Technical Report
INESC RT07/97

On-Line Step Size Adaptation

Luís B. Almeida, Thibault Langlois and José D. Amaral

INESC/IST, 9, Rua Alves Redol
1000 Lisbon, Portugal
e-mail: {lba,tl,jda}@inesc.pt

Category: **Algorithms and Architectures**
Sub-category: **online learning algorithms**

July 16, 1997

Abstract

Gradient-based methods are often used for optimization. They form the basis of several neural network training algorithms, including backpropagation. They are known to be slow, however. Several techniques exist for the acceleration of gradient-based optimization, but very few of them are applicable to stochastic (or real-time) optimization. This paper proposes a new step size adaptation technique, designed specifically for accelerating stochastic gradient optimization (and therefore also the real-time training of neural networks). The theoretical basis of the technique is discussed, and an experimental evaluation of the technique's performance is reported.

1 Introduction

Gradient descent/ascent is often used for optimization. In the neural networks area, it forms the basis of the widely used backpropagation algorithm [9]. Plain gradient based optimization is known to be slow, however. Several acceleration techniques have appeared over the years: momentum [9], adaptive step sizes [10], second order methods [3][6][2], conjugate gradients [8] etc. Most of these techniques (with the notable exception of momentum) were developed only for deterministic optimization (which corresponds to batch-mode training in neural networks). However, both in the training of neural networks and in other optimization problems it is often convenient to use stochastic (on-line) optimization. In the training of neural networks, this usually happens when the training set is very large. In other applications of optimization, one sometimes has access only to a noisy estimate of the gradient, making the use of deterministic gradient impossible.

Some proposals of stochastic step size adaptation procedures have appeared in the literature [4][5], [7]. However, none of them seems to be simple and general enough for widespread use. In this paper we propose a new, simple step size adaptation technique for stochastic gradient optimization, similar in spirit to the deterministic adaptive step sizes technique of [10] (see also [11]).

This paper is organized as follows. Section 2 briefly reviews the deterministic adaptation technique. Section 3 presents the new stochastic technique. Section 4 presents experimental results, and Section 5 concludes.

In this paper we will use the neural networks nomenclature. The parameters to be optimized are called weights, each on-line optimization step is said to correspond to the presentation of a pattern and each deterministic optimization step in referred to as an epoch.

2 Deterministic adaptive step sizes

The deterministic adaptive step sizes technique uses one step size parameter per weight. Each step size parameter is increased if the sign of the corresponding gradient component stays the same for two consecutive epochs, and is decreased otherwise:

$$\eta_{ij}^{(n)} = \begin{cases} \eta_{ij}^{(n-1)} u & \text{if } \frac{\partial E^{(n)}}{\partial w_{ij}} \frac{\partial E^{(n-1)}}{\partial w_{ij}} > 0 \\ \eta_{ij}^{(n-1)} d & \text{if } \frac{\partial E^{(n)}}{\partial w_{ij}} \frac{\partial E^{(n-1)}}{\partial w_{ij}} < 0 \end{cases} \tag{1}$$

where u is slightly above 1 and d is slightly below 1, $\frac{\partial E^{(n)}}{\partial w_{ij}}$ is a shorthand notation for

$$\left. \frac{\partial E}{\partial w_{ij}} \right|_{w_{ij}=w_{ij}^{(n)}}$$

and the superscript (n) denotes the n-th epoch. The weights are updated according to

$$w_{ij}^{(n+1)} = w_{ij}^{(n)} - \eta_{ij}^{(n)} \frac{\partial E^{(n)}}{\partial w_{ij}} \tag{2}$$

Momentum and error control can be combined with this technique, yielding a very efficient and robust batch-mode training method [1]. This technique is not readily adaptable to stochastic-mode training, because of the need to know the exact value of the gradient. Use of the gradient estimates available in stochastic mode (which we shall designate as *sign-based stochastic adaptation*) can easily lead to erroneous behavior. For example, if the distribution of the noise present in the estimates of the partial derivatives of the objective function becomes skewed, the probability of making step size increases will become higher, because the probability of having two successive errors with the same sign will increase. This may lead to the use of step sizes that are larger than desirable for the situation at hand, and even to divergence.

3 Stochastic adaptive step sizes

Like many other fast optimization algorithms, the stochastic step size adaptation algorithm that we propose is based on the assumption that the function to be minimized can be approximated by a quadratic function. We wish to keep the weight update equation (2) but with a new update rule for the η_{ij}. Since we want to perform step size adaptation for each weight independently of the other weights, we will further assume that the Hessian of the quadratic is diagonal. This diagonality assumption may seem rather strong, but is frequently made with good results. It is, for example, the basis of the second order method of [3][6] and is also implicit in the deterministic adaptation procedure described in Section 2. Given this assumption we can then treat the weights independently of one another, and therefore we only need to examine the case of a function of a single variable. Assume we wish to minimize the quadratic function

$$E(w) = \frac{a}{2}(w - m)^2 + b$$

where a, b and m are unknown parameters. The value of m, the location of the minimum, is what we wish to find. We will assume that, at the n-th iteration, we are at the known (i.e. non-random) point $w^{(n)}$, and that we have access to a noisy estimate of the derivative of E at that point,

$$d_n = E'\left[w^{(n)}\right] + \epsilon_n$$

where ϵ_n is a random variable with zero mean and with variance σ_n^2. We will obtain the next value of w according to

$$w^{(n+1)} = w^{(n)} - \eta^{(n)} d_n$$

and we would like to use the "best possible" value for the step size parameter $\eta^{(n)}$. One possible definition for this "best possible" value would be the one that minimizes

$$\langle [w^{(n+1)} - m]^2 \rangle$$

where $\langle . \rangle$ denotes expectation. It is therefore natural to try to set the derivative of this quantity relative to $\eta^{(n)}$ equal to zero. This derivative is

$$\frac{\partial \langle [w^{(n+1)} - m]^2 \rangle}{\partial \eta^{(n)}} = -2 \langle [w^{(n+1)} - m] d_n \rangle$$

Consider, however, that

$$\begin{aligned} \langle d_n d_{n+1} \rangle &= \langle d_n \{ a[w^{(n+1)} - m] + \epsilon_{n+1} \} \rangle \\ &= a \langle d_n [w^{(n+1)} - m] \rangle \end{aligned}$$

where, in the last equation, we have assumed that ϵ_n and ϵ_{n+1} are uncorrelated. Therefore,

$$\frac{\partial \langle [w^{(n+1)} - m]^2 \rangle}{\partial \eta^{(n)}} = -\frac{2}{a} \langle d_n d_{n+1} \rangle \tag{3}$$

The uncorrelatedness that we have mentioned above is a reasonable assumption in many situations. Specifically in the on-line training of neural networks, we assume that successive input patterns are selected at random, independently from one another. Since the pattern selected at iteration n is what determines the value of ϵ_n, the uncorrelatedness assumption seems natural in this case.

Since we cannot compute the expected value involved in the right hand side of (2) (and we also cannot obtain d_{n+1} without first choosing $\eta^{(n)}$), we cannot directly set the right hand side of (2) to zero, to obtain the best value for $\eta^{(n)}$. This is not hopeless, however. In many situations the best value for $\eta^{(n)}$ changes slowly as a function of n[1]. In such cases we can optimize $\eta^{(n)}$ by stochastic gradient descent

$$\eta^{(n+1)} = \eta^{(n)} + \beta d_n d_{n+1}$$

or by any alternative method that tends to drive $\langle d_n d_{n+1} \rangle$ to zero. Since our experience with the batch-mode adaptive step sizes technique shows that there are advantages in a geometric adaptation of the step size parameters, we will use

$$\eta^{(n+1)} = \eta^{(n)} e^{\beta d_n d_{n+1}} \tag{4}$$

which we shall call the *unnormalized* step sizes adaptation procedure.

Although this procedure does work, it is clear that the values of d_n and d_{n+1} (and therefore also the values of the exponential) will vary widely from one problem to another, or even within the same problem as we progress in the optimization. This can be compensated by adjusting the value of β, but

[1]We won't discuss in detail, here, which are these situations. We will mention, however, that the optimization of a quadratic function, like the one we are considering here, is one such situation if σ_m^2 changes slowly with n (or is constant), and n is sufficiently large.

we would like to obtain a procedure in which parameter tuning is reduced to a minimum. We therefore propose the *normalized* adaptation procedure

$$\eta_{n+1} = \eta^{(n)} e^{\beta \frac{d_n d_{n+1}}{v_n}} \tag{5}$$

where the normalization divisor v_n is an exponential average of the past squared derivative estimates, obtained through the recursion

$$v_n = \gamma v_{n-1} + (1-\gamma)(d_{n+1})^2,$$

the parameter γ controlling the effective length of the exponential average.

4 Experimental tests

Two types of experiments were performed to test the validity of the proposed stochastic step size adaptation procedure. The first consisted of finding the minimum of an analytically defined function; the second consisted of actually performing the on-line training of a neural network. For the first set of experiments we chose the well known Rosenbrock function as the function to be minimized. This is a function of two variables, defined as

$$F(x,y) = 100(y - x^2) + (x-1)^2$$

which has a narrow, gently sloping valley along the parabola $y = x^2$, with a single absolute minimum at the point (1,1). To create a stochastic gradient descent situation, i.i.d. random values were added to the partial derivatives of F before using them in the gradient descent procedure. These random values had a Laplacian distribution with a standard deviation of 1.44. Minimization was started at the point (-1,1), and was stopped when we reached a point where $F < 0.1$, a maximum of 10,000 tests were allowed.

Figure 1 shows the number of steps needed to reach the stopping criterion, as a function of the initial value of the step size parameter, for three stochastic gradient methods: (1) fixed step sizes, (2) the sign-based stochastic adaptation procedure described in Section 2, and (3) the normalized adaptation procedure given by equation (4). Due to time limitations, only one test was performed for each situation. We used $u = 1.2$ and $d = 1/u$, in the sign-based method (it is very important to have $d = 1/u$, for this method, otherwise it will diverge very often). For the new adaptation method we used $\beta = 0.01$ and $\gamma = 0.9$.

It is apparent that the new procedure shows a very low sensitivity to the initial value of the step size parameter, for a wide range of values of this parameter. Fixed step sizes show a great sensitivity to the step size parameter, as expected, and sign-based adaptation has somewhat an erratic behavior, with some situations in which it diverges. In other tests, where we used a skewed distribution for the noise added to the derivative, the number

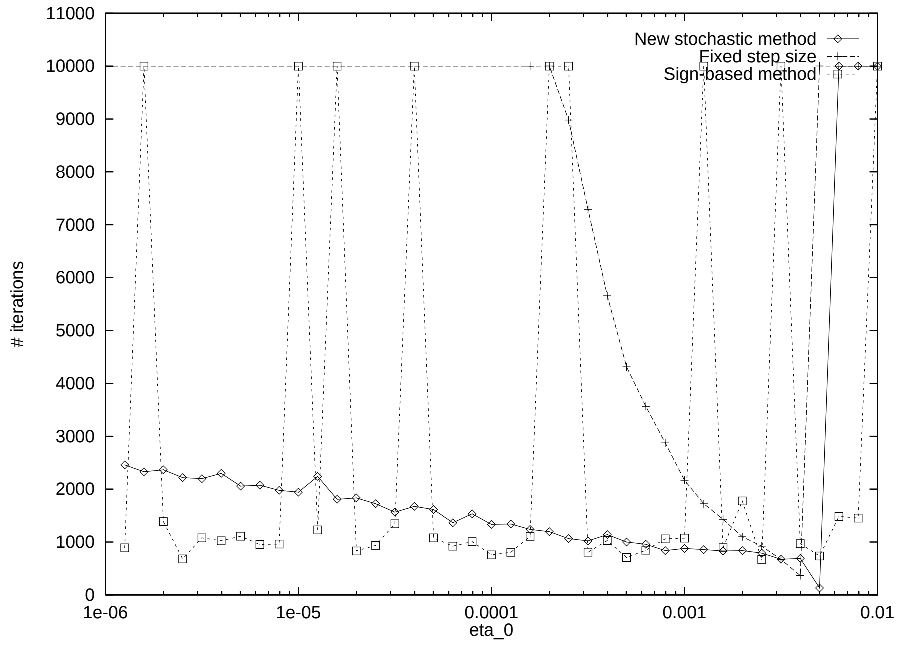

Figure 1: *Rosenbrock function: Number of training patterns necessary to reach $F < 0.1$ as a function of the initial step value.*

of cases of divergence of the sign-based procedure was higher, confirming the reasoning made at the end of Section 2.

The second experiment consisted of training a multilayer perceptron to implement the mapping

$$z = f(x,y) = \frac{\sin\left(20\sqrt{x^2+y^2}\right)}{20\sqrt{x^2+y^2}} + \frac{1}{5}\cos\left(10\sqrt{x^2+y^2}\right) + \frac{y}{2} - 0.3$$

The perceptron had a single hidden layer with 20 units, and was fully connected between successive layers, with no direct connections from input to output. The nonlinearities of all units were tanh functions. The training data were uniformly distributed in the interval [-1 1] on both axis. Training was stopped when the MSE error in the training set became less than 0.05. The training phase was stopped after 32,000 training patterns. The weights were initialized between -1 and 1 with a uniform distribution. The u and d parameters of the sign-based stochastic adaptation procedure were 1.1 and 0.909 respectively. The parameters used for the new stochastic adaptation method were: $\beta = 0.01$ and $\gamma = 0.95$.

Figure 2 shows the results of this experiment. We can see, once again, the low sensitivity of the new stochastic adaptation procedure, as well as the large sensitivity of the fixed step size one. In this figure, sign-based adaptation seems to perform quite well. However, it still had an erratic

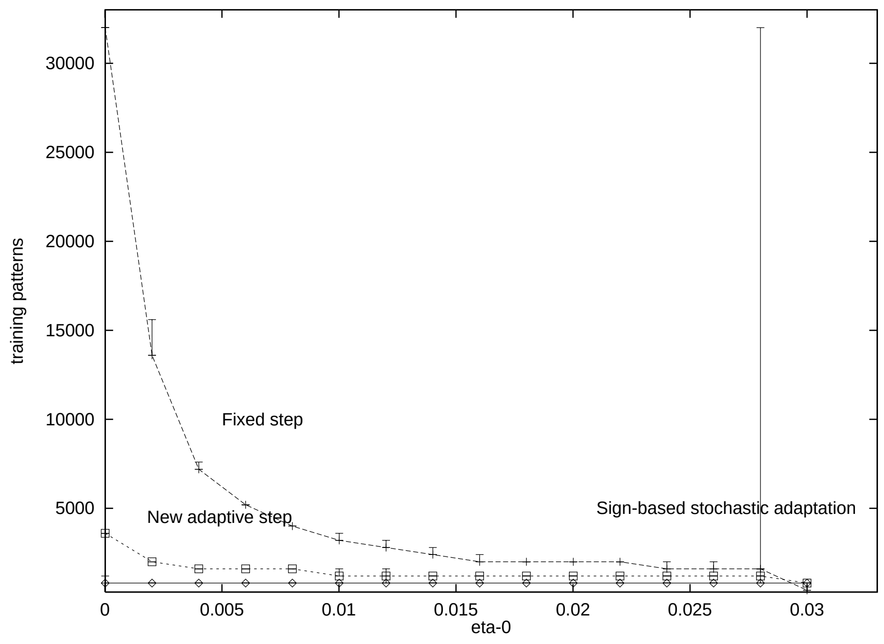

Figure 2: *MLP training: Number of training patterns necessary to obtain a MSE below 0.05 as a function of the initial step value (η_0). Three trainings were performed for each value of η_0. The curves show the medians of numbers of training patterns, and the vertical bars show the minimum and maximum for each case.*

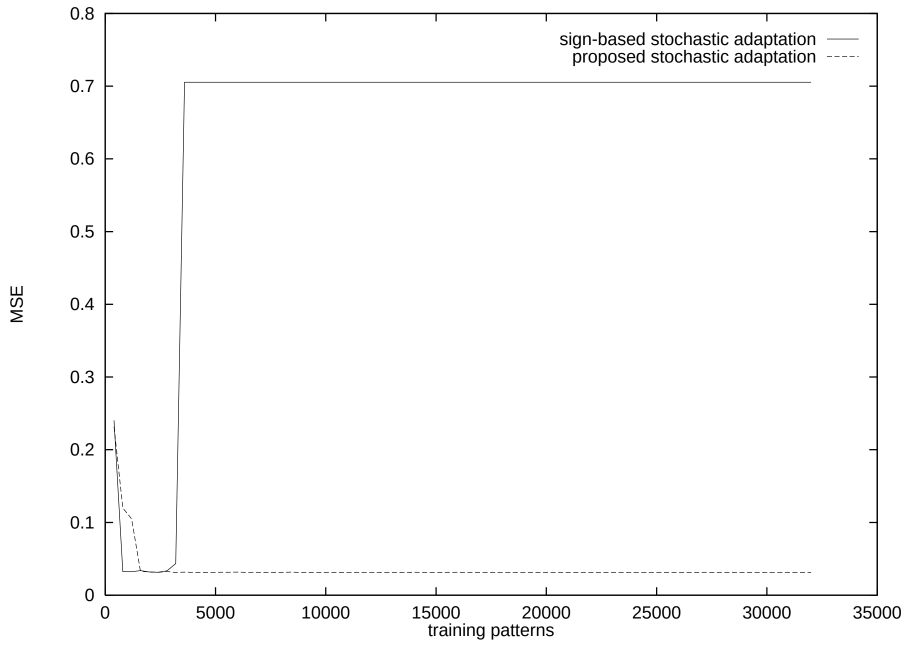

Figure 3: *Two typical learning curves obtained on the training set for sign-based adaptation and for the new stochastic adaptation procedure.*

performance that we illustrate in Figure 3. Here we show two learning curves, one for sign-based adaptation and another for the new adaptation procedure. Both curves are quite typical. The sign-based procedure diverged after an initial convergence phase, in 13 out of 15 cases. The new procedure never diverged.

5 Conclusions

We have proposed a new method for step size adaptation in stochastic gradient optimization. This method uses independent step sizes for all parameters, and adapts them using the derivative estimates available in the gradient optimization procedure. Experimental tests with the optimization of the Rosenbrock function and with the on-line training of a multilayer perceptron substantiate the effectiveness of the proposed procedure. While more extensive testing still needs to be done, the new procedure seems to be a good candidate for the acceleration of the on-line training of neural networks.

6 Acknowledgments

This work was partly funded by the Office of Naval Research under Navy Grant N00014-94-1-0858, by JNICT under contract PBIC/C/TIT/2455/95 and by the PRAXIS XXI program under contract 2/2.1/TIT/1585/95.

References

[1] Luís B. Almeida. Multilayered perceptron. In E. Fiesler and R. Beale, editors, *Handbook of Neural Computation*. Oxford University Press and IOP Publishing, 1996.

[2] R. Battiti. First- and second- order methods for learning: between steepest descent and newton's methods. *Neural Computation*, 4:141–166, 1992.

[3] S. Becker and Y. Le Cun. Improving the convergence of back-propagation learning with second order methods. Technical Report CRG-TR-88-5, University of Toronto, 1986.

[4] Christian Darken and John Moody. Learning rate schedules for faster stochastic gradient search. In *Neural Networks for Signal Processing 2 — Proceedings of the 1992 IEEE Workshop*, Piscataway, NJ, 1992. IEEE Press.

[5] Christian Darken and John Moody. Towards faster stochastic gradient search. In Moody, Hanson, and Lippmann, editors, *Advances in Neural Information Processing Systems 4*, Palo Alto, 1992. Morgan Kaufmann.

[6] S. Fahlman. Faster-learning variations on back-propagation: An empirical study. In *Connectionist Models Summer School*, Carnegie Mellon, 1988.

[7] Todd K. Leen and Genevieve B. Orr. Using curvature information for fast stochastic search. In M. I. Jordan M.C. Mozer and T. Petsche, editors, *Neural Information Processing Systems, 9*. MIT Press, 1996.

[8] M. F. Moller. A scaled conjugate gradient algorithm for fast supervised learning. *Neural Networks*, 6:525–533, 1993.

[9] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning internal representations by error propagation. In *PDP : Explorations in the Microstructures of Cognition*, volume 1, pages 318–362, NewYork:IRE, 1986.

[10] Fernando M. Silva and Luis B. Almeida. Speeding up backpropagation. In R. Eckmiller, editor, *Advanced Neural Computers*, pages 151–160. North Holland, 1990.

[11] T. Tollenaere. Supersab: Fast adaptive back propagation with good scaling properties. *Neural networks*, 3:561–573, 1990.