# An Evaluative Study of an Alternative to Back-propagation Algorithm for Neural Networks.

M. A. Saleh Elmohamed and Geoffrey Fox
Northeast Parallel Architectures Center,
Syracuse, NY 13244-4100
{saleh, gcf}@npac.syr.edu

ABSTRACT

This brief study focuses on a recently introduced algorithm called Chemotaxis. The biological plausibility, computational complexity, implementation on parallel machines, and the advantages and disadvantages of the algorithm are generally outlined.

This evaluation is not considered to be a survey nor is it considered to cover all aspects in or related to the algorithm in question. It is only meant to bring about what is "essential" about this method in light of the aforementioned criteria.

# Contents

# 1 Introduction

The major motivation behind this evaluation comes from an article by Francis Crick [6]. In his commentary, he gave a general description of the back-propagation training algorithm for feed-forward neural networks. He then, again in general terms, outlined the biological plausibility or lack of it of this algorithm. To bridge Crick's commentary with the topic of this paper, we would like to state some relevant quotes from his. He writes "*Its is hardly surprising that such achievements [training with back-prop] have produced a heady sense of euphoria. But is this what the brain actually does? Alas, the back-propagation nets are unrealistic in almost every respect, .... It is also extremely difficult to see how neurons would implement the back-prop algorithm. Taken at its face value this seems to require the rapid transmission of information backwards along the axon, that is antidromically to the synapses. It is unlikely that this happen in the brain ...*" . "*Obviously, what is really required is brain-like algorithm which produces results of the same general character as back-propagation.*"

In our opinion, an acceptable answer to the last statement is a recently introduced algorithm called *Chemotaxis* [7, 14] with resemblance to bacterial chemotaxis [3]. In his 1980 work, Koshland described many of the ideas that are the motivations of this algorithm.

# 2 Bacterial Chemotaxis, and Their Relations to Neurobiology

One possible question that could be asked in this context is: neurobiological systems involve interaction between many cells, but bacteria by nature are monocellular organisms; so, does the study of such systems have any relevance to neurobiology?

If we consider the brain as a giant computer with billions of cells interconnected with each other, then a fundamental understanding of the role of individual cells will not only be important per se but will also help in understanding the design of the overall system. Since we are willing to accept that the chemistry of the individual neurons is relevant to an understanding of the overall system, can we accept that the study of a bacterial cell is relevant to the understanding of the neuron?

The signaling system of the neuron and bacterium has *input* as well as *output* response. The neuron's output can be the release of a neurotransmitter or an electrical voltage. Since the neuron is far larger than the bacterium, it must use additional devices such as an action potential to transmit information over much larger distances. This is one of the distinguishing differences between the two cells. On the other hand, the bacterial system shows properties of adaptation, memory, receptor function, focusing of signals, and so on; which have high degrees of similarity to the equivalent processes of the neurons. One question that has not been answered is whether the processing unit is substantially more complex in a neuron than it is in a bacterium. If the answer is definitely "yes" then we could say that the study of the simpler system won't help us much understand the complex one.

Koshland [3] stated that “methylation” and “demethylation” utilized by the bacterial processing system is in ways similar to the use of “phosphorylation” and “dephosphorylation” in the mammalian system. Such analogies can strengthen the relation between the two systems. Since the model we are adapting here is of the bacteria, we need first to understand their behavior and then try to characterize it.

Summarizing from Koshland and Berg [3, 4]: the bacterium behavioral “response” permits them to migrate along the lines of the gradients, up or down, and this migration process is achieved by the control of “tumbling frequency”. Movements of a bacterium consist of either “tumbling” or “swimming”. When the direction of movement is considered favorable then they continue swimming in approximately straight line. When swimming, if their sensory system tells them such a direction is unfavorable (if attractant concentration does not increase), they generate tumbling and head in a new direction. It is not necessary that the new direction will be more favorable, because each tumble causes a random reorientation in the direction of movement. It has been shown by [4] that it is not the tumble itself that reorient the path of migration, but rather the frequency of tumbling.

So, what does a tumble do to a bacterium that was heading in the wrong direction? A tumble can have two possible outcomes:

1. It (the bacterium) can, by chance, continue swimming in the wrong direction, in which case it won’t move very far before it tumbles again.

2. If as a result of a tumble it reorients and starts swimming in a favorable direction, tumbling will be suppressed and the bacterium will swim a longer than normal distance.

The main point is that the overall tumbling frequency is sufficient to bias the “random walk” of the bacterium to move in a a favorable direction. One can consider this to be very inefficient, but if the sensory system can be selected so that the bacterium moves very short distances in the wrong direction and quite long distances in the right direction, the overall migration can involve little waste. The behavioral output of the processing system is either tumbling or no tumbling. Thus the output of the cell can be considered as an *on-off* phenomena such as the electrical discharge of a neuron. How the chemotaxis algorithm fits into the above behavior? To answer this question and determine what the algorithm implies and how it goes about modifying the synaptic weights, see next section of this report. The work of Koshland and Berg sheds more light on the relation between bacterial chemotaxis and neural systems. To help tie things nicely together in the next section, we’d like to outline some more relevant points.

One important question to ask from the above bacterial behavior is, what are exactly the responses required to direct migration favorably for bacteria swimming up spatial gradients of attractants (suppression of tumbling) or down the gradients of attractants (increase in tumbling)?

It has been shown that bacteria subjected to sudden temporal decrease in attractant concentration increase their tumble frequency and those subjected to sudden increase in attractant concentration are observed to suppress their tumbling frequency. In both cases the bacteria gradually return to their normal tumbling frequency. Also, it was stated that responses of cells of bacterium could be altered by environmental conditions such that some cells would amplify responses, other cells desensitize responses, and a third type of cells give pure additivity (outputting exact sum of incoming stimuli). From the neural networks point of view, this type of plasticity is exactly what one would look for when it comes to network learning. Furthermore, analogous to the cells of bacterium, we could claim that not all the units of a neural network are of the same type. That is, neurobiological plausible networks need to have "hybrid" structure. Also this structure can have a non-uniform pattern of connectivity.

An interesting inference from the work of Koshland and Berg is that, in the chemotactic response, stimuli are processed by a cell machinery that has certain constitutive factors and certain modifiable factors, such as induced receptors. More interesting than that is the feature that individual cell variations are not affected by heredity or environment or both. This last feature was only claimed but not proved.

To summarize; the neural system of the human brain is far more complex than the sensory system of bacteria. An exception to this may be the central processing of the bacterium which may be as complex as the processing of an adenyl cyclase system within a neuron. From this point we conclude by stating that a single bacterium cell can encompass many of the principles of higher behavioral systems, thus the analogy with neurons.

# 3 Chemotaxis Algorithm and Its Relation to Neurobiology

## 3.1 Definitions

**Definition 3.1** An $N \longrightarrow M \longrightarrow L \longrightarrow K$ is a feedforward neural network with $N$ (binary) inputs, $M$ hidden units in the first layer, $L$ hidden units in the second layer, and $K$ output units. Each unit (except those of the input layer) has a sigmoid response curve, given by the "squashing function":

$$f = \frac{1}{(1 + e^{-z})} \tag{1}$$

where z is the sum of the weighted inputs to the unit. □

**Definition 3.2** Let $w_{ijk}$ denote the synaptic weight of the k$th$ unit in the j$th$ layer with input $y_{(j-1)i}$ from the i$th$ unit in the previous $(j-1)st$ layer. Then

$$z_{jk} = \sum w_{ijk} y_{(j-1)i} \tag{2}$$

is the weighted sum of inputs to the $kth$ unit in the $jth$ layer. The thresholds are represented by $w_{0jk}$ and the convention $y_{0(j-1)} = -1$. The performance of the network is thus completely determined by the weights $w_{ijk}$, and measured as the sum of squared errors between target outputs $t_k$ and net outputs $o_k$, summed over "sample patterns" $p$. For more on this, see [1, 9]. □

**Definition 3.3** In back-propagation algorithm, the weight changes are determined as follows:

$$\Delta w(n) = \lambda \partial F + \mu \Delta w(n-1), \tag{3}$$

where $\partial F$ is the gradient of the error function, $\lambda$ is the learning rate and $\mu$ is a "momentum term" that adds a portion of the previous increment $\Delta w(n-1)$ to the gradient. The momentum term was introduced to overcome local minima. □

The main difference between the backpropagation algorithm and the chemotaxis algorithm (as we will see below) is in the determination of weight changes $\Delta w_{ijk}$. Taken together, the weights form a vector $w$ in weight space. The dimension of the weight space is the sum of all the weights and thresholds in the net. Training consists in a series of weight changes $\Delta w$ until the squared error function $F(w)$ is reduced to an acceptably small value.

## 3.2 Some Background on Search Algorithms

During a search one evaluates the objective function at chosen points. The points can be chosen systematically or at random. The way the points are chosen can make a big difference, especially when the dimension of the space that must be searched is high.

In a fully connected neural network we are talking about very large dimensions of the weight space, even for nets with few units. Our algorithms thus must search spaces of high dimension and exhaustive search is out of the question. This leaves only one option: Search through random sampling. This involves picking points at random, but "random" does not mean that they are chosen from a *uniform* probability distribution. Instead, if the currently lowest function is at $w^{(0)}$ we sample at points $w^{(0)} + \Delta w$, such that the steps $\Delta w$ are taken from s multivariate normal distribution (Gaussian distribution) with mean *zero* and standard deviation $H$. Here $H$ is a parameter that depends upon the objective function. It can be preset, or adjusted interactively or automatically while the algorithm is running. At each sample point $w^{(0)} + \Delta w$ the objective function is evaluated. If $F(w^{(0)} + \Delta w) < F(w^{(0)})$ then $w^{(0)}$ is replaced by $w^{(0)} + \Delta w$, if not then a new $\Delta w$ is generated and the process is iterated.

Each of these trials has a computational cost, namely evaluation of the objective function. The search effort should be organized such that the cumulative cost (number of function evaluations) of a search is minimized. The probability distribution of the $\Delta w$'s has a strong effect upon the length and possible failure of the search effort.

The importance of the Gaussian distribution, especially in high dimension, can easily be demonstrated by substituting another distribution such as a uniform distribution, If we choose the individual trial modifications of the synaptic weights independently from a uniform distribution we obtain in $N$-space a distribution that is uniform over the $N$-cube.

## 3.3 On Random Walk Algorithms and Chemotaxis

Algorithms, involving random walks, have long been the numerical method of choice for many particle problems of mathematical physics. Ceperly and Alder [10] reported that the many particle Schroedinger equation can be solved by a "quantum Monte Carlo method" that solves the equation by random walks in the many-dimensional state space. In the limit, for short stepsize and many trajectories, random walks are described by *diffusion equation* [3, 5, 4, 2]. Chemotaxis, too, in the limit is described by a reaction-diffusion equation [11, 12, 13]. Ceperly and Alder note that that the many-particle Schroedinger equation, written in Euclidean time, is equivalent to a bacterial chemotaxis equation.

## 3.4 On the Activation Function

Unless it is stated otherwise, we will be using the *sigmoid* activation function (sometimes called "squashing function"). They are the kind of functions used for applying back-propagation, which computes the gradient of the performance (error) function. Also, sigmoid functions are highly smooth, continuous, differentiable everywhere and widely used in neurophysiology.

## 3.5 The Chemotaxis Algorithm

As previously mentioned, this algorithm resembles bacterial chemotaxis. Also, random trial steps are taken with a Gaussian distribution, but when the algorithm takes a successful step, it attempts to keep going in the same direction until the objective function no longer decreases. In other words, this algorithm will attempt to follow a declining slope if it finds one.

1. **Initialization**:

   Set weights and node offsets to small random values.

2. **Present input and calculate outputs**:

   For a feed-forward network,

$$y_j = f(z_{jk}) = \frac{1}{(1 + e^{-(z_{jk})})} \tag{4}$$

   where

$$z_{jk} = \sum w_{ijk} y_i \tag{5}$$

3. **Determine "energy" of the objective function**:

$$E_{error} = \sum_{p}^{P} \sum_{k}^{K} (t_k - o_k)^2 \tag{6}$$

where: $t_k$ are the target outputs, $o_k$ are the calculated outputs, $K$ is the number of output nodes, and $P$ is the number of input vectors (sample patterns).

4. **Generate random vector**:

   $\Delta\vec{w}$ is drawn from a Gaussian distribution with mean $\mu = 0$ and standard deviation $\sigma = 1$.

5. **Randomly modify weights**:

   Tentative weight change:
$$t\vec{w} = \vec{w} + H\Delta\vec{w} \tag{7}$$

6. **Calculate objective function for tentative weight** ($E_temp$):

   $E_temp$ is generated by summing the squared output errors over the entire set of inputs.

7. **Compare new value with old**:

   *If* $E_{temp} < E_{error}$, fix weight changes and continue along current ray.

   - $\vec{w} \leftarrow t\vec{w}$ ,
   - Go to *step (5)* above.

   *else*,

   - do not change $\vec{w}$,
   - Go to *step (4)* above.

8. **Adjust the stepsize**($H$):

   $H$ is the stepsize parameter used in *step (5)* above.

   - Set $I_{success}$ to a value between 2 and 6, set $I_{failure}$ to a value between 12 and 30.
   - When the criterion $E_{temp} < E_{error}$ is true $I_{success}$ times for the same random ray, the parameter $H$ was increased. When $E_{temp}$ had failed more than $I_{failure}$ times, the parameter $H$ was reduced.

## 3.6 Analysis of the above algorithm

The range of random weights in the initialization step (step (1)) is between 0 and 1. A connection weight carrying a 0 value is inactive connection. A major reason of having initial weights that are small in value is to them within a reasonable and a bounded range throughout the training.

In addition to what is stated above on the activation function, in back-propagation, the shape of the sigmoid signal function implies that weights $w_{ijk}$ will stop growing if the magnitude of the activity $z_{jk}$ becomes too large, since then the derivative term $f^{'}(z_{jk})$ goes to zero.

In back-propagation, the objective of training is to minimize the objective function; thus reaching at an appropriate level of learning. What does this function or "phenomenon" correspond to in the bacterial chemotaxis world? Well, from the relation between the sensory system and the bacterial chemotaxis world, [3, 4] have indicated that when bacteria are swimming in a medium with varying concentrations, they generate random directions instead and keep going as a concentration increases or the "energy" or objective function decreases. If attractant concentration does not or no longer increases, then they stop, tumble, then emerge in a new direction at random angle to the old direction. In this way they move toward larger and larger concentrations values of the attractant and in this process they minimize the energy function. In other words, they optimize the "function" that describes the concentration of the chemo-attractant in the medium. Koshland stated that there is an analogy to this in the neural system. Moreover, there are a number of studies correlating membrane potential with changes in the bacterium tumbling frequencies. This may may have some significance since membrane potential can feed signals to the central processing system.

On generating a random vector (step (4)), for simplicity, if we assume that we are dealing with a one-dimensional space (instead of 3-D) in which the bacteria swim, hence where the weight vectors are chosen, we will have the following Gaussian probability distribution:

$$prob(x, c_i) = \frac{e^{\frac{(-0.5(x-\mu_i)^2)}{\sigma_i^2}}}{\sigma_i\sqrt{2\pi}} P(c_i) \tag{8}$$

where $x$ corresponds to the weight vector $\Delta\vec{w}$ that need to be drawn, and $c_i$ corresponds to the class of weight vectors representing the space where the bacteria swim. If the mean $\mu_i = 0$ and the standard deviation $\sigma_i = 1$ of the distribution, equation (8) can be re-written as:

$$prob(x, c_i) = \frac{e^{(-0.5x^2)}}{\sqrt{2\pi}} P(c_i) \tag{9}$$

which depends on the magnitude of $P(c_i)$. Usually, this magnitude is either given or known before hand. Also, we tend to select $x$ or $(\Delta\vec{w})$ such that $prob(x, c_i)$ is maximal.

The case of random modification to the weights (step (5)) is closely corresponds to the randomness phenomena exhibited by the bacteria and how they produce what is called "random walks". The use of parameter $H$ (in the range between 0.0 and 1.0) is similar to the step size parameter used in back-propagation except $H$ is modified dynamically. Between 0.0 and 1.0 is a region where the weight adjustment is smoothed by an amount proportional to $H$. The use of "tentative" change to weights is very much analogous to bacterial behavior stated in section (2), items (1 and 2).

In step (7) of the Chemotaxis algorithm, if $E_{temp} < E_{error}$ then it must be that after the tumble the bacterium is swimming in the right direction, since the chemo-attractant concentration is increasing. So, it is perfectly reasonable to make those "recently computed" tentative weights to be "permanent"and continue swimming along the same direction. $E_{temp} \geq E_{error}$ means that the chemo-attractant does not or no longer increases, hence retract the "recently computed" tentative weight and go to step (4). In summary, the relation between $E_{temp}$ and $E_{error}$ is similar to a bacterium swimming until it arrives at a place where the gradient no longer changes or its receptors were saturated and then would swim in a random manner in that locality. It is quite possible that it may make a wrong turn that will take it out of the locality. Sensing it is swimming in the wrong direction, it will increase its probability of tumbling and so in a very short interval. We could say (as indicated by Koshland) that bacteria "random walk" is heavily biased towards movement in favorable direction and its probability of responding in an appropriate manner is biased the instant it starts up or down a gradient.

### 3.7 Computational Complexity of the Algorithm

Steps (2, 4, 5, and 6) carry the highest time and space cost. Computing the sigmoid function is expensive whether it is done on sequential or massively parallel computers. This situation is similar for steps (2) and (6) because they involve computing exponentiation. Since step (2) is computed only once, the major cost lies in steps (4) and (6). If each is computed $N$ times then

$$cost = N(cost(step(4)) + cost(step(6))) + M \tag{10}$$

where $M$ is $(E_{temp}, E_{error})$ comparison cost which is approximately less than or equal to $N$. If the sigmoid is substituted with other less expensive function(s), then the overall cost of the Chemotaxis algorithm, on the average, is much less than the cost of back-propagation.

### 3.8 Parallel Implementation

It can be said that this is no more difficult than implementing back-propagation training algorithm on parallel computers. Steps (4 and 5) of the algorithm can be considered to have global effect, so in these steps can easily be computed on a massively parallel or distributed computing environment.

## 4 Advantages and Disadvantages of the Algorithm

### 4.1 Advantages

- Unlike back-propagation, there is no need for parameter optimization in Chemotaxis algorithm.
- It automatically adjust the stepsize $H$.
- Restriction of the number of hidden units didn't appear to present a significant obstacle to the algorithm. This was shown when the algorithm was tested on the encoder problem [8].

- Generally for nets with three or more layers of hidden units, Chemotaxis matches or outperforms back-propagation.

- Best of all, no error calculations or backward propagation of any value.

### 4.2 Disadvantages

- For nets with one hidden layer, it is slower than back-propagation.

- Its generalization ability is weak when the training set is small and I believe the same can be said about back-propagation algorithm.

- Chemotaxis worsen in performance as the dimension of the training vector increases.

Aside from these, the algorithm is flexible enough to allow dynamic net expansion (adding more hidden units) and net contraction. When training reached a prolonged plateau, the algorithm could be made to converge by training a smaller net, then expanding it and training on the full problem again.

## 5 Conclusion

We have presented an evaluation of an alternative to back-propagation for neural networks training. By no mean do we claim that this is a complete evaluation. As a matter of fact, it is not, especially when it comes to the question of biological plausibility of the method.

## Acknowledgment

My thanks to Professor Evan Relkin of the Sensory Research Institute at Syracuse University for fruitful comments and discussion on this work.

# References

[1] G. Tesauro and B. Janssens, "Scaling Relationships in Back-propagation Learning," *Complex Systems*, 1988, 2: 39-44.

[2] A. Okubo, Diffusion and Ecological Problems: Mathematical Models. vol. 10., New York, Springer Verlag, 1980. (*Biomathematics.*)

[3] D. Koshland, Bacterial Chemotaxis as a Model Behavioral System. New York, Raven Press, 1980.

[4] H. Berg, Random Walks in Biology. Princeton, Princeton University Press, 1983.

[5] H. Berg, "How Bacteria Swim," *Scientific American*, 1975, 233(2): 36-44.

[6] F. Crick, "The Recent Excitement about Neural Networks," *Nature*, 12 January, 1989; 337: 129-132.

[7] H. Bremermann and R. Anderson, "An Alternative to Back-Propagation: A Simple Rule of Synaptic Modification for Neural Net Training and Memory," PAM-483, Center for Pure and Applied Mathematics, Univ. of California, Berkeley, January 1990.

[8] D. Ackley, G. Hinton, T. Sejnowski, "A Learning Algorithm for Boltzmann Machines," *Cognitive Science*, 1985, 9: 147-169.

[9] D. Rumelhart, J. McClelland, G. Hinton, "Learning Internal Representations by Error Propagation," In: Rumelhart and McClelland, eds. *Parallel Distributed Processing Vol.I.* Cambridge, MA: MIT Press 1986, 318-362.

[10] D. Ceperley, B. Alder, "Quantum Monte Carlo," *Science*, 7 Feb 1986; 231: 555-560.

[11] E. Keller, L. Segel, *Journal of Theoretical Biology*, 1970, 26: 399.

[12] W. Alt, "Biased Random Walk Models for Chemotaxis and Related Diffusion Approximation," *J. Mathematical Biology*, 1980. 9: 147-177.

[13] R. Nossal, Mathematical Theories of Topotaxis. *Lecture Notes in Biomathematics*, 38: 410-439.

[14] A. Barto, "From Chemotaxis to Cooperativity: An Abstract Exercises in Neuronal Learning Strategies," in the Computing Neuron, Durbin, Miall, and Mitchison (eds.), Addison-Wesley, 1989.