Communicated by Geoffrey Hinton

Contrastive Learning and Neural Oscillations

Pierre Baldi
Jet Propulsion Laboratory and Division of Biology, California Institute of Technology, Pasadena, CA 91125 USA

Fernando Pineda
Applied Physics Laboratory and Department of Electrical and Computer Engineering, The Johns Hopkins University, Baltimore, MD 21218 USA

The concept of Contrastive Learning (CL) is developed as a family of possible learning algorithms for neural networks. CL is an extension of Deterministic Boltzmann Machines to more general dynamical systems. During learning, the network oscillates between two phases. One phase has a teacher signal and one phase has no teacher signal. The weights are updated using a learning rule that corresponds to gradient descent on a contrast function that measures the discrepancy between the free network and the network with a teacher signal. The CL approach provides a general unified framework for developing new learning algorithms. It also shows that many different types of clamping and teacher signals are possible. Several examples are given and an analysis of the landscape of the contrast function is proposed with some relevant predictions for the CL curves. An approach that may be suitable for collective analog implementations is described. Simulation results and possible extensions are briefly discussed together with a new conjecture regarding the function of certain oscillations in the brain. In the appendix, we also examine two extensions of contrastive learning to time-dependent trajectories.

1 Introduction

In this paper, we would like to develop the concept of Contrastive Learning (CL) as a family of possible learning algorithms for arbitrary convergent dynamical systems. CL is an extension of Deterministic Boltzmann Machines (Peterson and Anderson 1987) and Contrastive Hebbian Learning (Movellan 1990). Deterministic Boltzmann Machines are mean field approximations to Boltzmann Machines (Ackley *et al.* 1985). Contrastive Hebbian Learning is essentially a different method for deriving a Hebbian learning rule for Deterministic Boltzmann Machines. It is equivalent to the observation in Hinton (1989), based on a geometric argument, that Deterministic Boltzmann Machines perform gradient descent on a suitably defined cross-entropy function. The mathematical approach given here makes very few assumptions concerning the details of the activation dynamics beyond the requirement that it be a gradient dynamics. It is, therefore, a general approach that illuminates the common structure of Deterministic Boltzmann Machines and their variants. It also allows one to derive, almost by inspection, new learning algorithms for particular gradient systems. These algorithms are not necessarily Hebbian.

Consider the problem of training a neural network to associate a given set of input–output pairs. In the course of CL training, the network is run in alternation with and without a proper teacher signal applied to some of its units. The weights in the network are updated so as to reduce the discrepancy between the steady-state behavior of the free network (without teacher) and the forced network (with teacher). As we shall see, this waxing and waning of the teacher can also be approximated with continuous oscillations.

In Section 2, we describe CL for a general class of convergent dynamical systems. For clarity, the reader may want to particularize some of the statements to the usual additive neural network model with symmetric zero-diagonal interactions (see, for instance, Hopfield 1984) with

$$\frac{du_i}{dt} = -\frac{u_i}{\tau_i} + \sum_j w_{ij} V_j + I_i \tag{1.1}$$

and energy function

$$E^f(V, I, W) = -\frac{1}{2}\sum_{i,j} w_{ij} V_i V_j + \sum_i \frac{1}{\tau_i}\int_{\text{rest}}^{V_i} g^{-1}(v)\, dv - \sum_i I_i V_i \tag{1.2}$$

Throughout the paper, the superscripts f ("free") and t ("teacher") are used to distinguish quantities in the free system and in the system with teacher. In Section 3, we give four different examples of CL that go beyond the special case of Deterministic Boltzmann Machines and the additive neural network model. In Section 4, we analyze the landscape of the contrast function, argue that it is characterized by three types of regions, and make some predictions about CL curves. In Section 5, we present an approach that may be suitable for collective analog implementations and that uses oscillations to continuously approximate CL. We conclude in Section 6 with the results of some preliminary simulations and a few possible extensions and open questions. Finally, in the appendix we sketch how CL may be extended from learning fixed points to learning trajectories.

2 Contrastive Learning

To be more precise, consider an arbitrary convergent n-dimensional dynamical system, where the states are described by the vector $u = (u_1, \ldots, u_n)$.

The parameters of the dynamical system can be classified into two classes: the external parameters I and the internal parameters W. Only the internal parameters W are subject to modification in the course of learning. In the case of neural networks, the states u_i can be thought of as representing the membrane potentials of the units, while the vector $I = (I_1, \ldots, I_n)$ and the array W represent the external inputs and the connection weights, respectively (in addition, W may also include various gains and time constants). We assume that the convergent dynamical system is a gradient system governed by an energy function[1] $E^f(V, I, W)$ so that

$$\frac{du_i}{dt} = -\frac{\partial E^f}{\partial V_i} \tag{2.1}$$

with $V = (V_1, \ldots, V_n)$ and $V_i = g(\lambda u_i)$, where g is a monotonically increasing function such as the identity or one of the usual sigmoid transfer functions used in neural models (V_i can be interpreted in terms of firing rates). λ is a parameter corresponding to the gains or the temperature in the system. Here, and in the rest of the paper, all functions are assumed to be continuously differentiable, unless otherwise specified. Out of the n variables $V_1, \ldots, V_n$, the first l $V_1, \ldots, V_l$ are considered output variables. The goal of the learning algorithm is to adjust the internal parameters so that for a given fixed initial state $u(0)$ and external input I, the free system converges to a stable state where the output variables have the target values $T = (T_1, \ldots, T_l)$. For simplicity, we are dealing here only with one association $I \to T$ (the generalization to many input–output pairs by averaging in the usual way is straightforward). To train the free system, we introduce a second "forced" dynamical system. In the forced system the energy $E^t(V, I, W, T)$ has the form

$$E^t = E^f + F(V, I, W, T) \tag{2.2}$$

The function F denotes a generalized teacher forcing term. The function $F(V, I, W, T)$ is not completely arbitrary and is chosen such that $F(V, I, W, T) = 0$ if $T_i = V_i$ over all visible units. In addition, F must be continuously differentiable and bounded with respect to the Vs so that the function E^t governs the gradient dynamics of a convergent system defined by

$$\frac{du_i}{dt} = -\frac{\partial E^t}{\partial V_i} \tag{2.3}$$

The equilibrium point of the free and forced systems are denoted by V^f and V^t, respectively. At a fixed point, V becomes an implicit function of I, W, and T.

[1]Even more generality could be achieved by using an equation of the form $du/dt = -A(u)\nabla E$, where the matrix A satisfies some simple properties (see, for instance, Cohen and Grossberg 1983). The dynamics could be "downhill" without following the gradient.

If H^t and H^f are two similar functions defined on the Vs in the forced and free systems, then the difference in behavior between the two systems can be measured by taking the difference $H^t(V^t) - H^f(V^f)$ [or alternatively, as in the usual LMS approach, by using a function $H(V^t - V^f)$]. In contrastive learning, the internal parameters are adjusted by gradient descent so as to minimize this difference. A particularly interesting choice of H, which will be used in the rest of the paper, is when H is the energy function of the corresponding system. The main reason for that, in addition to being a natural extension of the Deterministic Boltzmann Machine algorithm, is that it leads to very simple learning rules. Indeed, we can define the contrast function C by

$$C(I, W, T) = E^t(V^t, I, W, T) - E^f(V^f, I, W) \tag{2.4}$$

The contrastive learning rule modifies an adjustable internal parameter w by

$$\Delta w = -\eta \frac{\partial C}{\partial w} = -\eta \left(\frac{\partial C}{\partial w} \right)_{\text{explicit}} \tag{2.5}$$

where η is the learning rate. To see the second equality in equation 2.5, notice that

$$\frac{\partial E^\gamma}{\partial w} = \left(\frac{\partial E^\gamma}{\partial w} \right)_{\text{explicit}} + \sum_k \frac{\partial E^\gamma}{\partial V_k^\gamma} \frac{\partial V_k^\gamma}{\partial w} \tag{2.6}$$

where γ is either f or t. Now at equilibrium $\partial E^\gamma / \partial V_k^\gamma = -du_k^\gamma/dt = 0$. Therefore *the implicit terms do not contribute to equation 2.5.* The fact that only the explicit dependence enters into the gradient means that the learning rule is simple and can often be written down from the contrast function by inspection.

It is essential to notice that the derivation of equation 2.5 is purely heuristic. In general, the contrast function C is not necessarily bounded in W. If, over the range of operation, both $E^f(V^f, I, W)$ and $E^t(V^t, I, W, T)$ are convex in W, then equation 2.5 achieves the desired goal but this is certainly a very unrealistic assumption. A more detailed treatment of the properties of the contrast function will be given in Section 4.

CL is based on successive alternating relaxations of a free and a forced system. The initial state of the activation dynamics must be set before each relaxation. In general, the state to which the activation dynamics is reset depends on the task being performed. There are two classes of tasks. A parametric input task is one where the initial state of the network is always the same (usually the origin) and the input information is fed in through the vector I. An initial state task, on the other hand, has the input information fed in as part of the initial state and I is always the same. Here, we concern ourselves with parametric input tasks only. Accordingly, the activation of the network is reset to zero before each relaxation. The CL approach, however, can also immediately be extended to the case of initial state tasks by initializing the fixed and forced systems in similar pattern-dependent ways and computing the contrast function and its gradient in the way described above.

3 Examples

3.1 Competitive Systems. A useful class of competitive dynamical systems, discussed in Cohen and Grossberg (1983), can be described by

$$\frac{du_i}{dt} = a_i(u_i)\left[-b_i(u_i) + \sum_j w_{ij}g_j(u_j) + I_i\right] \tag{3.1}$$

where $W = (w_{ij})$ is symmetric. Equation 3.1 includes as special cases the additive model and also a number of models from population biology. If we define the matrix

$$A_{ij}(u_i) = \frac{a_i(u_i)}{g_i'(u_i)}\delta_{ij} \tag{3.2}$$

then equation 3.1 is gradient dynamics of the form

$$\frac{du}{dt} = -A(u)\nabla E(u) \tag{3.3}$$

with energy function

$$E^{\mathrm{f}}(V, I, W) = -\frac{1}{2}\sum_{i,j} w_{ij}V_iV_j + \sum_i \int_0^{u_i} b_i(u)g_i'(u)\,du - \sum_i I_iV_i \tag{3.4}$$

The corresponding forced energy function is given by equation 2.2 where $F(V, I, W, T)$ is any convenient forcing term. If F has no explicit W dependence, the CL rule is simply

$$\Delta w_{ij} = \eta\left(V_i^{\mathrm{t}}V_j^{\mathrm{t}} - V_i^{\mathrm{f}}V_j^{\mathrm{f}}\right) \tag{3.5}$$

The form of this learning rule is identical to that used in Deterministic Boltzmann Machines (DBM) (Peterson and Anderson 1987) and in Contrastive Hebbian Learning (CHL) (Movellan 1990). Both DBM and CHL are based on the additive model [$a_i(u_i) = 1$ and $b_i(u_i) = u_i$] and the learning rule $\Delta w_{ij} = \eta(V^{\mathrm{c}}_iV^{\mathrm{c}}_j - V^{\mathrm{f}}_iV^{\mathrm{f}}_j)$ where V_i^{c} denotes the equilibrium activities in the network where the output (and input) units are clamped to their target values.

To show that DBM are a special case of the general framework, we need to interpret clamping in terms of a teacher forcing term. This is easily done by writing

$$\frac{du_i}{dt} = -u_i + g_i^{-1}(T_i) \qquad \text{for } i = 1, \ldots, l \tag{3.6}$$

which relaxes to $u_i = g^{-1}(T_i)$.[2] It is simple to check that the network with such corresponding teacher has the energy function

$$\begin{aligned} E^{\mathrm{t}}(V, I, W, T) &= E^{\mathrm{f}}(T_1, \ldots, T_l, V_{l+1}, \ldots, V_n, I, W) \\ &\quad - \sum_{i=1}^{l} [u_i - g^{-1}(T_i)] V_i \end{aligned} \tag{3.7}$$

By applying equation 2.5 to the corresponding contrast function, one immediately gets equation 3.5 with in fact $V_i^{\mathrm{c}} = V_i^{\mathrm{t}} (= T_i$ for $i = 1, \ldots, l)$.

The previous considerations extend immediately to neural networks consisting of higher order (or $\Sigma\Pi$) units with the proper symmetric interconnections. For instance, in the case of third-order interactions one can replace equation 3.5 by $\Delta w_{ijk} = \eta(V_i^{\mathrm{t}} V_j^{\mathrm{t}} V_k^{\mathrm{t}} - V_i^{\mathrm{f}} V_j^{\mathrm{f}} V_k^{\mathrm{f}})$. These examples can also be extended to networks of simple or higher order threshold gates with the proper modifications.

3.2 Simple Error Teacher Signal. In this example, we consider a particularly simple form of teacher term in the additive model. The free network satisfies equations 1.1 and 1.2. In the forced network, the activation equation contains a teacher signal which is a simple measure of the error and is given by

$$\frac{du_i}{dt} = -\frac{u_i}{\tau_i} + \sum_j w_{ij} V_j + I_i + \gamma_i (T_i - V_i)^{\alpha} \tag{3.8}$$

where γ_i may be nonzero only for the output units and α is a positive parameter. The associated energy function is

$$\begin{aligned} E^{\mathrm{t}}(V, I, W, T) &= -\frac{1}{2} \sum_{i,j} w_{ij} V_i V_j + \sum_i \frac{1}{\tau_i} \int_{\mathrm{rest}}^{V_i} g^{-1}(v)\, dv \\ &\quad - \sum_i I_i V_i + \sum_{i=1}^{l} \frac{\gamma_i}{\alpha + 1} (T_i - V_i)^{\alpha+1} \end{aligned} \tag{3.9}$$

For simplicity, we shall assume that on the output units $\gamma_i = \gamma$. Different values of the parameter α yield different models. For instance, when $\alpha = 1$ we obtain the usual LMS error term in the energy function. A value of $\alpha = 1/3$ has been used in relation to the terminal attractors approach (Zak 1989). By applying equation 2.5, the corresponding CL rule is $\Delta w_{ij} = \eta(V_i^{\mathrm{t}} V_j^{\mathrm{t}} - V_i^{\mathrm{f}} V_j^{\mathrm{f}})$. Notice that in contrast to DBM where the output units are clamped to their target values, the output units here may relax to fixed points of equation 3.8 satisfying $V_i^{\mathrm{t}} \neq T_i$. Of course, as $\gamma \to \infty$, $V_i^{\mathrm{t}} \to T_i$ $(i = 1, \ldots, l)$. It is worth noticing that in this example (as in the previous ones), the teacher forcing term F does not depend

[2]Alternatively, to prevent the initial relaxation of the clamped units, one could write $du_i/dt = 0$ and require the initial conditions to be $u_i(0) = g^{-1}(T_i)$, $i = 1, \ldots, l$.

explicitly on W. Thus the expression of the resulting CL rule is identical for an entire family of systems corresponding to different values of the parameters α and γ. However, these values affect the learning implicitly because the corresponding systems have different dynamics and relax to different fixed points while being trained.

3.3 Coupled Oscillators Models. This fourth example is mainly meant to illustrate the CL formalism on a different class of models for networks of coupled oscillators (see, for instance, Baldi and Meir 1990 and references therein) used recently in conjunction with the study of oscillatory brain activity. Networks of coupled oscillators can be studied by representing each oscillator by a single variable, its phase u_i. The oscillators are associated with the vertices of a graph of interactions. Each edge in the graph corresponds to a symmetric coupling strength w_{ij}. One possibility is to model the evolution of the phases by the system of equations

$$\frac{du_i}{dt} = \sum_j w_{ij} \sin(u_j - u_i) \tag{3.10}$$

The corresponding energy function is

$$E^f(u, W) = -\frac{1}{2} \sum_{i,j} w_{ij} \cos(u_i - u_j) \tag{3.11}$$

If we let T_i denote a set of target angles, then a possible associated forced system can be described by

$$\frac{du_i}{dt} = \sum_j w_{ij} \sin(u_j - u_i) + \gamma \sin(T_i - u_i) \tag{3.12}$$

with

$$E^t(u, W) = -\frac{1}{2} \sum_{i,j} w_{ij} \cos(u_i - u_j) + \gamma \sum_i \cos(T_i - u_i) \tag{3.13}$$

By inspection, this results in the CL learning rule

$$\Delta w_{ij} = \eta[\cos(u_i^t - u_j^t) - \cos(u_i^f - u_j^f)] \tag{3.14}$$

In some sense, this learning rule is still Hebbian since if we use the complex exponential $z_k = e^{iu_k}$, then $\Delta w_{kl} = \eta \, Re \, [z_k^t \bar{z}_l^t - z_k^f \bar{z}_l^f]$.

In many examples, the CL rule is a very simple Hebbian one and this is attractive from a hardware perspective. It should be noticed, however, that this is a consequence of the fact that the explicit dependence of the energy function on the weights rests on a quadratic form. As can be seen from the examples of higher order DBM and coupled oscillator models, more complicated learning rules can be derived by introducing different terms in the energy function with an explicit dependence on w_{ij}.

The previous examples also show that the notion of clamping a unit is not a precise one and that different algorithms can be defined depending on the degree of softness contained in the clamping. In hard clamping, all the variables pertaining to one output unit are kept fixed at their target values and only these target values appear in the contribution of the unit to the learning rule. In soft clamping, some of the variables may evolve. In fact, by varying the parameter γ in Section 3.2, one can easily envision a continuum of possible clampings. The description we have given of a DBM is based on hard clamping. However, one can conceive a softer DBM, for instance, where the value V_i of any output unit is held constant at a value $V_i^c = T_i$ while the network relaxes. The internal states u_i of the output unit may then evolve according to equation 1.1 and equilibrate to a final value u_i^t (such a network always reaches a fixed point although, strictly speaking, it has no energy function). $V_i^t = g(u_i^t)$ may then be used to adjust the weights rather than the clamped value V_i^c.

4 The Landscape of the Contrast Function

We shall now analyze the typical landscape of the contrast error function $C(W)$ as a function of the internal parameters W. For simplicity, we shall deal with the case of Deterministic Boltzmann Machines with hard clamping or with their version with a softer teacher term given by equation 3.8. We shall argue that, in general, the landscape of C contains three interesting and characteristic regions. A region corresponding to the initial stage of learning characterized by rapid progress and smooth descent. A region corresponding to an intermediary stage possibly characterized by abrupt discontinuities due to basin hopping phenomena. A third region associated with a final stage, found in the neighborhood of an optimal set of weights W, characterized by local convexity and smooth descent.

To begin with, it should be noticed that the contrast function is an average of several contrast functions, one for each pattern. To emphasize this important point and only in this section, we shall use a p superscript to denote the pattern dependence. Thus, for instance, $C = \sum_p C^p = \sum_p E^{tp}(V^{tp}) - E^{fp}(V^{fp})$. Furthermore, even for one pattern, the contrast function $C^p(W)$ is not bounded and is *not* continuous everywhere because there are values of W for which V^{tp} or V^{fp} vary abruptly. If one tries to learn a unique association pair by CL, the descent, however, is in general smooth. It can easily be seen that the contrast function is continuous over the restricted region covered by the corresponding descent procedure. Yet, it is when we try to learn several associations simultaneously and satisfy somewhat conflicting constraints that gradient descent leads us to regions of the parameter space where the contrast functions corresponding to the individual patterns may be discontinuous. We shall call a fracture any point or connected set of points where

C is discontinuous (that is where at least one of the C^p is discontinuous and the corresponding V^{tp} or V^{fp} varies abruptly). Thus a fracture is a point or a set of connected points in weight space associated with a basin boundary going through the initial state (usually the origin) in the activation space of either the free or forced system for at least one of the patterns (see also Pineda 1988). In general, basin hopping can result from abrupt disruption in the flow of the system at bifurcation points or from the smooth evolution of basin boundaries in the course of learning. Notice that when a bifurcation occurs and a new fixed point is created, the newly created basin boundaries may cross the origin only some time after.

In the initial stage, when training is started with very small initial weights, the units in the network operate near their linear regime and, for each pattern, both the free and forced networks have a unique fixed point. Thus, for each pattern p and as the weights W begin to evolve, V^{tp} and V^{fp} vary continuously. Thus, for small weights, each $C^p(W)$ is continuous and differentiable. Thus the total contrast function $C(W)$ is also continuous and differentiable and the learning curve decreases smoothly. This smooth descent lasts at least until the first bifurcation occurs in one of the networks corrresponding to one of the patterns. The first bifurcation creates the first additional fixed points and therefore the first basin boundary capable of causing a discontinuity.

In the intermediary stage, which is probably the most crucial for successful learning, the conflict introduced by the different patterns can become apparent. The learning trajectory may run into and cross fractures. Every time this happens, the contrast function jumps abruptly up or down. We believe that such a stage is inevitable in any reasonably challenging problem (for instance, we have seen no discontinuities in the case of one, two, or even three pattern learning from the XOR table; on the other hand, these tasks are easy and linearly separable) and the phenomenon is accentuated by large learning rates, noise, or increasing approximations to the gradient. It remains to analyze what happens in the proximity of an optimal set of weights W^* that achieves perfect learning, that is, such that, for each pattern, the equilibrium values of the output units in the free and forced network are identical and identical to the targets. In addition, at an optimum $W = W^*$ we can also assume that the hidden units in the free and forced systems equilibrate for each pattern to the same values. Configuration of optimal weights without this property could conceivably exist. These would likely be unstable when gradient descent updates are performed after each pattern (this is because $\partial C^p/\partial w_{ij} = V_i^{tp}V_j^{tp} - V_i^{fp}V_j^{fp} \neq 0$ for many patterns although $\partial C/\partial w_{ij}$ may be 0 on the average).

So, with these assumptions, for an optimal set of parameters W^*, $C^p(W^*) = 0$ for every p and therefore $C(W^*) = 0$. Yet, the converse is not necessarily true. Now, it is reasonable to consider that no fracture passes through W^*. Otherwise, the problem would be inherently intractable. It would mean that the set of training patterns cannot be loaded on the chosen architecture in any stable way or, equivalently, that there is no region in weight space with a proper interior that contains an optimal solution. Thus, if we assume that the network is capable of implementing the given function, each C^p and C must be continuous in a neighborhood of W^*. It is easy to see that this implies that each C^p and C is also differentiable at W^*. From the form of the CL rule, it is obvious that if W^* is optimal, then $\partial C^p/\partial W = 0$ for each p (and $\partial C/\partial W = 0$) at W^*. In the case of Deterministic Boltzmann Machines with hard clamping the converse is also true. Indeed, let us assume that $V_i^{cp}V_j^{cp} = V_i^{fp}V_j^{fp}$ for every p and every i and j. Since the inputs are clamped to the same values in the free and clamped network, by simple propagation through the network this implies that $V_i^{cp} = V_i^{fp}$ everywhere and therefore we are at an optimum. In the case of softer forms of Deterministic Boltzmann Machines (as in equation 3.8), it is not difficult to show that if every connected component of the graph of connections contains at least one cycle of odd length, then $\partial C^p/\partial W = 0$ everywhere implies that $V_i^{tp} = V_i^{fp}$ everywhere and therefore W must be optimal. This sufficient condition for equivalence on the structure of the graph of connections is usually easily realized (for instance, in the case of fully interconnected or of typical random graphs). So, without great loss of generality, we can assume that

$$\frac{\partial C^p}{\partial W} = 0 \text{ for every } p \Longleftrightarrow W \text{ is optimal} \tag{4.1}$$

Thus, the only critical points of C satisfying $\partial C^p/\partial W = 0$ everywhere are the points W that lead to a perfect implementation of the input/output function. We are going to show that, in general, these critical points W^* are local minima of C and therefore C is convex (not necessarily strictly convex if W^* is not isolated) in a neighborhood of any such W^*. Since $C = \sum_p C^p$, it is sufficient to show that each C^p is convex in a neighborhood of W^*. Consequently, in the rest of the discussion we shall assume now that the pattern p is fixed.

When both the free and forced systems are started at the origin with a critical configuration of weights W^*, they both converge to the same vector of activities V^{*p}. In neural networks, we can safely assume that V^{*p} is an attractor (thus V^{*p} is asymptotically stable). Let $B_W(V^{fp})$ denote the domain of attraction of V^{fp} in the free system associated with W. $B_{W^*}(V^{*p})$ is an open connected set that contains V^{*p} and the origin (see Fig. 1). 0 and V^{*fp} are in the interior of $B_{W^*}(V^{*fp})$. For sufficiently small perturbations ΔW^* of W^*, the fixed points V^{fp} and V^{tp} vary continuously as a function of $W = W^* + \Delta W^*$. Let $d(W)$ denote the smallest distance from V^{fp} to the boundary of its basin $B_W(V^{fp})$ (when these quantities are defined, which is the case in a small neighborhood of W^*). Clearly, $d(W^*) > 0$. We can find $d_0 > 0$, $\epsilon_1 > 0$, and $\epsilon_2 > 0$ such that for any

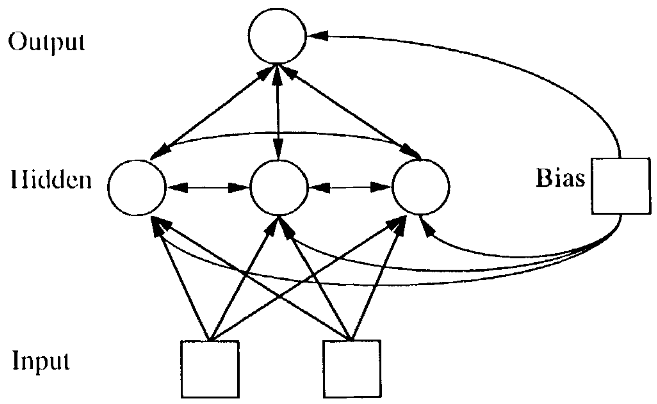

Figure 1: When the free (respectively forced) system is started at the origin 0, it converges to V^* (respectively V^*) when the internal parameters are W^*, and to V^f (respectively V^t) when the internal parameters are $W^* + \Delta W^*$. The contours are meant to represent basin boundaries. A representation of the perturbation in the space of internal parameters is shown in the inset.

perturbation ΔW^* of the parameters:

$$||\Delta W^*|| < \epsilon_1 \Longrightarrow d_0 < d(W) \tag{4.2}$$

and

$$||\Delta W^*|| < \epsilon_2 \Longrightarrow ||V^{fp} - V^{*p}|| < \frac{d_0}{2} \tag{4.3}$$

Thus, for any perturbation satisfying $||\Delta W^*|| < \epsilon_3 = \inf(\epsilon_1, \epsilon_2)$ we simultaneously have $d(W) > d_0$ and $||V^{fp} - V^{*p}|| < d_0/2$, which ensures that the open ball with radius $d_0/2$ centered at V^{*p} is entirely contained in $B_W(V^{fp})$. Now, by continuity in the forced system, we can find $\epsilon_4 > 0$ such that

$$||\Delta W^*|| < \epsilon_4 \Longrightarrow ||V^t - V^*|| < \frac{d_0}{2} \tag{4.4}$$

Finally, for any perturbation satisfying $||\Delta W^*|| < \epsilon = \inf(\epsilon_3, \epsilon_4)$ we have that $V^{tp}(W)$ is contained in the ball of radius $d_0/2$ centered at V^{*p} and therefore also in $B_{W^*+\Delta W^*}(V^{fp})$. But since the activation dynamics is a gradient dynamics, within the basin of attraction of V^{fp} we must have

$$E^{fp}(V^{fp}) \leq E^{fp}(V^{tp}) = E^{tp}(V^{tp}) \tag{4.5}$$

and therefore $C^p(W^* + \Delta W^*) \geq 0$. Hence, for any $||\Delta W^*|| < \epsilon$, $C^p(W^* + \Delta W^*) \geq C^p(W^*) = 0$. Thus the critical point W^* is a local minimum of each C^p. Each C^p is convex around W^* and the same holds for C.

It is important to observe that, in a given problem, the previous analysis does not say anything about the size of the neighborhood of W^* over which C is convex (nor do we know with certainty that such a neighborhood is always entered). This neighborhood can conceivably be small. Furthermore, nothing can be inferred at this stage on the duration of each one of the stages in the course of contrastive training. If at some point, however, the contrast function becomes negative, then one knows that the optimum has been missed and training should be stopped and restarted from a point where C is positive. Finally, the previous analysis assumes the existence of a set of weights that can do the task perfectly, without any error. Additional work is required for the case of approximate learning.

5 Oscillations and Collective Implementations

In this section, we consider some issues concerning the implementation of CL algorithms in collective physical analog systems. We cast the algorithm in the form of a set of nonautonomous coupled ordinary differential equations with fast and slow time scales, governing the simultaneous evolution of both the activation and the weight dynamics. The formal description we have given thus far of CL relies on two different dynamical systems, a free system and a forced system. For implementations, however, it is desirable to have a collective dynamical system that alternates between forced and free phases. This can be achieved in several ways by making use of some basic oscillatory mechanism to alternate between the phases and leads to algorithms that are local in space but not necessarily in time. For instance, in Example 3.2 we can introduce an oscillatory mechanism in the activation dynamics by considering the parameter γ to be a periodic function $\gamma(t)$ resulting in

$$\frac{du_i}{dt} = -\frac{u_i}{\tau_i} + \sum_j w_{ij}V_j + I_i + \gamma_i(t)(T_i - V_i)^\alpha \tag{5.1}$$

where $\gamma_i(t) = 0$ for the internal units and $\gamma_i(t) = \gamma(t)$ for the output units. If $\gamma(t)$ changes slowly enough and if the activation dynamics is fast enough, the network is almost always at steady state. For example, $\gamma(t)$ could be a square wave oscillation with magnitude γ and frequency ω [i.e., $\gamma(t) = \gamma$ or 0, depending on whether the teacher is on or off]. The network departs from steady state only during the transient after $\gamma(t)$ suddenly changes its value.

We now consider several possibilities for using oscillations in the weight updates. The most obvious approach is to perform one update per relaxation. The learning rate must be small and alternate in sign (positive when the system is forced and negative when the system is free). In this simple approach, the weight dynamics hemstiches its way downhill. It follows the true gradient of the contrast function only in the $\eta \rightarrow 0$ limit. We have found that this method is particularly prone to discontinuities associated with basin hopping phenomena. This leads to learning curves that exhibit extreme sawtooth and/or spiking behavior and often do not converge. A better approximation to equation 2.5 is to continuously integrate the weights based on the difference of some running average of $V_i V_j$ while the teacher signal is on and off. That is, something like

$$\Delta w_{ij} = \eta \frac{1}{t_1 - t_0} \int_{t_0}^{t_1} \beta(t) V_i(t) V_j(t)\, dt \tag{5.2}$$

where $\beta(t)$ can be a bipolar square wave or any other kind of bipolar oscillation and $\beta(t)$ is phase locked with $\gamma(t)$ [such that $\beta(t) < 0$ if $\gamma(t) = 0$]. In the implementation of equation 5.2, one needs to choose an integration time interval $t_1 - t_0$ of the order of several teacher signal periods and to be able to update the weights at regular multiples of this time interval. In an analog system, a suitable alternative may be to approximate equation 2.5 using the differential equations

$$\tau_w \frac{dw_{ij}}{dt} = \frac{1}{\tau_s} s_{ij} \tag{5.3}$$

with

$$\frac{ds_{ij}}{dt} = -\frac{1}{\tau_s} s_{ij} + \beta(t) V_i(t) V_j(t) \tag{5.4}$$

Equation 5.4 is a leaky integrator that averages over the forced and free gradients with opposite signs and thus calculates an approximation to the gradient of the contrast function. If $\tau \ll \tau_s \ll 2\pi/\omega$, then s_{ij} remains close at any time to its instantaneous equilibrium value $\tau_s \beta(t) V_i(t) V_j(t)$ in which case the right-hand side of equation 5.3 becomes just equal to $\beta(t) V_i(t) V_j(t)$. Thus w_{ij} integrates the instantaneous correlation between the activities V_i and V_j. Yet in practice, the filter should be rather operated with $\tau_s \gg \tau$ and $\tau_s \geq 2\pi/\omega$ in order to get good averaging over several oscillations of the teacher signal. In summary, for this scheme to work, it is essential to keep in mind that there are at least four time scales involved: the relaxation time of the activities τ, the period of the 0-1 teacher signal $2\pi/\omega$ (and the phase-locked ± 1 weight modulation), the time between pattern changes τ_p and the relaxation time of the weights τ_w. These should satisfy $\tau < 2\pi/\omega < \tau_p < \tau_w$. In principle, even with τ small, it cannot be guaranteed that the activations always converge faster than the patterns changes or faster than the oscillations. This is because there can be large variations in the convergence time of the activation dynamics as the weights evolve. In practice, it seems to be sufficient to have the activation dynamics very fast compared to other time scales so that the transients in the relaxation remain short. A similar caveat applies to Deterministic Boltzmann Machines.

It should be noticed that in CL, the waxing and waning of the clamping or of the teacher signal provide a basic underlying rhythm. In general, the time scale of the synaptic weight updates can be much larger than the period of this rhythm. However, one can envision the case where the time scale of the updates is close to the period of the rhythm. In this range of parameters, CL requires some mechanism for fast Hebbian synaptic plasticity consistent with the ideas developed by von der Malsburg (see von der Malsburg 1981).

6 Simulations and Conclusion

We have tested the CL algorithm corresponding to equations 3.8 and 3.9 on a small size but nonlinearly separable problem: XOR. The network architecture consists of three input lines (one for the bias), three hidden units and one output unit (see Fig. 2). All the units receive an input bias, but only the hidden units are connected to the inputs. In addition, the hidden units and the output are fully interconnected in a symmetric way.

We use equation 2.5 to descend along the gradient, as described in the previous section. All the gains of the units are kept constant ($\lambda = 1$) and no annealing is used.[3] The learning rate is $\eta = 0.01$. In these simulations, patterns are presented in cyclic fashion although no major differences seem to arise when presentations are done randomly. The network is relaxed using a fourth order Runge Kutta method but the results are essentially identical when Euler's integration method is used with a small step size. Figure 3 shows the results of the simulations for different values of γ. As the strength of the teacher signal γ is reduced, learning becomes slower. As can be seen, there are spikes that occur simultaneously in the contrast function and in the LMS error. These discontinuities are a consequence of basin hopping. It is easy to test this hypothesis numerically by freezing the weights just before a given spike and probing the structure of the energy surface by starting the activation dynamics at random points in a neighborhood of the origin. As expected, we observe that the system converges to one of several final states, depending sensitively on the particular initial state. To demonstrate that this algorithm actually trains the input-to-hidden weights, it is necessary to perform an additional control to eliminate the possibility that the hidden units start with an internal representation that is linearly separable by the output unit. To perform this control, we trained the network in two ways. In one case, we froze the input-to-hidden weights and in the second

[3]It is worthwhile to stress that our purpose here is only to demonstrate the basic principles and not to develop optimal algorithms for digital computers. Indeed, we can achieve significant speed-ups by using the usual tricks such as annealing, momentum terms, and higher learning rates (up to $\eta = 1$).

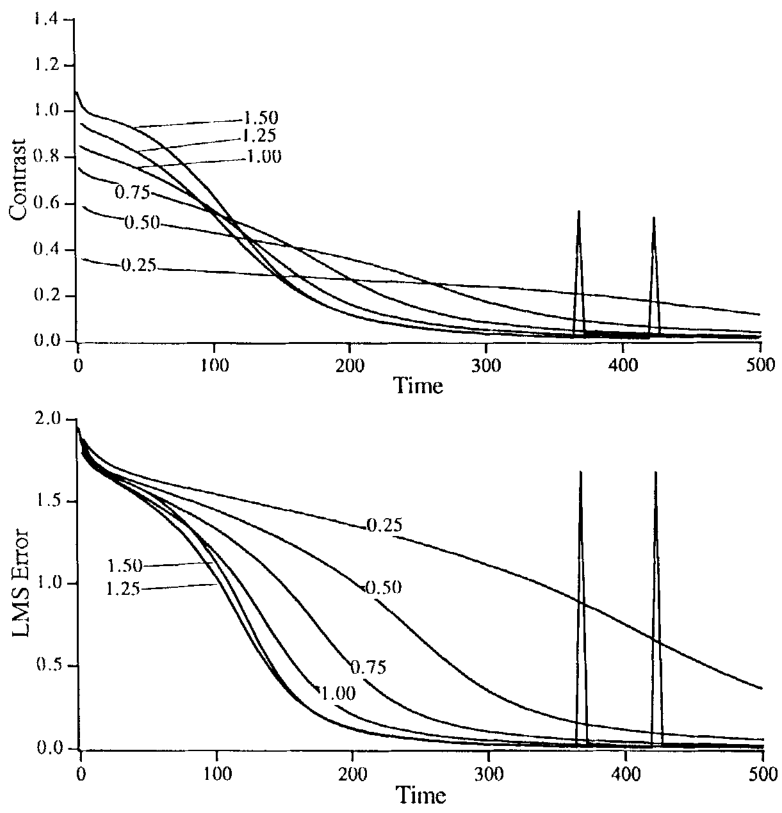

Figure 2: The network architecture used in all simulations.

case we let them adapt. We found a set of parameters where networks were untrainable with frozen input-to-hidden weights, but trainable with adaptable input-to-hidden weights. This proves that the CL algorithm is training the input-to-hidden weights. The present work needs to be extended toward problems and networks of larger size. The ratio of the number of visible units to the number of hidden units may then become an important parameter.

Finally, we would like to conclude with the conjecture that oscillations of the sort discussed in this paper may possibly play a role in biological neural systems and perhaps help in the interpretation of some of the rhythms found in nervous tissues, especially in circuits that are believed to participate in associative memory functions (for instance in hippocam-

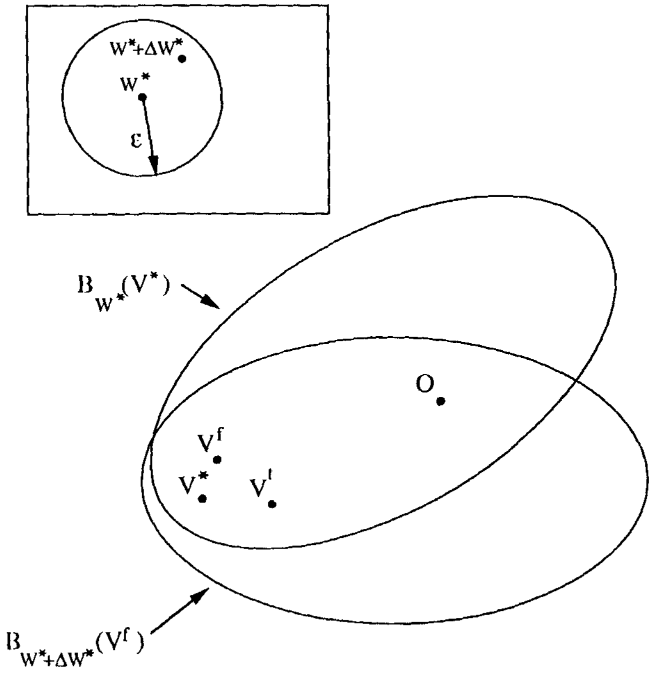

Figure 3: Time evolution of the contrast function and the LMS error function associated with equations 3.8 and 3.9 for different values of the parameter γ, with $\alpha = 1$, $\lambda = 1$, $\eta = 0.01$ and the target values T_i are ± 0.99. Relaxations are performed using fourth order Runge–Kutta method with a time step of 0.1, starting from activations initialized to 0. Initial weights are normally distributed with a mean of 0 and a variance of 0.5. Patterns are presented cyclically, once per weight update. Each iteration on the time axis corresponds to one pattern presentation, thus to the relaxation of a free and a forced network.

pus or piriform cortex). Functionally, these oscillations would correspond to some form of rehearseal whereby a waxing and waning teacher signal generated by reverberating short-term memory circuits would induce stable synaptic modifications and storage in long-term memory circuits. Physically, they would correspond to oscillations in the degree of clamping of various neuronal assemblies. These oscillations would act as local clocks in analog hardware, ensuring that communications and computations occur in the proper sequence. Intuitively, it would seem more likely for this rehearsing process to be associated with a relatively slow rhythm, perhaps in the θ range (3–10 Hz) rather than with the faster γ oscillations

(40–80 Hz) that have recently been linked to functions such as phase labeling and binding (see, for instance, Gray *et al.* 1989 and Atiya and Baldi 1989). However, a form of CL based on γ oscillations originated in sensory cortices may be able to account for the sort of memory traces revealed by experiments on priming effects (Tulving and Schacter 1990). Obviously, additional work is required to explore such hypotheses.

7 Appendix: Contrastive Learning of Trajectories

The same minimization principle that leads to simple learning algorithms for convergent dynamical systems can be extended to time-dependent recurrent networks where the visible units are to learn a time-dependent trajectory in response to a time-dependent input. In this appendix, we include for completeness two possible directions for this extension. In the first case, we assume that there are delays in the interactions and we exploit the fact that the limit trajectories are minima of suitably defined Lyapunov functions. In the second case, we assume no delays and exploit the fact that trajectories can be expressed as extrema of functionals or actions.

7.1 Time-Dependent Contrastive Learning with Delayed Interactions. In Herz *et al.* (1991), it is shown that if multiple pathways with different time delays are introduced in the usual binary Hopfield model, then cycles of period P can be stored as local minima of a time-dependent Lyapunov function that decreases as the network is relaxed. Here, we apply the contrastive learning approach to this setting with discrete time and continuously valued model neurons (Herz 1991). To be specific, consider a network of n units with activations defined by

$$V_i(t+1) = g_i[u_i(t)] \tag{a.1}$$

where g_i is a differentiable, bounded, and monotonically increasing input/output function. $u_i(t)$ is the usual local field or internal membrane potential and is defined by

$$u_i(t) = \sum_{j=1}^{n} \sum_{\tau=0}^{P-2} w_{ij}(\tau) V_j(t-\tau) + I_i(t) \tag{a.2}$$

$w_{ij}(\tau)$ is the weight of the connection from j to i with delay τ. Since the time is discretized, all the delays are integers. In addition, for simplicity, we assume that $w_{ij}(\tau) = 0$ for $\tau \geq P-1$. The task of the network is to learn to associate a periodic output cycle of length P to a periodic input cycle also of length P. The output can be read on all the units in the networks or alternatively from a subset of "visible" units. Provided the couplings satisfy an extended synaptic symmetry

$$w_{ij}(\tau) = w_{ij}[P - (2+\tau)(\mathrm{mod} P)] \tag{a.3}$$

it is not difficult to see that the network has a time-dependent energy function given by

$$
\begin{aligned}
E(V, I, W, t) = \\
& -\frac{1}{2}\sum_{i,j=1}^{n}\sum_{\tau,\alpha=0}^{P-1} w_{ij}(\tau)V_i(t-\alpha)V_j[t-(\alpha+\tau+1)(\mathrm{mod}P)] \\
& + \sum_{i=1}^{n}\sum_{\tau=0}^{P-2}\{G_i\,[V_i(t-\tau)] - I_i(t-1-\tau)V_i(t-\tau)\} \qquad \text{(a.4)}
\end{aligned}
$$

with

$$
G_i(V_i) = \int_0^{V_i} g_i^{-1}(x)\,dx \qquad \text{(a.5)}
$$

At each synchronous time step of the dynamic defined by equations a.1 and a.2, $\Delta E \le 0$ and since E, for fixed weights, is a bounded function of V, the function E must converge to a stable value. Once E has stabilized, the network is either at a fixed point, either on a cycle of period P (or, possibly, a divisor of P). Learning can be attempted by using, for instance, a generalization in time of Hebb's rule. In the CL approach, as for the case of Boltzmann Machines, one can define a forced network that relaxes with its inputs and outputs clamped to the respective target limit cycles. A time-dependent contrast function in this case is defined by

$$
C[W, I(t), T(t), t] = E^{\mathrm{t}}[V^{\mathrm{t}}, I(t), W, T(t), t] - E^{\mathrm{f}}[V^{\mathrm{f}}, I(t), W, t] \qquad \text{(a.6)}
$$

where V^{t} and V^{f} represent the instantaneous activities in the free and forced networks. By differentiating equation a.6 with respect to w, one immediately obtains the CL rule

$$
\begin{aligned}
\Delta w_{ij}(\tau) \;=\; & \eta\Big\{\sum_{\alpha=0}^{P-1} V_i^{\mathrm{t}}(t-\alpha)V_j^{\mathrm{t}}\,[(t-(\alpha+\tau+1)(\mathrm{mod}P)] \\
& - V_i^{\mathrm{f}}(t-\alpha)V_j^{\mathrm{f}}[t-(\alpha+\tau+1)(\mathrm{mod}P)]\Big\} \qquad \text{(a.7)}
\end{aligned}
$$

which is a generalized version of the usual Hebbian CL rule of Deterministic Boltzmann Machines.

7.2 Lagrange Machines. In this section, we briefly extend CL to time-dependent problems using a variational formalism. Consider a free system with an action along the trajectory Γ^{f} in the space of the activations V^{f}, during the time interval $[t_0, t_1]$, given by

$$
S^{\mathrm{f}}(\Gamma^{\mathrm{f}}) = \int_{t_0}^{t_1} L^{\mathrm{f}}[V^{\mathrm{f}}, \dot{V}^{\mathrm{f}}, W, I(t), t]\,dt \qquad \text{(a.8)}
$$

The dynamics of the system extremizes this action and corresponds to the Lagrange equations

$$\frac{d}{dt}\frac{\partial L^{\mathrm{f}}}{\partial \dot{V}^{\mathrm{f}}} = \frac{\partial L^{\mathrm{f}}}{\partial V^{\mathrm{f}}} \tag{a.9}$$

Similar relations hold in the forced system with

$$S^{\mathrm{t}}(\Gamma^{\mathrm{t}}) = \int_{t_0}^{t_1} L^{\mathrm{t}}[V^{\mathrm{t}}, \dot{V}^{\mathrm{t}}, W, I(t), T(t), t]\,dt \tag{a.10}$$

We can define the contrast function $C(W) = S^{\mathrm{t}}(\Gamma^{\mathrm{t}}) - S^{\mathrm{f}}(\Gamma^{\mathrm{f}})$ so that the internal parameters are updated according to $\Delta w = -\eta(\partial C/\partial w)$, *calculated along the two actual trajectories followed by the free and forced systems.* As in the fixed point algorithms, the differentiation of the actions contains both implicit and explicit terms. The implicit terms can be simplified by integrating by parts and using the Lagrange equations. Finally, we get

$$\Delta w = -\eta \sum_i \left[\frac{\partial L^{\mathrm{t}}}{\partial \dot{V}_i^{\mathrm{t}}}\frac{\partial V_i^{\mathrm{t}}}{\partial w} - \frac{\partial L^{\mathrm{f}}}{\partial \dot{V}_i^{\mathrm{f}}}\frac{\partial V_i^{\mathrm{f}}}{\partial w}\right]_{t_0}^{t_1} - \eta \int_{t_0}^{t_1} \frac{\partial L^{\mathrm{t}}}{\partial w} - \frac{\partial L^{\mathrm{f}}}{\partial w}\,dt \tag{a.11}$$

Now there are several possible choices for L^{f} and L^{t} and two observations can be made. First, if the dependence of L^{f} (and L^{t}) on w is based on a quadratic form such as $L^{\mathrm{f}} = -\frac{1}{2}\sum_{i,j} w_{ij} V_i^{\mathrm{f}} V_j^{\mathrm{f}} + V_i^{\mathrm{f}} I_i(t) + \sum_i f_i(V_i^{\mathrm{f}}, \dot{V}_i^{\mathrm{f}})$, then the integral in equation a.11 is Hebbian, similar to the Deterministic Boltzmann Machine learning rule, and equal to the average of the difference between the local covariance of activities. Second, by properly choosing the initial conditions in the free and forced systems, it is easy to have the corresponding boundary term in equation a.11 vanish. Finally, it should be noticed that with a nondissipative dynamics, the trajectories that are learned can vary with the input but cannot be stored as attractors.

Acknowledgments

We would like to thank the referee for several useful comments. This work is supported by ONR Contract NAS7-100/918 and a McDonnell-Pew grant to P. B., and AFOSR Grant ISSA-91-0017 to F. P.

References

Ackley, D. H., Hinton, G. E., and Sejnowski, T. J. 1985. A learning algorithm for Boltzmann Machines. *Cog. Sci.* **9**, 147–169.

Atiya, A., and Baldi, P. 1989. Oscillations and synchronizations in neural networks: An exploration of the labeling hypothesis. *Int. J. Neural Syst.* **1**(2), 103–124.

Baldi, P., and Meir, R. 1990. Computing with arrays of coupled oscillators: An application to preattentive texture discrimination. *Neural Comp.* **2**(4), 458–471.

Cohen, M. A., and Grossberg, S. 1983. Absolute stability of global pattern formation and parallel memory storage by competitive neural networks. *IEEE Transact. Syst. Man Cyber.* SMC **13**(5), 815–826.

Gray, C. M., König, P., Engel, A. K., and Singer, W. 1989. Oscillatory responses in cat visual cortex exhibit inter-columnar synchronization which reflects global stimulus properties. *Nature (London)* **338**, 334–337.

Herz, A. V. M. 1991. Global analysis of parallel analog networks with retarded feedback. *Phys. Rev. A* **44**(2), 1415–1418.

Herz, A. V. M., Li, Z., and van Hemmen, J. L. 1991. Statistical mechanics of temporal association in neural networks with transmission delays. *Phys. Rev. Lett.* **66**(10), 1370–1373.

Hinton, J. 1989. Deterministic Boltzmann learning performs steepest descent in weight-space. *Neural Comp.* **1**, 143–150.

Hopfield, J. J. 1984. Neurons with graded responses have collective computational properties like those of two-state neurons. *Proc. Natl. Acad. Sci. U.S.A.* **81**, 3088–3092.

Movellan, J. 1990. Contrastive Hebbian learning in the continuous Hopfield model. *Proceedings of the 1990 Carnegie Mellon Summer School.* Morgan Kaufmann, San Mateo, CA.

Peterson, C., and Anderson, J. R. 1987. A mean field theory learning algorithm for neural networks. *Complex Syst.* **1**, 995–1019.

Pineda, F. 1988. Dynamics and architecture for neural computation. *J. Complex.* **4**, 216–245.

Tulving, E., and Schacter, D. L. 1990. Priming and human memory systems. *Science* **247**, 301–306.

von der Malsburg, C. 1981. The correlation theory of brain function. Internal Report 81-2, Department of Neurobiology, Max Planck Institute for Biophysical Chemistry.

Zak, M. 1989. Terminal attractors in neural networks. *Neural Networks* **2**, 259–274.

Received 6 February 1991; accepted 14 June 1991.