14489	28978	660/1000
SDLS	12225	24454	990/1000
BPM	10142	20284	540/1000
ABP	1975	3950	910/1000
FR	620	3121	420/1000
PR	649	2124	960/1000
PR-FR	750	3473	1000/1000
MQprop	159	739	1000/1000

The termination condition is to locate a minimizer with function value less than or equal to 0.001. The results are summarized in Table 3. It is clear that MQprop achieves faster training than all other methods.

Conclusions

In this contribution the convergence of the Qprop method has been considered. Some modifications of the classical Qprop algorithm have been presented and a strategy for alleviating the use of highly problem-dependent heuristic learning parameters that are necessary in order to secure the stability of this algorithm have been proposed. A new theorem that guarantees the convergence of the proposed modified Qprop has been proved. This modified Qprop scheme exhibits rapid convergence and provides stable learning and therefore, a greater possibility of good performance.

References

[1] P. Brodatz, Textures - a photographic album for artists and designers, Dover, New York, (1966).

[2] C.G. Broyden, A class of methods for solving nonlinear simultaneous equations, Math. Comp., 19, 577–593, (1965).

[3] J.E. Dennis Jr and J. Moré, A characterization of superlinear convergence and its applications to quasi Newton methods, Math. Comp., 28, 548–560, (1974).

[4] J.E. Dennis Jr and R.B. Schnabel, Numerical methods for unconstrained optimization and nonlinear equations, Prentice Hall, Inc., Englewood Cliffs, New Jersey, (1983).

[5] J.E. Dennis Jr and R.B. Schnabel, A view of unconstrained optimization, in: Handbooks in OR & MS, vol. 1, eds. G.L. Nemhauser et al., Elsevier Science Publishers B.V., North-Holland, (1989).

[6] S.E. Fahlman, Faster-learning variations on back-propagation: an empirical study. In Proceedings of the 1988 Connectionist Models Summer School, D.S. Touretzky, G.E. Hinton, and T.J. Sejnowski, eds., pp. 38–51, Morgan Kaufmann, San Mateo, CA, (1988).

[7] J.C. Gilbert and J. Nocedal, Global convergence properties of conjugate gradient methods for optimization, SIAM J. Optimization, 2, 21–42, (1992).

[8] R. Haralick, K. Shanmugan, and I. Dinstein, Textural features for image classification, IEEE Trans. System, Man and Cybernetics, 3, 610–621, (1973).

[9] S. Haykin, Neural Networks: A Comprehensive Foundation, Macmillan College Publishing Company, (1994).

[10] R.A. Jacobs, Increased rates of convergence through learning rate adaptation, Neural Networks, 1, 295–307, (1988).

[11] G.D. Magoulas, M.N. Vrahatis, and G.S. Androulakis, A new method in neural network supervised training with imprecision. In Proceedings of the IEEE 3rd International Conference on Electronics, Circuits and Systems, pp. 287–290, (1996).

[12] G.D. Magoulas, M.N. Vrahatis, and G.S. Androulakis, Effective back-propagation with variable stepsize. Neural Networks, 10, 69–82, (1997).

[13] J. Nocedal, Theory of algorithms for unconstrained optimization, Acta Numerica, 199–242, (1992).

[14] J.M. Ortega and W.C. Rheinboldt Iterative Solution of Nonlinear Equations in Several Variables, Academic Press, NY, (1970).

[15] V.P. Plagianakos, D.G. Sotiropoulos, and M.N. Vrahatis, Automatic adaptation of learning rate for Backpropagation Neural Networks, Recent Advances in circuits and systems, Nikos E. Mastorakis, ed., World Scientific, (1998).

[16] E. Polak, Optimization: Algorithms and Consistent Approximations, Springer-Verlag, New York, (1997).

[17] D.E. Rumelhart, G.E. Hinton, and R.J. Williams, Learning Internal Representations by Error Propagation. In D. E. Rumelhart, and J. L. McClelland, eds., Parallel Distributed Processing: Explorations in the Microstructure of Cognition, 1, pp. 318–362. MIT Press, Cambridge, Massachusetts, (1986).

[18] A. Sperduti and A. Starita, Speed up learning and network optimization with extended back-propagation, Neural Networks, 6, 365–383, (1993).

[19] T.P. Vogl, J.K. Mangis, J.K. Rigler, W.T. Zink, and D.L. Alkon, Accelerating the convergence of the backpropagation method, Biological Cybernetics, 59, 257–263, (1988).

[20] P. Wolfe, Convergence conditions for ascent methods, SIAM Review, 11, 226–235, (1969).

[21] P. Wolfe, Convergence conditions for ascent methods. II: Some corrections, SIAM Review, 13, 185–188, (1971).