Pergamon

Comput. & Graphics, Vol. 18, No. 4, pp. 593–597, 1994

Printed in Great Britain
0097-8493/94 $6.00 + .00

0097-8493(94)E0053-7

Chaos and Graphics

OSCILLATING ITERATION PATHS IN NEURAL NETWORKS LEARNING

RAÜL ROJAS
Freie Universität Berlin, Institut für Informatik, Takustr. 9, Berlin 14195, Germany

Abstract—In this paper we show that finding optimal combinations of learning and momentum rate for the standard backpropagation algorithm used to train neural networks involves difficult trade-offs. Gradient descent can be accelerated with a larger step size and momentum rate, but the stability of the iteration process is affected by certain combinations of parameters. We show in which cases backpropagation produces an oscillatory behavior and how its simple feed-back nature can lead to chaotic iterations. Some graphics illustrate the kind of problems which can be found when applying backpropagation and which are often disregarded.

1. INTRODUCTION

Neural networks can be defined as directed graphs of computing elements in which the nodes of the network evaluate a certain primitive function of their input[1]. Information flows through the edges of the network, where it is multiplied by the corresponding edge's weight. Learning in neural networks consists of finding the appropriate weights so that a given function can be approximated by the network as closely as possible. Starting with random weights, an iterative process refines the network's parameters until a good enough approximation to the given function has been found. The function to be approximated is known usually only from a set of input-output examples, *i.e.*, the training set. The error function of the network measures the deviation between desired and current output for a given input from the training set. In most cases it is a continuous and differentiable function of the network's weights, and a simple strategy, like gradient descent, can lead to optimal weight combinations. The standard backpropagation algorithm is just a straightforward implementation of gradient descent on the error function. If the weight adjustments are done immediately after each pattern presentation we speak of *on-line* backpropagation. If the weight adjustments are done after all input patterns have been processed with the network and all deviations have been measured, we speak of *off-line* or *batch* backpropagation. The size of the correction step in the negative gradient direction is determined by a constant called the *learning rate*.

A popular variation of the basic learning algorithm is the introduction of a *momentum term*. The gradient of the error function is computed for each new parameter combination but instead of just following the negative gradient direction, a weighted average of the current gradient and the previous correction direction is computed at each step. This average is used as the new correction direction. Theoretically, this approach should provide the search process with some inertia and could help to avoid excessive oscillations in narrow valleys of the error function. However, there is empirical evidence that trying to adjust the learning and momentum rate to minimize the processing time is a hard computational problem. It has also been shown that backpropagation leads to chaotic behavior in the case of continuous units[2]. In this paper we show what are the trade-offs involved in choosing a specific learning and momentum rate, and that chaotic behavior can also be observed in the discrete case with the backpropagation feed-back rule and large momentum rates. We show, moreover, that such large momentum rates are necessary when the optimal size of the learning step is unknown and the form of the error function is highly degenerate.

In standard backpropagation, as explained before, some input-output patterns are fed into a network, and the error function E is determined at the output. In a network with n different weights $w_1, w_2, \ldots, w_n$ the i-th correction step for weight w_k is given by

$$\Delta w_k(i) = -\gamma \frac{\partial E}{\partial w_k} + \alpha \Delta w_k(i-1),$$

where γ and α are the learning and momentum rate, respectively. Normally we are interested in accelerating the convergence to a minimum of the error function, and this can be done by increasing the learning rate up to an optimal value. Several fast learning algorithms for neural networks work by trying to find the best value of γ, which still guarantees convergence. The introduction of the momentum rate allows the attenuation of oscillations in the iteration process.

Adjusting both learning parameters to yield the best possible convergence is normally done by trial and error, or even some kind of random search[3]. Since the optimal parameters are highly dependent on the learning task, no general strategy has been developed to deal with this problem. In this paper we show why some parameter combinations yield poor results.

2. THE LINEAR ASSOCIATOR

Let us first consider the case of a linear associator, that is, a single computing element with associated weights $w_1, w_2, \ldots, w_m$ and, which for the input $x_1, x_2, \ldots, x_m$ produces $w_1x_1 + \cdots + w_mx_m$ as output.

The input-output patterns in the training set are the p ordered pairs $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_p, y_p)$, whereby the input patterns are vectors of dimension m and the output patterns are scalars. The weights of the linear associator can be ordered in an m-dimensional column vector $\mathbf{w}$ and the learning task consists of finding that $\mathbf{w}$, which minimizes the quadratic error

$$E = \sum_{i=1}^{m} \| \mathbf{x}_i \mathbf{w} - y_i \|^2.$$

By defining a $p \times m$ matrix $\mathbf{X}$ whose rows are the vectors $\mathbf{x}_1, \ldots, \mathbf{x}_p$ and a column vector $\mathbf{y}$ whose elements are the scalars $y_1, \ldots, y_p$, the learning task reduces to the minimization of

$$\begin{aligned} E &= \|\mathbf{X}\mathbf{w} - \mathbf{y}\|^2 \\ &= (\mathbf{X}\mathbf{w} - \mathbf{y})^T(\mathbf{X}\mathbf{w} - \mathbf{y}) \\ &= \mathbf{w}^T(X^T\mathbf{X})\mathbf{w} - 2\mathbf{y}^T\mathbf{X}\mathbf{w} + \mathbf{y}^T\mathbf{y}. \end{aligned}$$

Since this is a quadratic function, the minimum is found at $\mathbf{w} = \mathbf{X}^+\mathbf{y}$, where $\mathbf{X}^+$ is the pseudoinverse of the matrix $\mathbf{X}$. A solution can also be found using backpropagation.

The quadratic function E can be thought of as a paraboloid in m-dimensional space. The lengths of its principal axes are determined by the magnitude of the eigenvalues of the covariance matrix $\mathbf{X}^T\mathbf{X}$. Gradient descent is most effective when the principal axes of the quadratic form are all of the same length. In this case, the gradient vector points directly toward the minimum of the error function. When the axes of the paraboloid are of very different size, the gradient direction can lead to oscillations in the iteration process[4].

Let us consider the simple case of the quadratic function $ax^2 + by^2$. Gradient descent yields the iteration rule

$$\Delta x(i) = -2\gamma a x + \alpha \Delta x(i-1)$$

in the x direction and

$$\Delta y(i) = -2\gamma b x + \alpha \Delta y(i-1)$$

in the y direction. An optimal parameter combination in the x direction is $\gamma = 1/2a$ and $\alpha = 0$. In the y direction the optimal combination is $\gamma = 1/2b$ and $\alpha = 0$. Since the iteration proceeds with a single γ value, we have to find a compromise between these two options. Intuitively an intermediate γ should do best, when the momentum term is zero. Figure 1 shows the number of iterations needed to find the minimum of the error function to a given precision as a function of γ, when $a = 1.5$ and $b = 1$. The optimal value for γ is the one found at the intersection of the two curves. The global optimal γ is larger than the optimal γ in the x direction and smaller than the optimal γ in the y direction. This means that there will be some oscillations in the y direction and slow convergence in the x direction, but this is the best possible compromise. It is obvious that in the m-dimensional case we could have oscillations in some of the principal directions and slow convergence in others. A simple strategy that is used by some fast learning algorithms to avoid these problems consists of using a different learning rate for each weight, that is a different γ for each direction in weight space[5].

3. MINIMIZING OSCILLATIONS

Since the lengths of the principal axes of the error function are given by the eigenvalues of the covariance matrix $\mathbf{X}^T\mathbf{X}$, and since one of these eigenvalues could be much larger than the other ones, the range of possible values for γ reduces accordingly. Nevertheless, a very small γ and the oscillations it produces can be neutralized by increasing the momentum term. A detailed discussion of the one-dimensional case provides

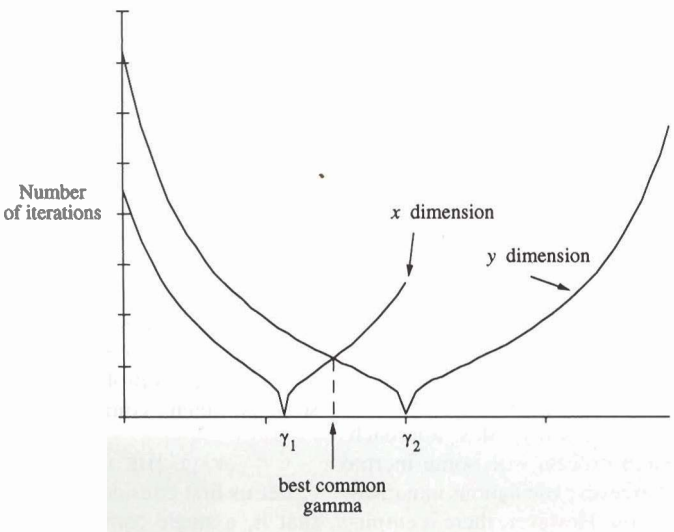

Fig. 1. Optimal γ in the two-dimensional case.

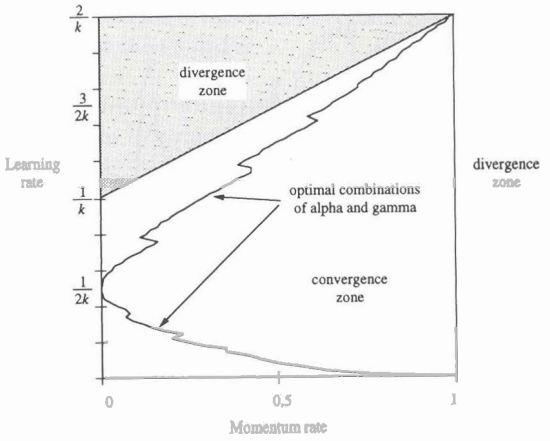

Fig. 2. Convergence and nonconvergence regions.

us with the necessary insight for the understanding of more complex cases.

In the one-dimensional case, that is when minimizing functions of type kx^2, the optimal step length is given by $\gamma = 1/2k$. A step length of $\gamma = 1/k$ produces an oscillation between the initial point x_0 and $-x_0$. Any γ greater than $2/k$ leads to an "explosion" of the iteration process. Figure 2 shows the main regions for parameter combinations of γ and α. These regions were determined by iterating in the one-dimensional case and integrating the length of the iteration path. Parameter combinations in the divergence region lead to the explosion of the iteration process. Parameter combinations in the boundary between regions lead to stable oscillations.

Figure 2 provides us with some interesting information. Any value of γ greater than four times the constant $1/2k$ cannot be balanced with any value of α. Values of α greater than 1 are prohibited since they lead to a geometric explosion of the iteration process. Any value of γ between the explosion threshold $1/k$ and $2/k$ can be made to lead to convergence by a large enough α. For any given γ between $1/k$ and $2/k$ there exist two points in which the iteration process falls in a stable oscillation, namely at the boundaries between regions. For values of γ under the optimal value

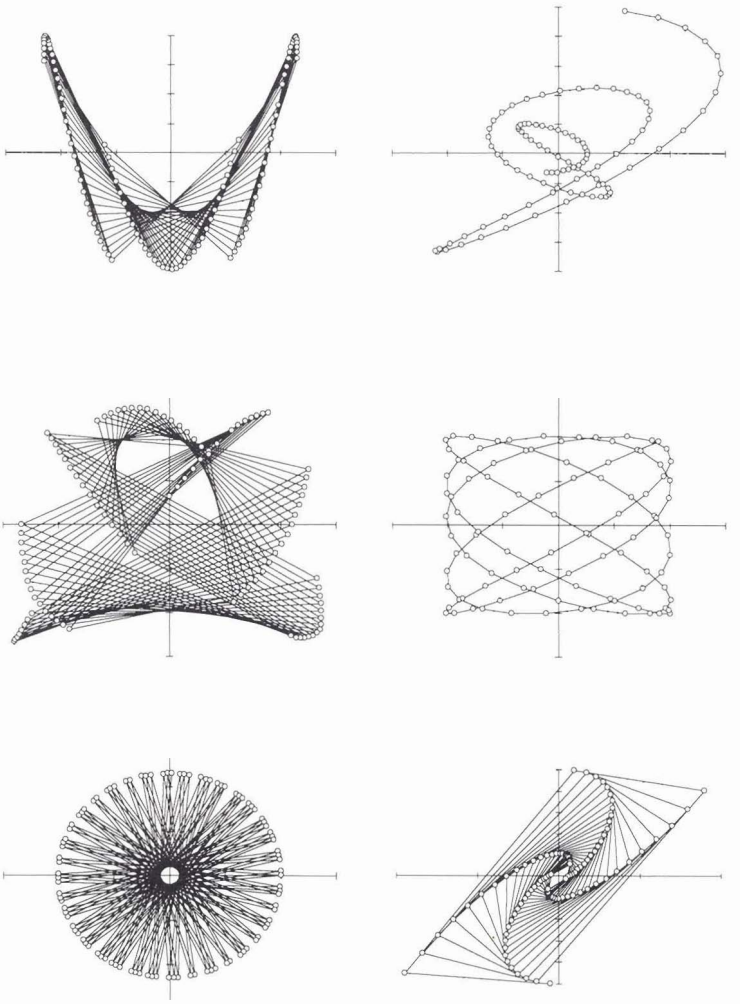

Fig. 3. Paths in weight space for backpropagation learning (linear associators).

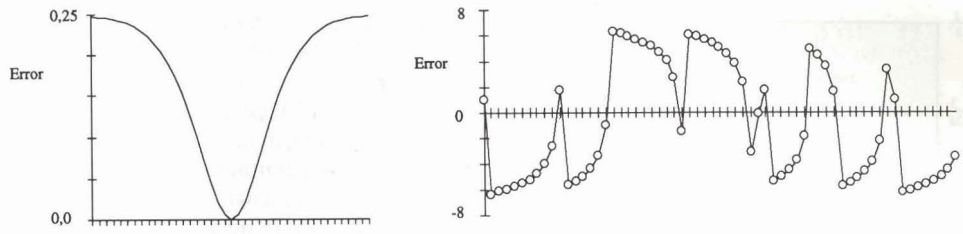

Fig. 4. Bounded nonlinear error function and the result of several iterations.

$1/2k$, the convergence speed is optimal for a unique α. The optimal combinations of α and γ are the ones represented by the jagged line in the diagram.

The more interesting message we get from Fig. 2 is the following: In the case where in some direction in weight space the principal axis of the error function is very small compared to another axis, we should try to achieve a compromise by adjusting the momentum rate in such a way that the oscillating directions become less oscillating and the directions with slow convergence improve their convergence speed. Obviously when dealing with n directions in weight space, this compromise could be dominated by a single direction in weight space.

4. CRITICAL PARAMETER COMBINATIONS

Backpropagation is normally used in those cases in which we do not have an analytic expression of the function to be optimized. A learning rate γ has to be chosen without any previous knowledge of the covariance matrix of the input. In on-line learning the training patterns are also not always defined in advance, and are generated one by one. A conservative approach is then trying to minimize the risk by choosing a very small learning rate. But in this case backpropagation can be trapped in a local minimum of a nonlinear error function. The learning rate should then be increased.

In the case of a covariance matrix $\mathbf{X}^T\mathbf{X}$ with some very large eigenvalues, a given choice of γ could lead to divergence in the associated direction in weight space (assuming for simplicity that the principal directions of the quadratic form are aligned with the coordinate axis). Let us assume that the selected gamma is near to the explosion point $2/k$ found in the one-dimensional case and shown in Fig. 2. In this case only values of the momentum term near to one can guarantee convergence, but oscillations in some of the directions in weight space can become synchronized. The result is oscillating paths in weight space, reminiscent of Lisajous figures. Figure 3 shows some paths in a two-dimensional weight space for several linear associators trained with momentum rates close to one and different γ values. In some cases the trajectories lead to convergence after several thousand iterations. In others a momentum rate equal to one precludes convergence of the iteration process. In many cases in which backpropagation does not converge, at least it diverges with elegance.

The adjustment of the learning and momentum rate in the nonlinear case is even more difficult than in the linear case, because there is no fast explosion of the iteration process. At least in the quadratic case whenever the learning rate is excessively large, the iteration process leads rapidly to an overflow that alerts the programmer that the step size should be reduced. But in the nonlinear case the output of the network and the error function are bounded and no overflow occurs there. In regions far from local minima the gradient of the error function nearly becomes zero and the weight adjustments also. The divergence regions of the quadratic case can now become oscillatory regions. In this case even with larger and larger step sizes the iteration returns to the convex part of the error function. Figure 4 shows the possible shape of the error function for a linear associator with sigmoidal output and the associated oscillation process for this kind of error function in the one dimensional case. The jagged form of the iteration curve is reminiscent of the kind of learning curves shown in many papers about learning in nonlinear neural networks.

5. CONCLUSIONS

In this paper we have shown that backpropagation with momentum can exhibit a highly oscillating behavior under some parameter combinations. Presumably this is the kind of situation found in some applications when the learning rate is just too large. Although in the quadratic case mainly large momentum rates lead to oscillations, in the nonlinear case a gamma which is excessively large can also produce oscillations even when no momentum rate is present.

Researchers in the field of neural networks should be concerned not only with the possibility of getting stuck in local minima of the error function when learning rates are too small, but also with the possibility of falling into the oscillatory traps of backpropagation when the learning rate is too big. Learning algorithms should try to balance the speed-up they are attempting to obtain with the risk of divergence involved in doing so. Two different kind of remedies are available: adaptive learning rates and statistical preprocessing of the learning set in order to decorrelate the input patterns trying to avoid the deleterious effect of too large eigenvalues of the covariance matrix [6]. In the case of highly correlated input patterns the analysis of the convergence regions for backpropagation becomes more

complex and the iteration paths become more irregular than those shown in Fig. 3.

REFERENCES

1. R. Rojas, *Theorie der neuronalen Netze*, Springer-Verlag, Berlin (1993).
2. H. van der Maas, P. F. Verschure, and P. Molenaar, A note on chaotic behavior in simple neural networks. *Neural Networks* **3**, 119–122 (1990).
3. J. D. Schaffer, D. Whitley, and L. J. Eshelman, Combinations of genetic algorithms and neural networks: A survey of the state of the art. In *International Workshop on Combinations of Genetic Algorithms and Neural Networks*, IEEE Computer Society Press, Los Alamitos, 1–37 (1992).
4. M. Pfister, On data correlation and neural networks. *Technical Report B-11/93*, Freie Universität Berlin, Fachbereich Mathematik und Informatik (1993).
5. R. A. Jacobs, Increased rates of convergence through learning rate adaptation. *Neural Networks* **1**, 295–307 (1988).
6. L. B. Almeida and F. M. Silva, Speeding-up backpropagation by data orthonormalization. In T. Kohonen, K. Mäkisara (Eds.), *Artificial Neural Networks*, North-Holland, Amsterdam, 943–948 (1991).