Algorithmica (1991) 6: 466-478

Algorithmica

Stochastic Neural Networks[1]

Eugene Wong[2]

Abstract. The first purpose of this paper is to present a class of algorithms for finding the global minimum of a continuous-variable function defined on a hypercube. These algorithms, based on both diffusion processes and simulated annealing, are implementable as analog integrated circuits. Such circuits can be viewed as generalizations of neural networks of the Hopfield type, and are called "diffusion machines."

Our second objective is to show that "learning" in these networks can be achieved by a set of three interconnected diffusion machines: one that learns, one to model the desired behavior, and one to compute the weight changes.

Key Words. Neural network, Simulated annealing, Diffusion.

1. Hopfield Networks. It is well known [1], [2] that a neural network can be used to compute a local minimum of a function $E(x)$ defined on a hypercube $[0, 1]^n$ as follows. Let $v_i(t)$ be the state at node i at time t and set

$$v_i(t) = g(u_i(t)), \tag{1.1}$$

$$\frac{du_i(t)}{dt} = -E_i(v(t)), \tag{1.2}$$

where $E_i(v) = (\partial/\partial v_i)\, E(v)$ and g is an increasing function. The special case of a quadratic function

$$E(v) = -\tfrac{1}{2} \sum_{i,\, j} w_{ij} v_i v_j - \sum_i \theta_i v_i, \tag{1.3}$$

where we assume $w_{ij} = w_{ji}$, results in

$$E_i(v) = -\sum_j w_{ij} v_j - \theta_i, \tag{1.4}$$

which is particularly well suited for realization as an analog integrated circuit.

As Hopfield and Tank [3] and others (e.g., [4]) have shown, a variety of computational problems of considerable complexity can be reduced to computing the global minimum of a quadratic function. With current technology, a network with several hundred nodes and with the dynamics given by (1.1)-(1.3) can

[1] This research was supported in part by U.S. Army Research Office Grant DAAL03-89-K-0128.
[2] Department of Electrical Engineering and Computer Science, University of California at Berkeley, CA 94720, USA.

Received May, 1989; revised October, 1989. Communicated by Alberto Sangiovanni-Vincentelli.

probably be built on a single chip. The potential for simple and fast computation thus created is exciting indeed.

However, (1.1) and (1.2) represent essentially a gradient-descent method for minimization, and such methods do not usually reach a global minimum. To see this we write, using (1.1) and (1.2),

$$\frac{d}{dt}E(v(t)) = \sum_i E_i(v(t))\frac{dv_i(t)}{dt} \tag{1.5}$$

$$= \sum_i E_i(v(t))g'(u_i(t))\frac{du_i(t)}{dt}$$

$$= -\sum_i g'(u_i(t))E_i^2(v(t)) \leq 0,$$

which shows that E is decreasing in t but not *strictly* decreasing and any equilibrium reached may only be a local minimum.

2. Boltzmann Machines. For some problems it is sufficient to restrict v_i to binary values, say $v_i = 0, 1$. The collection of the states at all the nodes of the network $v(t) = \{v_i(t)\}$ now takes values in $\{0, 1\}^n$, and $v(t)$ is called the *configuration* of the network at time t. A Boltzmann machine [5] is a network where $v(t)$ is a $\{0, 1\}^n$ valued discrete-time Markov chain with state transitions defined as follows:

For each $v \in \{0, 1\}^n$ define a neighborhood $N(v)$ as a subset of $\{0, 1\}^n$. We assume that $v' \in N(v) \Rightarrow v \in N(v')$ and $v \notin N(v)$. At time $t + 1$ we choose a $v' \in N(v(t))$ at random (say with equal probabilities) and set

$$v(t+1) = \begin{cases} v' & \text{with probability } p(\Delta E), \\ v(t) & \text{with probability } 1 - p(\Delta E), \end{cases} \tag{2.1}$$

where ΔE is given by

$$\Delta E = E(v') - E(v(t)) \tag{2.2}$$

and the *acceptance probability* $p(\Delta E)$ is of the form

$$p(\Delta E) = e^{-(1/2T)\Delta E} f(|\Delta E|) \tag{2.3}$$

for some decreasing function f. Familiar examples include

$$p(\Delta E) = \min(1, e^{-(1/T)\Delta E})$$

$$= e^{-(1/2T)\Delta E} e^{-(1/2T)|\Delta E|}$$

and

$$p(\Delta E) = \frac{1}{1 + e^{(1/T)\Delta E}} = \frac{e^{-(1/2T)\Delta E}}{2\cosh(\Delta E/2T)}.$$

It is clear that $v(\cdot)$ has a bias for moving in the direction of negative ΔE, but will move with nonzero probabilities even for positive ΔE.

The one-step transition probability is given by

$$\begin{aligned} P(v|v_0) &= \text{Prob}(v(t+1) = v | v(t) = v_0) \\ &= \begin{cases} \dfrac{1}{|N(v_0)|} p(E(v) - E(v_0)), & v \in N(v_0), \\ 0, & v \neq v_0, \quad v \notin N(v_0), \end{cases} \end{aligned} \tag{2.4}$$

and

$$1 - \frac{1}{|N(v_0)|} \sum_{v' \in N(v_0)} p(E(v') - E(v_0)), \qquad v = v_0,$$

where $|N(v_0)|$ denotes the cardinality of $N(v_0)$. With p given by (2.3), we have

$$\begin{aligned} &\sum_{v_0} P(v|v_0) e^{-(1/T)E(v_0)} |N(v_0)| \\ &\quad = P(v|v) e^{-(1/T)E(v)} |N(v)| + \sum_{v_0 \in N(v)} P(v|v_0) e^{-(1/T)E(v_0)} |N(v_0)| \\ &\quad = |N(v)| e^{-(1/T)E(v)} - \sum_{v' \in N(v)} f(|E(v') - E(v)|) e^{-(1/2T)[E(v) + E(v')]} \\ &\qquad + \sum_{v_0 \in N(v)} f(|E(v) - E(v_0)|) e^{-(1/2T)[E(v) + E(v_0)]} \\ &\quad = |N(v)| e^{-(1/T)E(v)}. \end{aligned}$$

It follows that a probability distribution of the form

$$P(v(t) = v) = K |N(v)| e^{-(1/T)E(v)}$$

is left invariant by the transitions.

If, in addition, the Markov chain is irreducible, i.e., every v can be reached from any initial configuration v_0, then

$$P(v(t) = v | v(t_0) = v_0) \underset{(t - t_0) \to \infty}{\to} K |N(v)| e^{-(1/T)E(v)}, \tag{2.5}$$

where K is the normalizing constant. If we assume $|N(v)|$ is independent of v, then

the stationary distribution is simply

$$P(v) = \frac{1}{Z} e^{-(1/T)E(v)}, \tag{2.6}$$

where $Z = \sum_v e^{-(1/T)E(v)}$ is called the partition function in statistical mechanics.

Equation (2.6) is called the Gibbs or Boltzmann distribution, and the Markov chain $v(t)$ is called a *Gibbs field.* A network with such a $v(t)$ has been called a *Boltzmann machine* [5].

3. Simulated Annealing. Because the stationary distribution of a Boltzmann machine is given by

$$P(v) = \frac{1}{Z} e^{-(1/T)E(v)}, \tag{3.1}$$

the peaks of $P(v)$ coincide with the minima of $E(v)$. As the parameter T (temperature) decreases to zero, $P(v)$ will approach a set of singularities at the global minima of $E(v)$. This is the principle on which *simulated annealing* is based [6].

Suppose that we choose a sequence $\{T_k\}$ decreasing to 0 sufficiently slowly so that, for large k, $v(k)$ is distributed approximately according to

$$P_k(v) = \frac{1}{Z_k} e^{-(1/T_k)E(v)}.$$

Then we would expect $v(k)$ to converge to a global minimum. This is indeed the case for T_k of the form

$$T_k = \frac{c}{\ln(1+k)}, \tag{3.2}$$

where c is a "sufficiently large" constant [7], [8].

Simulated annealing can be extended to the continuous variable case. This is done with the Langevin algorithm [9], [10], which is defined by a set of stochastic differential equations of the form

$$dv_i(t) = -E_i(v(t))\,dt + \sqrt{2T}\,dW_i(t), \tag{3.3}$$

where $E_i = (\partial/\partial v_i)\,E$ as before and $\{W_i\}$ is a set of independent Wiener processes. The goal, once again, is to get a stationary distribution for $v(t)$ characterized by a density function of the form

$$P_0(v) = \frac{1}{Z} e^{-(1/T)E(v)}. \tag{3.4}$$

However, for $v \in \mathbb{R}^n$ there may be no density function of this form since $e^{(-(1/T)E)}$ may not be integrable. For $v \in [0, 1]^n$, (3.3) also does not guarantee a stationary density of the form (3.4). Intuitively, we can interpret $v(t)$ as the position of a particle undergoing random motion according to (3.3). To ensure a stationary density of the form (3.4), we have to prevent the particle from escaping the hypercube $[0, 1]^n$. This requires a set of boundary conditions known as the "reflecting boundary" conditions at every boundary $v_i = 0, 1$ [9].

4. Diffusion Machines: Stochastic Hopfield Networks. We propose a scheme that is a modification of both the Langevin algorithm and the Hopfield network. Suppose that we inject noise in a Hopfield network so that at the ith node the equations of dynamics are now given by (see (1.1) and (1.2))

$$v_i(t) = g(u_i(t)), \tag{4.1}$$

$$du_i(t) = -E_i(v(t))\,dt + \alpha_i(u(t))\,dW_i(t), \tag{4.2}$$

where (4.2) is a stochastic differential equation of the Ito type [11]. As in the Langevin algorithm, $\{W_i\}$ are independent Wiener processes.

The question we now pose is the following: Can α_i's be found so that $v(t)$ is a stationary Markov process with the following stationary density?

$$p_0(v) = \frac{1}{Z} e^{-(1/T)E(v)}. \tag{4.3}$$

The answer is surprisingly simple. The required α_i is given by

$$\alpha_i(u(t)) = \sqrt{\frac{2T}{g'(u_i(t))}} \tag{4.4}$$

so that (4.2) becomes

$$du_i(t) = -E_i(v(t))\,dt + \sqrt{\frac{2T}{g'(u_i(t))}}\,dW_i(t). \tag{4.5}$$

Furthermore, if we denote

$$f(x) = g'(g^{-1}(x)), \tag{4.6}$$

then $v_i(t)$ satisfies the stochastic differential equation

$$dv_i(t) = -f(v_i(t))E_i(v(t))\,dt + Tf'(v_i(t))\,dt + \sqrt{2Tf(v_i(t))}\,dW_i(t). \tag{4.7}$$

As is explained in Section 6, if $f(x) = 0$ at $x = 0, 1$, then stationarity of $v(t)$ is assured. If not, a reflecting boundary is needed at each $v_i = 0, 1$. Observe that f

depends only on the nonlinearity g. Hence, stationarity of the process $v(t)$ can be ensured by a proper choice of g. In this sense, we can understand the role of g to be one of stabilizing the network.

To derive (4.4), we first note that with a smooth g we can use the Ito differentiation formula [12] and derive a set of stochastic differential equations for v_i which are of the form

$$dv_i(t) = m_i(v(t))\,dt + \sigma_i(v(t))\,dW_i(t). \tag{4.8}$$

The transition density $p(v, t \mid v_0, t_0)$ of $v(t)$ must satisfy the Fokker–Planck equation

$$\frac{\partial p}{\partial t} = \sum_i \frac{\partial}{\partial v_i}\left[\frac{1}{2}\frac{\partial}{\partial v_i}(\sigma_i^2 p) - m_i p\right]. \tag{4.9}$$

It follows that if (4.3) is to be the stationary density, then we must have

$$\sum_i \frac{\partial}{\partial v_i}\left[\frac{1}{2}\frac{\partial}{\partial v_i}(\sigma_i^2 p_0) - m_i p_0\right] = 0 \tag{4.10}$$

which is satisfied if m_i and σ_i^2 satisfy

$$m_i = \tfrac{1}{2}\sigma_i^2\left(-\frac{1}{T}E_i(v)\right) + \frac{\partial}{\partial v_i}(\tfrac{1}{2}\sigma_i^2). \tag{4.11}$$

Since (4.8) is derived from (4.1) and (4.2) using the Ito differentiation formula, we have

$$dv_i(t) = g'(u_i(t))\,du_i(t) + \tfrac{1}{2}g''(u_i(t))\alpha_i^2(u(t))\,dt. \tag{4.12}$$

Comparing (4.12) and (4.8), we get

$$\sigma_i(g(u)) = g'(u_i)\alpha_i(u) \tag{4.13}$$

and

$$m_i(v) = -f(v_i)E_i(v) + \tfrac{1}{2}(\ln f)'(v_i)\sigma_i^2(v), \tag{4.14}$$

where f is given by

$$f(x) = g'(g^{-1}(x)). \tag{4.6}$$

□

Comparing (4.14) with (4.11) now yields

$$\tfrac{1}{2}\sigma_i^2(v) = Tf(v_i) \tag{4.15}$$

which is a relationship of great simplicity.

The simplicity of the coefficient of the noise term as given by (4.4) or (4.15) cannot be overemphasized. It is both fortuitous and surprising. The required form for $\alpha_i(u)$ depends only on u_i. Thus, the noise term at each node is *local.* Furthermore, the noise term in no way depends on $E(v)$ or the weights. Thus, any weight adjustment procedure would not affect the nodes except through their input.

A network with dynamics governed by (4.1) and (4.5) (equivalently (4.7)) is called a *diffusion machine.* We propose that it be used as the basis for studying simulated annealing and machine learning [5]. As a neural computing system, it has a number of important advantages. First, it is quite general. There is no need to assume that the minimum occurs at a corner of the cube. Second, it allows the nonlinearity g to play a stabilizing role in ensuring stationarity. Finally, and most importantly, it is well suited for direct circuit implementation, thus providing potentially much faster computation for both annealing and machine learning.

5. An Example. A favorite choice of g is

$$g(x) = \tfrac{1}{2}\left(1 + \tanh\frac{x}{a}\right). \tag{5.1}$$

Its popularity is due to the fact that the tanh function is easy to realize in CMOS circuits operating in subthreshold mode. This choice for g yields

$$g'(x) = \frac{1}{2a}\left(1 - \tanh^2\frac{x}{a}\right) = \frac{1}{2a}\left(\cosh\frac{x}{a}\right)^{-2} \tag{5.2}$$

and

$$\begin{aligned} f(y) = g'(g^{-1}(y)) &= \frac{1}{2a}[1 - (2y-1)^2] \\ &= \frac{2}{a}y(1-y). \end{aligned} \tag{5.3}$$

Equations (4.5) and (4.7) now become

$$du_i(t) = -E_i(v(t))\,dt + 2\sqrt{aT}\cosh\left(\frac{u_i(t)}{a}\right)dW_i(t), \tag{5.4}$$

$$\begin{aligned} dv_i(t) = &-\frac{2}{a}v_i(t)[1 - v_i(t)]E_i(v(t))\,dt + \frac{2T}{a}[1 - 2v_i(t)]\,dt \\ &+ 2\sqrt{\frac{T}{a}v_i(t)[1 - v_i(t)]}\,dW_i(t). \end{aligned} \tag{5.5}$$

As a second example, consider the case

$$g(x) = \begin{cases} x, & 0 \le x \le 1, \\ 1, & x > 1, \\ 0, & x < 0. \end{cases}$$

This example corresponds to the Langevin algorithm as considered in [9]. Because $g^{-1}(v)$ does not exist in this case, this example is not really a diffusion machine. If it is to be considered at all, reflecting boundaries at $v_i = 0, 1$ are required [9].

6. Rate of Convergence. A diffusion machine can be used with a cooling schedule $\{T_k\}$ to achieve simulated annealing. For that purpose an estimate of the rate at which

$$p(v, t | v_0, t_0) \to p_0(v)$$

is needed. Diffusion theory provides a powerful approach to such estimates (see [10]).

Because the coefficients σ_i and m_i in the Fokker–Planck equation (4.9) do not depend on t, a separation-of-variables argument shows that $p(v, t | v_0, 0)$ can be expressed in the form [12]

$$p(v, t | v_0, 0) = p_0(v) \sum_{\lambda} e^{-\lambda t} \psi_\lambda(v) \psi_\lambda(v_0), \tag{6.1}$$

where λ are the eigenvalues and ψ_λ are the normalized eigenfunctions of the equation

$$T \sum_i \frac{\partial}{\partial v_i} \left[p_0(v) f(v_i) \frac{\partial \psi_\lambda(v)}{\partial v_i} \right] + \lambda p_0(v) \psi_\lambda(v) = 0. \tag{6.2}$$

If $f(v_i)$ is zero at $v_i = 0, 1$, then, by multiplying each term in (6.2) by ψ_λ and integrating, we get

$$\lambda \int_{[0,1]^n} p_0(v) \psi_\lambda^2(v)\, dv = T \sum_i \int_{[0,1]^n} p_0(v) f(v_i) \left[\frac{\partial \psi_\lambda(v)}{\partial v_i} \right]^2 dv.$$

It is clear that $\lambda = 0$ is the smallest eigenvalue with $\psi_0(v) = 1$, and the eigenvalues can be ordered

$$0 = \lambda_0 < \lambda_1 \le \lambda_2 \cdots.$$

The corresponding eigenfunctions ψ_k are orthogonal for different k and can be

normalized. These results show that $f(x) = 0, x = 0, 1$ is a sufficient condition for stationarity.

Since ψ_1 is normalized and orthogonal to $\psi_0 = 1$, we get

$$\lambda_1 = \min_{\psi} T \int_{[0,1]^n} p_0(v) \sum_i f(v_i) \left(\frac{\partial \psi(v)}{\partial v_i} \right)^2 dv \tag{6.3}$$

subject to the conditions

$$\int_{[0,1]^n} p_0(v) \psi^2(v)\, dv = 1 \tag{6.4}$$

and

$$\int_{[0,1]^n} p_0(v) \psi(v)\, dv = 0. \tag{6.5}$$

It is clear that $\lambda_1 > 0$, and its dependence on T and g can be studied via (6.3).

From (6.1) we get

$$\begin{aligned} |p(v, t|v_0, 0) - p_0(v)| &= |\sum_{k=1}^{\infty} e^{-\lambda_k t} \psi_k(v) \psi_k(v_0)| \\ &= e^{-\lambda_1 t} |\sum_{k=1}^{\infty} e^{-(\lambda_k - \lambda_1)t} \psi_k(v) \psi_k(v_0)|. \end{aligned}$$

If we denote

$$k(v) = e^{\lambda_1} |p(v, 1|v, 0) - p_0(v)|,$$

then for $t > 1$

$$\begin{aligned} |p(v, t|v_0, 0) - p_0(v)| &\leq e^{-\lambda_1 t} \sqrt{k(v) k(v_0)} \\ &\leq e^{-\lambda_1 t} \sup_v k(v), \end{aligned} \tag{6.6}$$

which gives an estimate of the rate of convergence of the transition density to the equilibrium distribution.

7. Analog Realization. Equations (4.5) and (4.7) are Ito equations. To realize them in analog circuits using Gaussian wideband noise requires a correction term [11]. The origin of the correction term is rather technical. It has to do with the fact that $dW(t)$, the differential increment of a Wiener process, is proportional to $\sqrt{dt}$,

rather than dt, and that stochastic differential equations are based on a forward-difference approximation. It turns out that if $\dot{W}$ is the derivative of a Wiener process, then $F(W(t))\dot{W}(t)\,dt$ is like $F(W(t + \frac{1}{2}dt))\,dW(t)$. The correction term can then be computed using Taylor series and the estimate $|dW(t)| = \sqrt{dt}$. With the correction term, (4.5) and (4.7) can be rewritten as

$$\dot{u}_i(t) = -E_i(v(t)) + \frac{T}{2}\frac{g''(u_i(t))}{[g'(u_i(t))]^2} + \sqrt{\frac{2T}{g'(u_i(t))}}\,\eta_i(t), \tag{7.1}$$

$$\dot{v}_i(t) = -f(v_i(t))E_i(v(t)) + \frac{T}{2}f'(v_i(t)) + \sqrt{2Tf(v_i(t))}\eta_i(t), \tag{7.2}$$

where $\eta_i = \dot{W}_i$ is a Gaussian white noise.

For the example given in Section 5, we can write for (7.1)

$$\dot{u}_i(t) = -E_i(v(t)) - 2T\sinh\frac{2u_i(t)}{a} + 2\sqrt{aT}\cosh\frac{n_i(t)}{a}\,\eta_i(t). \tag{7.3}$$

A block diagram realization of (7.3) is given in Figure 1.

8. A Continuous Operating Learning System. Hinton *et al.* [5] defined a learning problem for Boltzmann machines that can be extended to diffusion machines. Suppose that the n nodes in the network are divided into two groups: *visible* nodes and *hidden* nodes. The space $[0, 1]^n$ is correspondingly factored into the cartesian product $V \times H$, where $V = \{v_i\colon i \text{ visible}\}$ and $H = \{v_i\colon i \text{ hidden}\}$. Now, suppose that a probability density function $\tilde{p}$ is specified in V and we want to find a set of weights w_{ij} so that the stationary distribution of $v_i(t)$ on the visible nodes will be as close to $\tilde{p}$ as possible.

We now modify our earlier notation to make the dependencies more explicit. Let $p_0(v, h; w)$ denote the stationary density of the entire network for $v \in V$, $h \in H$, and

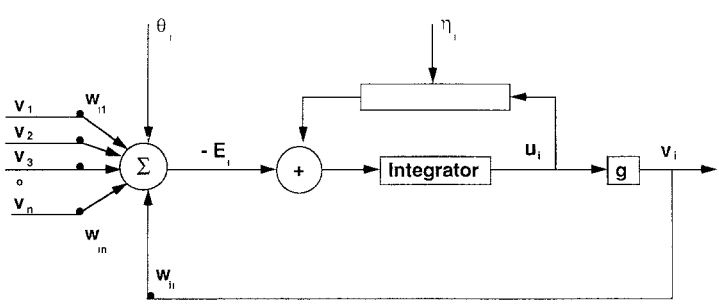

Fig. 1. A node in a diffusion machine.

let w denote the weights. Let the density on V alone be denoted by

$$p_0(v: w) = \int_H p_0(v, h; w)\, dh. \tag{8.1}$$

The problem now is to find w so that $p_0(v; w)$ approximates $\tilde{p}(v)$.

Following [5], we use the asymmetric divergence

$$G(w) = \int_V \tilde{p}(v) \ln\left[\frac{\tilde{p}(v)}{p_0(v; w)}\right] dv. \tag{8.2}$$

as a measure of approximation, and choose w to minimize $G(w)$. Now,

$$p_0(v, h; w) = \frac{1}{Z(w)} e^{-(1/T)E(v, h; w)}$$

and

$$Z(w) = \int_{V \times H} e^{-(1/T)E(v, h; w)}\, dv\, dh. \tag{8.3}$$

Hence,

$$p_0(v; w) = \frac{\int_H e^{-(1/T)E(v, h; w)}\, dh}{\int_{V \times H} e^{-(1/T)E(v, h; w)}\, dv\, dh} \tag{8.4}$$

and

$$\frac{\partial G(w)}{\partial w_{ij}} = \frac{1}{T} \int_{V \times H} \frac{\partial E(v, h; w)}{\partial w_{ij}} [p_0(v, h; w) - \tilde{p}(v) p_0(h|v; w)]\, dv\, dh, \tag{8.5}$$

where

$$p_0(h|v; w) = \frac{p_0(v, h; w)}{p_0(v; w)}. \tag{8.6}$$

If we denote by $\mathscr{E}_0$ the expectation with respect to $p_0(v, h; w)$ and by $\tilde{\mathscr{E}}$ the expectation with respect to $\tilde{p}(v) p_0(h|v; w)$, then we can write

$$\frac{\partial G(w)}{\partial w_{ij}} = \frac{1}{T}\left[\mathscr{E}_0\left(\frac{\partial E}{\partial w_{ij}}\right) - \tilde{\mathscr{E}}\left(\frac{\partial E}{\partial w_{ij}}\right)\right]. \tag{8.7}$$

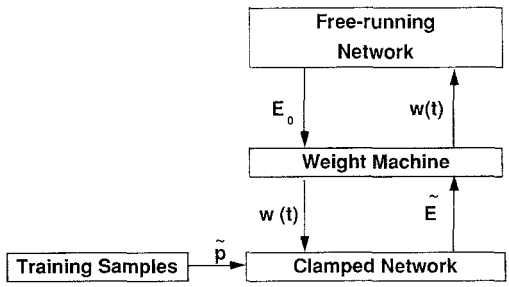

Fig. 2. A continuously operating learning system.

We first observe that if E is the quadratic function given by (1.3), then

$$\frac{\partial E}{\partial w_{ij}} = \begin{cases} -v_i v_j & \text{for } i \neq j, \\ -\frac{1}{2} v_i^2 & \text{for } i = j. \end{cases}$$

Hence, we can estimate $\partial G/\partial w_{ij}$ by running the network in two modes: (a) *a free-running* mode that yields $\mathscr{E}_0$, and (b) a *clamped* mode where for the visible nodes $\{v_i\}$ are found to have a distribution given by $\tilde{p}$ to yield $\tilde{\mathscr{E}}$.

Next, we observe that we can use the Langevin algorithm to minimize $G(w)$. Specifically, we can set

$$dw_{ij}(t) = -G_{ij}(w(t))\, dt + \sqrt{2S}\, dZ_{ij}(t),$$

where $G_{ij} = \partial G/\partial w_{ij}$ comes from the two copies of the network running in two different modes, Z_{ij} are independent Wiener processes, and S is the temperature for the "weight machine" and is different from the temperature T of the networks used to generate G_{ij}. A block diagram of the "learning dynamics" is given in Figure 2.

Intuitively, if we change $w(t)$ slowly in comparison to the dynamics of the two networks, then we should expect the two networks to reach approximate equilibrium before the weights are changed significantly. The continously operating nature of a diffusion machine in the learning mode makes it a most attractive system.

9. Conclusion. In this paper we consider a class of stochastic networks, which we call *diffusion machines*, that result from a modification of continuous-variable Hopfield networks. We show that by injecting white Gaussian noise in a specific way at each node of a Hopfield network, we obtain a diffusion process with a stationary density of the Boltzmann form. It follows that cooling of the noise sources can be used to achieve annealing, which in turn can be used to obtain a global minimum. As such it is closely related to both the Boltzmann machine and the Langevin algorithm, but superior to both in terms of possible integrated circuit realization [13].

Learning algorithms similar to those proposed for Boltzmann machines are a particularly interesting problem for study. We propose an arrangement consisting of three coupled diffusion machines that perform the functions of training, learning, and weight-adjustment concurrently in continuous operation. This is in contrast to learning in Boltzmann machines which alternates between a learning phase and an "equilibriating" phase. A substantial speed advantage for diffusion machines is likely.

References

[1] J. J. Hopfield, "Neurons with graded response have collective computational properties like those of two-state neurons, *Proc. Nat. Acad. Sci. USA*, **81** (1984), 3088–3092.

[2] L. O. Chua and G. N. Lin, Nonlinear programming without computation, *IEEE Trans. Circuits and Systems*, **31** (1984), 182–188.

[3] J. J. Hopfield and D. W. Tank, Neural computation of decisions optimization problems, *Biol. Cybernet*, **52** (1985), 141–152.

[4] M. Takeda and J. W. Goodman, Neural networks for computation: number representations and programming complexity, *Appl. Optics*, **25** (1986), 3033–3046.

[5] D. H. Ackley, G. W. Hinton and T. J. Sejnowski, A learning algorithm for Boltzmann machines, *Cognitive Sci.* **9** (1985), 147–169.

[6] S. Kirkpatrick, C. D. Gelatt, Jr., and M. P. Vecchi, Optimization by simulated annealing, *Science*, **220** (1983), 671–680.

[7] D. Mitra, F. Romeo, and A. Sangiovanni-Vincentelli, Convergence and finite-time behavior of simulated annealing, *Adv. in Appl. Probab.*, **18** (1986), 747–771.

[8] B. Hajek, Cooling schedules for optimal annealing, *Math. Oper. Res.*, **13** (1988), 311–319.

[9] S. Geman and C. R. Hwang, Diffusions for global optimization, *SIAM J. Control Optim.*, **24** (1986), 1031–1043.

[10] B. Guidas, Global optimization via the Langevin equation, *Proc. 24th IEEE Conference on Decision and Control*, 1985, pp. 774–778.

[11] E. Wong and M. Zakai, On the convergence of ordinary integrals to stochastic integrals, *Ann. Math Statist.*, **36** (1965), 1560–1564.

[12] E. Wong and B. Hajek, *Stochastic Processes in Engineering Systems*, Springer-Verlag, New York, 1984.

[13] J. Alspector and R. B. Allen, A neuromorphic VLSI learning system, in *Advanced Research in VLSI, Proc. 1987 Stanford Conference*, P. Losleben, ed., MIT Press, Cambridge, MA, 1987, pp. 313–349.