Lending Direction to Neural Networks

Richard S. Zemel[1]

Christopher K. I. Williams[1]

Michael C. Mozer[2]

Abstract

We present a general formulation for a network of stochastic directional units. This formulation is an extension of the Boltzmann machine in which the units are not binary, but take on values on a cyclic range, between 0 and 2π radians. This measure is appropriate to many domains, representing cyclic or angular values, e.g., wind direction, days of the week, phases of the moon. The state of each unit in a Directional-Unit Boltzmann Machine (DUBM) is described by a complex variable, where the phase component specifies a direction; the weights are also complex variables. We associate a quadratic energy function, and corresponding probability, with each DUBM configuration. The conditional distribution of a unit's stochastic state is a circular version of the Gaussian probability distribution, known as the von Mises distribution. In a mean-field approximation to a stochastic DUBM, the phase component of a unit's state represents its mean direction, and the magnitude component specifies the degree of certainty associated with this direction. This combination of a value and a certainty provides additional representational power in a unit. We present a proof that the settling dynamics for a mean-field DUBM cause convergence to a free energy minimum. Finally, we describe a learning algorithm and simulations that demonstrate a mean-field DUBM's ability to learn interesting mappings.

1 Introduction

Many kinds of information can naturally be represented in terms of angular, or directional, variables. A circular range forms a suitable representation for explicitly directional information, such as wind direction, as well as for information where the underlying range is periodic, such as days of the week or months of the year. In computer vision, tangent fields and optic

[1]Department of Computer Science, University of Toronto, Toronto, ONT, M5S 1A4 CANADA

[2]Department of Computer Science & Institute of Cognitive Science, University of Colorado, Boulder, CO, 80309–0430

flow fields are represented as fields of oriented line segments, each of which can be described by a magnitude and direction. Directions can also be used to represent a set of symbolic labels, e.g., object label A at 0, and object label B at $\pi/2$ radians. We discuss below some advantages of representing symbolic labels with directional units.

These and many other phenomena can be usefully encoded using a *directional* representation—a polar coordinate representation of complex values in which the phase parameter indicates a direction between 0 and 2π radians. We have devised a general formulation of networks of stochastic directional units. This formulation is a novel generalization of a Boltzmann machine (Ackley, Hinton and Sejnowski, 1985) in which the units are not binary, but instead take on directional values between 0 and 2π radians. This paper presents the theoretical basis of a *directional-unit Boltzmann machine* (DUBM), including an energy function and underlying probabilistic model. We also present a simple derivation of a deterministic version of a DUBM and a convergence proof for its dynamics. Finally, we describe a generalization of the Boltzmann machine learning algorithm, and demonstrate that a DUBM is able to learn interesting mappings.

2 Stochastic Directional Unit Boltzmann Machines

2.1 The DUBM Energy Function

We formulate a network composed of directional units—a DUBM—in an analogous manner to a Boltzmann machine with binary-state units. In the Hopfield spin-system (Hopfield, 1982) and in a Boltzmann machine, the units are binary, and the energy is defined to be the sum of the pairwise products of the units' states and their connecting weights. This energy can be written in quadratic form: $E(\mathbf{s}) = -1/2\ \mathbf{s}^T\mathbf{C}\mathbf{s}$, where the weight matrix $\mathbf{C}$ is a real symmetric matrix, and $\mathbf{s}$ is the vector of the units' binary states in a particular global configuration.

In a DUBM each stochastic directional unit takes on values on the unit circle. We associate with unit j a random variable Z_j; a particular state of j is described by a complex number with magnitude one and direction, or phase τ_j: $z_j = e^{i\tau_j}$.

The weights of a DUBM also take on complex values. The weight from unit k to unit j is: $w_{jk} = b_{jk}e^{i\theta_{jk}}$. The activation function of a unit is a generalization of the dot product to the complex domain: the net input $x_j = \mathbf{z} \cdot \mathbf{w_j} = \sum_k z_k w_{jk}^*$, where $\mathbf{z}$ is the vector of the units' complex states in a particular global configuration, and the asterisk indicates the complex conjugate operation. Figure 1 shows an example of this multiplication operation. We constrain the weight matrix W to be Hermitian: $W^T = W^*$, where the diagonal elements of the matrix are zero. Note that if the components are real, then $\mathbf{W}^T = \mathbf{W}$, which is a real symmetric matrix. Thus, the Hermitian form is a natural generalization of weight symmetry to the complex domain.

This definition of $\mathbf{W}$ leads to a Hermitian quadratic form that generalizes the real quadratic

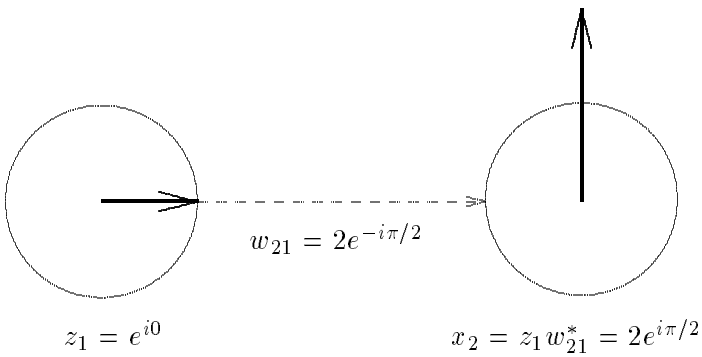


Figure 1: This figure shows the effect of using complex weights. Unit 1 is in state z_1, and the weight from unit 1 to 2 is w_{21}. Multiplying the state of unit 1 by the complex conjugate of this weight doubles the magnitude and rotates the phase by $\pi/2$ radians.

form of the Hopfield energy function:

$$E(\mathbf{z}) \;=\; -1/2\ \mathbf{z}^{*T}\mathbf{W}\mathbf{z} \;=\; -1/2\ \sum_{j,k} z_j z_k^* w_{jk} \tag{1}$$

Noest (1988) independently proposed this energy function. It is similar to that used in Fradkin, Huberman, and Shenker's (1978) generalization of the XY model of statistical mechanics to allow arbitary weight phases θ_{jk}, and coupled oscillator models, e.g., Baldi and Meir (1990). We discuss below the relationship between DUBMs and these models.

2.2 The Circular Normal Distribution

We can define a probability distribution over the possible states of a stochastic network by using the Boltzmann factor. In a binary-unit Boltzmann machine, a binary random variable S_j describes the state of unit j, and it has the relative probability distribution:

$$\frac{p(S_j = 1)}{p(S_j = 0)} \;=\; \frac{e^{-\beta E(S_j=1)}}{e^{-\beta E(S_j=0)}} \;=\; e^{\beta x_j}$$

where $x_j \;=\; \sum_k w_{jk} s_k$ is the net input to unit j, and $\beta = 1/T$. The probability that a unit is on is thus described by the familiar logistic function:

$$p(S_j = 1) \;=\; (1 + e^{-\beta x_j})^{-1} \;=\; \sigma(\beta x_j)$$

We use a similar method to define a probability distribution over the states of directional units; it turns out that this distribution is well-known for directional data. In a DUBM, we can describe the energy as a function of the state of a particular unit j:

$$\begin{aligned} E(Z_j = z_j) \;&=\; -1/2\ [\sum_k z_j z_k^* w_{jk} + \sum_k z_k z_j^* w_{kj}] \\ &=\; -1/2\ [z_j x_j^* + (z_j x_j^*)^*] \end{aligned} \tag{2}$$

where

$$x_j = \sum_k z_k w^*_{jk} \tag{3}$$

We can consider x_j as the net input to unit j; it captures the relevant information in the local field of the unit. We denote the magnitude and phase of x_j as a_j and α_j respectively, such that

$$x_j \;=\; a_j e^{i\alpha_j} \tag{4}$$

Using Equations 2, 3, and 4, we find that $E(Z_j = z_j) \;=\; -a_j \cos(\tau_j - \alpha_j)$. Applying the Boltzmann factor, the probability that unit j is in a particular state is proportional to:

$$p(Z_j = z_j) \;\propto\; e^{-\beta E(Z_j = z_j)} \;=\; e^{\beta a_j \cos(\tau_j - \alpha_j)} \tag{5}$$

This probability distribution for a unit's state corresponds to a distribution known as the *von Mises*, or *circular normal*, distribution (Mardia, 1972) that in many ways acts on a circular range as the Gaussian distribution acts on a linear (i.e., non-circular) range.[1] Two parameters completely characterize this distribution: a mean direction $\overline{\tau} = (0, 2\pi]$ and a concentration parameter $m > 0$ that behaves like the reciprocal of the variance of a Gaussian distribution on a linear random variable. The probability density function of a circular normal random variable Z is:

$$p(\tau;\, \overline{\tau}, m) \;=\; \frac{1}{2\pi I_0(m)}\, e^{m \cos(\tau - \overline{\tau})} \tag{6}$$

The normalization factor $I_0(m)$ is the modified Bessel function of the first kind and order zero.[2]

From Equation 5, we see that if a unit adopts states according to its contribution to the system energy, it will be a circular normal variable with mean direction α_j and concentration parameter $m_j \;=\; \beta a_j$. These parameters are directly determined by the net input to the unit.

Figure 2 shows a circular normal density function for Z_j, the state of unit j. This figure also shows the expected value of its stochastic state, which we define as:

$$y_j \;= < Z_j > = \; r_j e^{i\gamma_j} \tag{7}$$

where γ_j, the phase of y_j, is the mean direction and r_j, the magnitude of y_j, is the *resultant length*. For a circular normal random variable, its expected direction γ_j is simply the mean direction of the random variable:

$$\gamma_j \;=\; \overline{\tau_j} \tag{8}$$

[1] The circular normal resembles the Gaussian in that (a) the maximum likelihood estimate of the mean direction of a random variable x is the sample mean if and only if x is a circular normal random variable; and (b) the circular normal distribution is the maximum entropy distribution if a mean direction and circular variance are fixed. Some of the other important properties of the Gaussian, such as the central limit theorem, apply to another circular distribution, the *wrapped normal*.

[2] An integral representation of this function is $I_0(m) = \frac{1}{\pi}\int_0^\pi e^{\pm m\cos\theta} d\theta$. It can be computed by numerical routines.

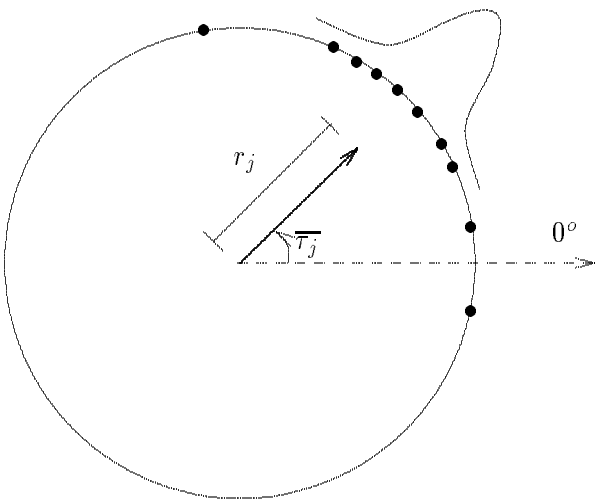


Figure 2: This figure shows a circular normal density function laid over a unit circle. The dots along the circle represent samples of the circular normal random variable Z_j. The expected direction of Z_j, $\overline{\tau_j}$, is $\pi/4$; its resultant length r_j is proportional to m_j. If m_j is large, then most of the samples of Z_j will be concentrated on a small arc about $\overline{\tau_j}$, and $r_j \rightarrow 1$. As m_j decreases, the sample spread increases, and $r_j \rightarrow 0$.

The calculation of r_j, the resultant length, is more involved. From Mardia (1972) we find that[3]:

$$r_j = \frac{I_1(m_j)}{I_0(m_j)} \tag{9}$$

From the figure we can see that when most of the samples of Z_j are concentrated on a small arc about the mean, r_j will approach length one. This corresponds to a large concentration parameter ($m_j = \beta a_j$), which means that in Equation 6, small deviations from the mean direction will produce much smaller exponents than that of the mean, and the probability distribution will be concentrated about the mean. Conversely, for small m_j, the distribution approaches the uniform distribution on the circle, and the resultant length falls toward zero. For a uniform distribution, $r_j = 0$. A plot of r_j against m_j is shown in Figure 3.

Note that the concentration parameter for a unit's circular normal density function is the product of a_j, the magnitude of its input, and β, the reciprocal of the system temperature. Higher temperatures will thus have the effect of making this distribution more uniform, just as they do in a binary-unit Boltzmann machine.

[3] An integral representation of the modified Bessel function of the first kind and order k is $I_k(m) = \frac{1}{\pi}\int_0^{\pi} e^{m\cos\theta}\cos(k\theta)d\theta$. Note that $I_1(m) = dI_0(m)/dm$.

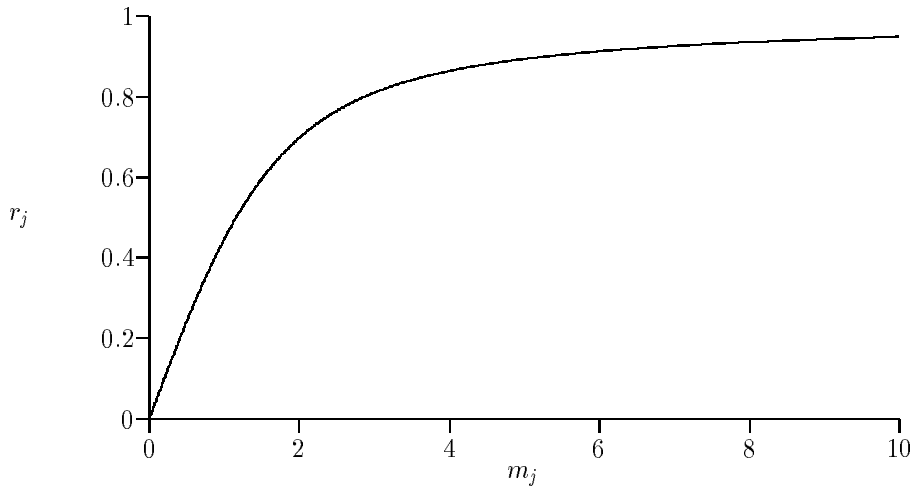


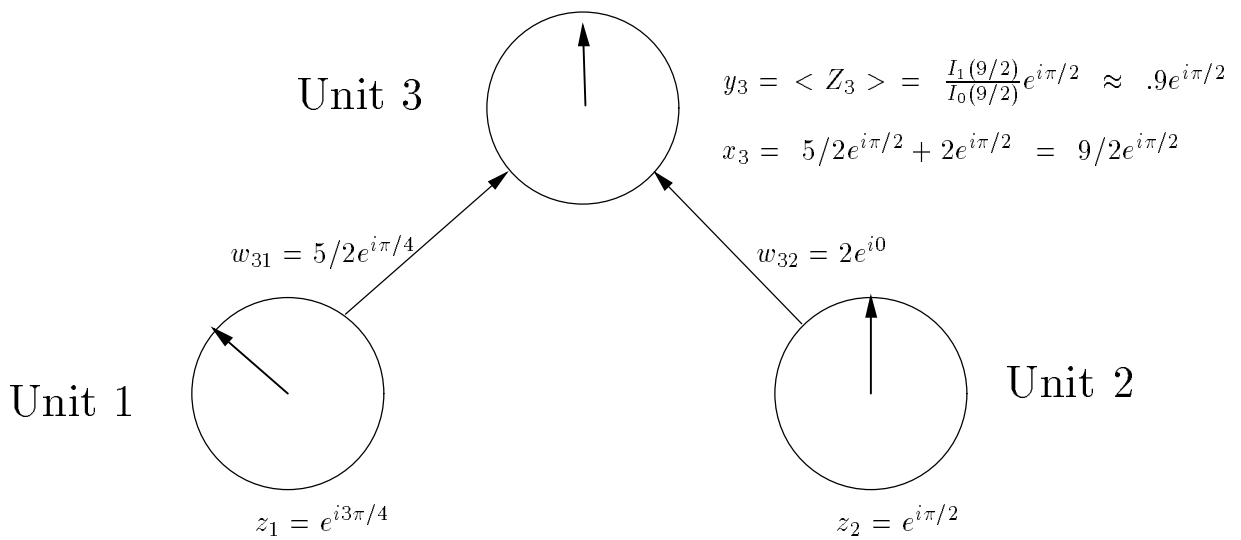
Figure 3: This figure shows the function defined by Equation 9 that maps m_j, the concentration parameter associated with the circular normal distribution of unit j, to r_j, its resultant length.

3 Emergent Properties of a DUBM

3.1 Evidence accumulation and representing certainty

A network of directional units as defined above contains two important emergent properties. The first property is that the magnitude of the net input to unit j describes the extent to which its various inputs "agree". Intuitively, one can think of each component $z_k w^*_{jk}$ of the sum that comprises x_j as predicting a phase for unit j. When the phases of these components are equal, the magnitude of x_j, a_j, is maximized. If these phase predictions are far apart, then they will act to cancel each other out, and produce a small a_j. Given x_j, we can compute the expected value of the output of unit j (Equation 11). The expected direction of the unit roughly represents the weighted average of the phase predictions, while the resultant length, a number between zero and one, is a monotonic function of a_j and hence describes the agreement between the various predictions. Figure 4 illustrates this property with two examples showing the computation of the input and expected output of a unit in a simple 3-unit DUBM.

The key idea here is that the resultant length directly describes the degree of certainty in the expected direction of unit j. Thus, a DUBM naturally incorporates a representation of the system's confidence in a value. This ability to combine several sources of evidence, and not only represent a value but also describe the certainty of that value is an important property that may be useful in a variety of domains.



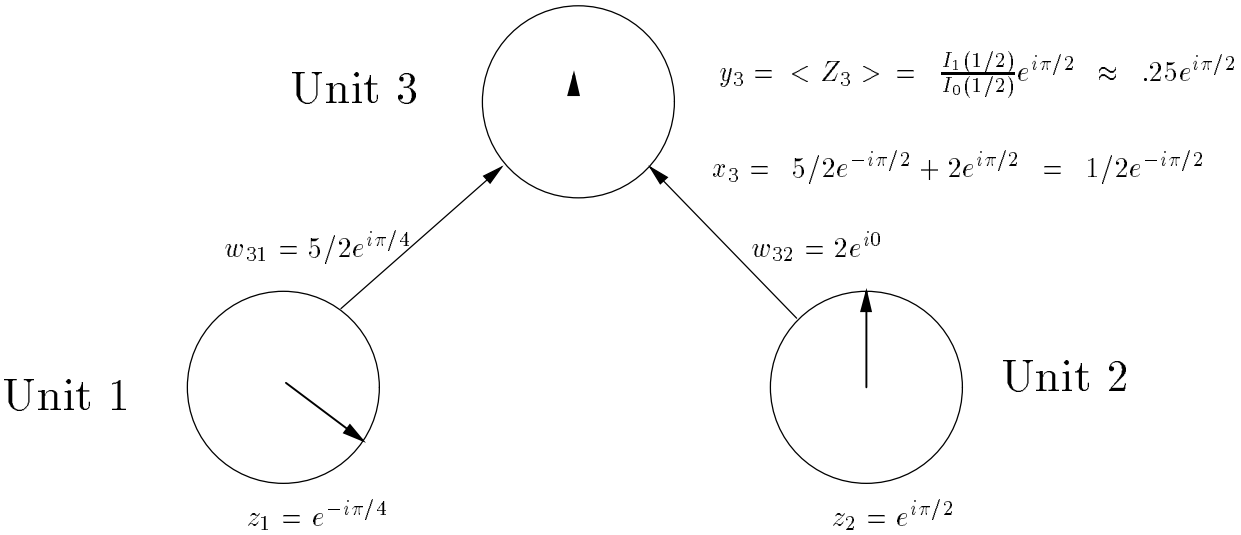


Figure 4: These two diagrams graphically depict the computation of the input and expected output of a unit. Unit 3 is connected to units 1 and 2. Its input, x_3, is computed as described in Equation 3, while its expected value, y_3, can be computed directly from its input, as described in Equations 8 and 9. In the top diagram, the input from the units 1 and 2 "agree", i.e., their contributions to x_3 have the same phase. The second diagram shows the effect of rotating the state of unit 1 by π: the phases of the two components of x_j no longer agree, resulting in a smaller magnitude, and hence a smaller resultant length.

3.2 Rotation invariance

The second emergent property is that the DUBM energy is globally *rotation-invariant—E* is unaffected when the same rotation is applied to all units' states in the network. This can be easily shown from Equation 1. Rotating a unit state z_j by some phase δ is equivalent to multiplying z_j by $e^{i\delta}$. If we apply this to each unit, then each $e^{i\delta}$ will be cancelled out in the $z_j z_k^*$ term in Equation 1. For each DUBM configuration, there is thus an equivalence class of configurations which have the same energy.

In a similar way, we find that the magnitude of x_j is *rotation-invariant*. That is, when we obtain a new input, x'_j by translating the phases of all of the other units by some phase δ, the magnitude a_j is unaffected:

$$x'_j = \sum_k b_{jk} e^{i(\tau_k + \delta - \theta_{jk})} = a_j e^{i(\alpha_j + \delta)} = x_j e^{i\delta}$$

This property underlies one of the key advantages of the representation: both the magnitude of a unit's state as well as system energy depend on the *relative* rather than *absolute* phases of the units.

4 Deterministic Directional Unit Boltzmann Machines

4.1 Approximating the stochastic system

Calculating properties of a large stochastic system is time-consuming: a simulation of such a system must be allowed to run for a sufficient length of time to build up statistics of the probability distribution across the possible configurations. Just as in deterministic binary-unit Boltzmann machines (Peterson and Anderson, 1987; Hinton, 1989), we can greatly reduce the computational time required if we invoke the mean-field approximation, which states that once the system has reached equilibrium, the stochastic variables can be approximated by their mean values. In this approximation, the variables are treated as independent, and the system probability distribution is simply the product of the probability distributions for the individual units.[4]

Gislén, Peterson, and Söderberg (1992) originally proposed a mean-field theory for networks of directional (or "rotor") units, but only considered the case of real-valued weights. They derived the mean-field consistency equations by using the saddle-point method to approximate the *partition function* of the system: $\int e^{-\beta E(\boldsymbol{\theta})} d\boldsymbol{\theta}$, where $\boldsymbol{\theta}$ indexes possible configurations of the system. Our approach provides an alternative, perhaps more intuitive derivation, due to the use of the circular normal distribution.

We can directly describe these mean values based on the circular normal interpretation.

[4] The tradeoff for the increase in computational speed of the mean-field approximation is a decrease in the number of states that can be represented, and a loss of information concerning the correlations between units' states.

We still denote the net input to a unit j as x_j:

$$x_j \;=\; \sum_k y_k w^*_{jk} \;=\; a_j e^{i\alpha_j} \tag{10}$$

Once equilibrium has been reached, the state of unit j is y_j, the expected value of Z_j given the mean-field approximation. We have already shown that Z_j is a circular normal random variable with mean α_j and concentration parameter βa_j; thus from Equations 8 and 9:

$$y_j \;=\; r_j e^{i\gamma_j} \;=\; \frac{I_1(\beta a_j)}{I_0(\beta a_j)} e^{i\alpha_j} \tag{11}$$

In the stochastic as well as the deterministic system, units evolve to minimize the *free energy*, $F \;=\; <E> - TH$. We calculate H, the entropy of the system, using the circular normal distribution and the mean-field approximation. The entropy of a circular normal random variable Z is: $H(Z) \;=\; -mr + \log(2\pi I_0(m))$. Since under the mean-field approximation, the units are considered independent, the entropy of the mean-field DUBM is simply the sum of the entropies of the individual units' states. The mean-field free energy is:

$$F_{MF} \;=\; \sum_{j<k} -r_j r_k b_{jk} \cos(\gamma_j - \gamma_k + \theta_{jk}) \;-\; T \sum_j [-\beta a_j r_j + \log(2\pi I_0(\beta a_j))] \tag{12}$$

We can derive mean-field consistency equations for x_j and y_j by minimizing F_{MF} with respect to each variable independently, i.e., by finding when the partial derivatives of F_{MF} with respect to each of these variables equal zero. The resulting equations match the mean-field equations (Equations 10 and 11) derived directly from the circular normal probability density function. They also match the special case derived by Gislén *et al.* for real-valued weights.

4.2 Running a deterministic DUBM

We have implemented a DUBM using the mean-field approximation. The goal is to solve for a consistent set of $\mathbf{x}$ and $\mathbf{y}$ values; there are several possible methods to settle the network to this state. One is to perform asynchronous updates of Equations 10 and 11. We have selected an alternative method, performing synchronous updates of the discrete-time approximation of the set of differential equations based on the net input to each unit j.

We update the x_j variables using the following differential equation:

$$\frac{dx_j}{dt} \;=\; -x_j + \sum_k y_k w^*_{jk} \tag{13}$$

which has Equation 10 as its steady-state solution.

In order to adapt x_j, we actually adapt its real and imaginary components: $x_j^R = a_j \cos\alpha_j$ and $x_j^I = a_j \sin\alpha_j$. The steady-state solutions for these two quantities are equivalent to, and

follow directly from Equation 10, and their update equations follow directly from Equation 13:

$$\begin{aligned}
\frac{dx_j^R}{dt} &= -x_j^R + \sum_k Re(y_k^* w_{jk}) = -x_j^R + \sum_k r_k b_{jk} \cos(\gamma_k - \theta_{jk}) \\
\frac{dx_j^I}{dt} &= -x_j^I + \sum_k Im(y_k^* w_{jk}) = -x_j^I + \sum_k r_k b_{jk} \sin(\gamma_k - \theta_{jk})
\end{aligned}$$

We use synchronous updates of these differential equations. At each step, we update x_j^R and x_j^I:

$$\begin{aligned}
x_j^R(t) &= (1-\eta) x_j^R(t-1) + \eta \sum_k r_k(t-1) b_{jk} \cos(\gamma_k(t-1) - \theta_{jk}) \\
x_j^I(t) &= (1-\eta) x_j^I(t-1) + \eta \sum_k r_k(t-1) b_{jk} \sin(\gamma_k(t-1) - \theta_{jk})
\end{aligned}$$

where $0 < \eta < 1$ is a variable step-size parameter that is selected to encourage the system to approach a steady-state solution while avoiding oscillations. We then compute the magnitude and phase of the input to unit j: $a_j(t) = \sqrt{(x_j^R(t))^2 + (x_j^I(t))^2}$ and $\alpha_j(t) = \arctan\left(\frac{x_j^I(t)}{x_j^R(t)}\right)$. These are then used to directly compute the state—$r_j(t)$ and $\gamma_j(t)$—of each unit j (see Equation 11). In the simulations, we typically set $\eta = 0.2$. We also use simulated annealing to help find good minima of the free energy F. This is implemented using an annealing schedule which exponentially lowers the temperature until $T = 1$.

Just as for the Hopfield binary-state network, it can be shown that the free energy F_{MF} always decreases during the dynamical evolution described in Equation 13. The equilibrium solutions are free energy minima. The Appendix contains a convergence proof for F_{MF}, in the style of Hopfield (1984).

5 Learning in DUBMs

The units in a DUBM can be arranged in a variety of architectures. The appropriate method for determining weight values for the network depends on the particular class of network architecture. In a standard autoassociative network, containing a single set of interconnected units as in a Hopfield network, the weights can be set directly from the training patterns. If hidden units are required to perform a task, then an algorithm for learning the weights is required. We have explored two learning algorithms. The first is a generalization of recurrent back-propagation (Pineda, 1987; Almeida, 1987) to a network of directional units. The complex representations in a DUBM complicates this algorithm, as it requires several matrix inversions. The second algorithm generalizes the Boltzmann machine training algorithm (Ackley, Hinton and Sejnowski, 1985; Peterson and Anderson, 1987) to these networks. This algorithm is considerably simpler; we thus focus on the second algorithm here, and results obtained from it are described in the following section.

We assume that the DUBM contains two sets of units, visible and hidden, and concentrate on a particular type of architecture, in which the *visible* units are split into two sets, *input* and *output*. The network is trained using a set of patterns, each specifying values for the input units and target values for each of the output units. We can regard the target output values as describing a desired set of probabilities. If we take the output units as representing a probability distribution across a set of directional values, then an appropriate error measure is the relative entropy of these probability distributions. This is the asymmetric divergence, or Kullback information measure used in binary-unit Boltzmann machines. The derivation of the learning rule for a DUBM parallels that for the standard Boltzmann machine. Let ω and ν index possible configurations of the output and input units, respectively. The goal is to make the network learn assocations from ν to ω such that the network's distribution $P^{\omega|\nu}$ resembles the desired distribution $R^{\omega|\nu}$ for each ν. If the possible inputs ν occur with probability p^ν, then the error measure, G, is:

$$\begin{aligned} G &= \int_\nu p^\nu \int_\omega R^{\omega|\nu} \log(R^{\omega|\nu}/P^{\omega|\nu})\, d\omega\, d\nu \\ &= G_0 - \int_\nu p^\nu \int_\omega R^{\omega|\nu} \log P^{\omega|\nu}\, d\omega\, d\nu \end{aligned}$$

In the stochastic network, $P^{\omega|\nu} = e^{-\beta F^{\omega\nu}}/e^{-\beta F^\nu}$, where $F^{\omega\nu}$ is the minimum free energy when the input units are in configuration ν and the output units are clamped to configuration ω, and F^ν is the minimum free energy when only the input units are set. If we assume that the training set specifies the occurence probabilities p^ν, as well as the probabilities $R^{\omega|\nu}$ for the ω and ν, then G simplifies to the following expression:

$$G = G_0 + \beta[\overline{F^{\omega\nu}} - \overline{F^\nu}]$$

Thus, the objective reduces to minimizing the difference between $F^{\omega\nu}$ and F^ν, averaged over the training set. For the mean-field DUBM, we use this same objective, where the free energies are now the mean-field free energies, as described in Section 4.1 (see Equation 12).

We differentiate this objective function with respect to the magnitude and direction of the complex-valued weights to determine the weight update rule. The partial derivative of G with respect to a weight is the difference between the partials of the two mean-field free energies, $F^{\omega\nu}_{MF}$ and F^ν_{MF}, with respect to the weight. On a given training case, the required derivatives are given by:

$$\begin{aligned} \partial F_{MF}/\partial b_{jk} &= -r_j r_k \cos(\gamma_j - \gamma_k + \theta_{jk}) \\ \partial F_{MF}/\partial \theta_{jk} &= r_j r_k b_{jk} \sin(\gamma_j - \gamma_k + \theta_{jk}) \end{aligned}$$

Note that the derivatives only involve the explicit derivatives of F_{MF} with respect to the weight parameters because we have settled to equilibrium (Hinton, 1989; Baldi and Pineda, 1991). The learning algorithm uses these gradients to find weight values that will minimize the objective G over a training set.

6 Experimental Results

The focus of this paper has been on the theoretical ideas underlying a generalization of a Boltzmann machine to directional units. We present below some illustrative examples to show that an adaptive network of directional units can be used in a range of paradigms, including associative memory, input/output mappings, and pattern completion.

6.1 Simple autoassociative DUBMs

The first set of experiments considers a simple autoassociative DUBM, which contains no hidden units, and the units are fully connected. As in a standard Hopfield network, the weights are set directly from the training patterns; they equal the superposition of the outer product of the patterns:

$$w_{jk} = \frac{1}{N} \sum_{p=1}^{N} y_j^p y_k^{p*} \tag{14}$$

where p indexes the training patterns.

We have experimented with a number of different networks. Figure 5 shows a network containing 30 units that has stored 3 patterns. The weights shown are generated as described above; the figure also shows two examples of the network settling from a corrupted version of one of the stored patterns to a state near the pattern. These patterns thus form stable attractors, as the network can perform pattern completion and clean-up from noisy inputs. The rotation-invariance property of the energy function allows any rotated version of a training pattern to also act as an attractor.

We have run several experiments with simple autoassociative DUBMs. The empirical results parallel those for binary-unit autoassociative networks. Using an error criterion that measures the distance from an equilibrium state to the closest stored pattern, we find that the mean error rapidly increases as the number of stored patterns increases. For example, the performance of the network shown in Figure 5 rapidly degrades for more than 4 orthogonal patterns; the patterns themselves no longer act as fixed-points, and many random initial states end in states far from any stored pattern. In addition, more orthogonal patterns can be stored than random patterns. See Noest (1988) for an analysis of the capacity of an autoassociative DUBM with sparse and asymmetric connections.

6.2 Learning input/output mappings

We have also used the mean-field DUBM learning algorithm to learn the weights in networks containing hidden units. Due to the presence of the hidden units, the network must settle via the synchronous updating procedure described in Section 4.2 on each training pattern, first with the input clamped, and then with both the input and target output clamped.

We have experimented with a task that is well-suited to a directional representation. There is a single-jointed robot arm, anchored at a point, as shown in Figure 6. The input consists of two angles: the angle between the first arm segment and the positive x-axis (α),

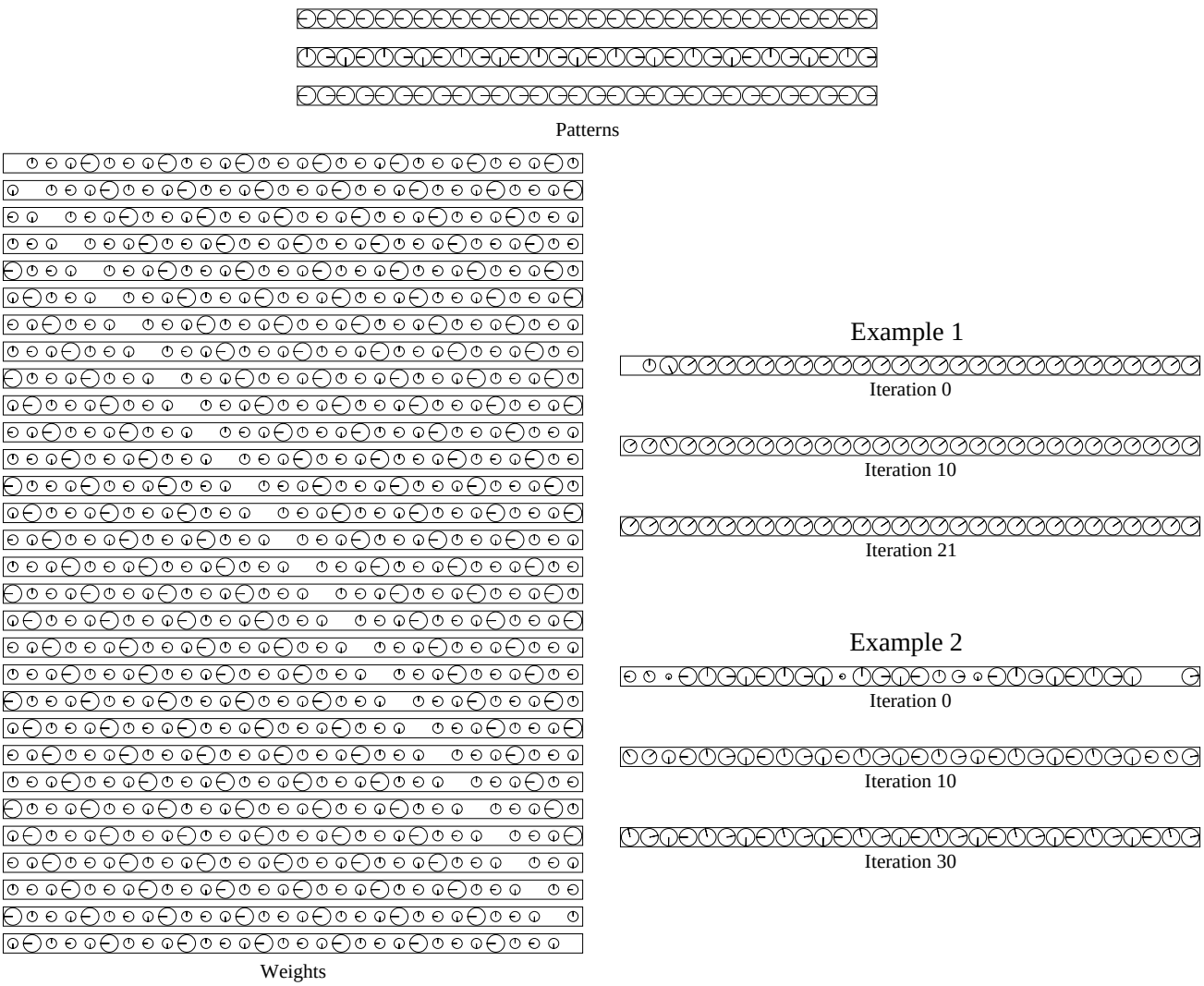


Figure 5: This figure shows the patterns and weights stored in an autoassociative network of 30 directional units. The area of each circle is proportional to the magnitude of the complex value, and the orientation of the notch indicates its phase. The weights are the outer product of the 3 patterns shown at the top. Each row shows the incoming weights to a particular unit. Note that the self weights (along the diagonal) have zero magnitude, and the symmetry of the diagram about the diagonal reflects the Hermitian property of the weights. On the right are two examples of the network settling to a fixed-point. The first example ends up near the first of the three stored patterns after 21 iterations (modulo a global rotation). The second example is considerably corrupted; it requires 30 iterations to approach the second stored pattern.

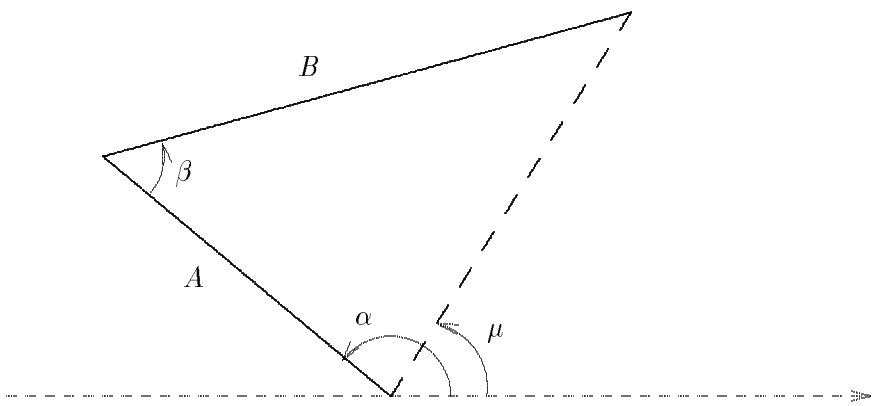


Figure 6: A graphical display of a training case for the robot arm problem. The arm is anchored on the horizontal axis, and the lengths of its two segments, A and B, are fixed. The two angles, α and β, are given as input for each training case, and the target output for the net is the angle μ.

and the angle between the two arm segments (β). The two segments each have a fixed length, A and B; these are not explicitly given to the network. The output is the angle between the line connecting the two ends of the arm and the x-axis (μ). This target angle is related in a complex, non-linear way to the input angles—the network must learn to approximate the following trigonometric relationship:

$$\mu = \arctan\left(\frac{A\sin\alpha - B\sin(\alpha+\beta)}{A\cos\alpha - B\cos(\alpha+\beta)}\right)$$

With 500 training cases, a DUBM with 2 input units and 8 hidden units was able to learn the task so that it could accurately estimate μ for novel patterns. The learning required 200 iterations of the conjugate gradient training algorithm. On each of 100 testing patterns, the resultant length of the output unit exceeded .85, and the mean error on the angle was less than .05 radians. The network can also learn the task with as few as 5 hidden units, with a concomitant decrease in learning speed. The compact nature of this network shows that the directional units form a natural, efficient representation for this problem.

6.3 Complex pattern completion

Our earlier work describes a large-scale DUBM that attacks a difficult problem in computer vision: image segmentation. In MAGIC (Mozer, Zemel and Behrmann, 1992; Mozer et al., 1992), directional values are used to represent alternative labels that can be assigned to image features. The goal of MAGIC is to learn to assign appropriate object labels to a set of image features (e.g., edge segments) based on a set of examples. The idea is that the features of a given object should have consistent phases, with each object taking on its own

phase. The units in the network are arranged into two layers—feature and hidden—and the computation proceeds by randomly initializing the phases of the units in the feature layer, and settling on a labeling through a relaxation procedure. The units in the hidden layer learn to detect spatially local configurations of the image features that are labeled in a consistent manner across the training examples.

MAGIC successfully learns to segment novel scenes consisting of overlapping geometric objects. The emergent DUBM properties described in Section 3 are essential to MAGIC's ability to perform this task. The evidence accumulation property allows units to act as feature detectors, expressing constraints in their complex-valued weight patterns. The complex weights are necessary in MAGIC, as the weights encode statistical regularities in the relationships between image features, e.g., that two features typically belong to the same object (i.e., have similar phase values) or to different objects (i.e., are out of phase). The fact that a unit's resultant length reflects the certainty in a phase label allows the system to decide which phase labels to use when updating labels of neighboring features: the initially random phases are ignored, while confident labels are propagated. Finally, the rotation-invariance property allows the system to assign labels to features in a manner consistent with the relationships described in the weights, where it is the *relative* rather than *absolute* phases of the units that are important.

7 Discussion

We have presented a general formulation for networks of directional units, called directional-unit Boltzmann machines, or DUBMs. This formulation is based on an underlying probabilistic model, and we have developed and experimented with a mean-field version of the stochastic system. A deterministic DUBM provides a natural representation for both computed values (the phase of a unit), and the certainty associated with these values (the magnitude of the unit's state). This formulation provides a well-founded activation function for directional units, based on this mean-field approximation, as well as a convergence proof for the system dynamics. In addition, this type of network has many potential applications: it is well-suited to data that can naturally be represented as directions, such as vector fields that arise in optic flow or tangent field measurements of computer vision, as well as for information that can take advantage of the cyclic nature of the directional representation, and tasks where *relative* rather than *absolute* information is important.

Our work on directional units fits in a long tradition of work on phase-based representations. The well-known XY models of statistical mechanics, as well as coupled oscillator models, describe the dynamics of a population of oscillators in which each oscillator is described by a single parameter, its phase. Several researchers have recently studied such systems of coupled oscillators (Kammen, Holmes and Koch, 1989; Baldi and Meir, 1990), and have applied these methods to model experimental findings concerning oscillations and phase-locking of cortical neurons in cats (Gray et al., 1989; Eckhorn et al., 1988). This work relates to the suggestion (von der Malsburg, 1981) that synchronized oscillations provide a biologically plausible mechanism of labeling features through temporal correlations among

neural signals. Other related work – mentioned earlier – includes that of Gislén, Peterson, and Söderberg (1992), which describes a mean-field analysis for a system of directional units, and considers the general case where units assume multi-dimensional directional states on a hypersphere. Also, Noest (1988), proposed an energy function for phase-based units and analyzed the capacity of an autoassociative network which minimizes this energy function.

Several important distinctions exist between our work and these other models. First, just as a binary-unit Boltzmann machine extends the representational power of the Hopfield network by adding hidden units, our formulation extends the power of autoassociative coupled oscillator models. A DUBM can contain layers of hidden units, which enables the network to solve tasks that require a representation of higher-order relations between elements of input patterns, such as the robot-arm task described above.[5]

In addition, our formulation allows the weights to be complex-valued, which is important for encoding desired phase relationships between two directional units. This is a crucial property for systems looking for patterns of phase relationships, as was discussed in Section 6.3. One could conceivably encode phase relationships with scalar weights, but this would require additional connections between units, as well as an additional term in the energy function.

Finally, our work introduces the probabilistic interpretation of networks of directional units. We introduce the notion that the density function across directions for a given unit can be described by the circular normal, and show that the mean-field equations for such a unit correspond to the expected value of this distribution. This approach greatly simplifies the derivation of the deterministic system dynamics. Our work demonstrates that networks composed of directional units can learn interesting mappings. MAGIC, as described in (Mozer et al., 1992), is a large-scale DUBM that learns to segment novel images composed of a wide range of overlapping geometric objects.

One extension of the current model is appropriate for tasks that involve learning a mapping from a set of input patterns to a set of output patterns. We can consider a different type of architecture for these tasks, where the directional units are used in a strictly feedforward manner; such a network can then be trained by a modified form of the back-propagation training algorithm. In this case, the convergence properties of a DUBM as well as the correspondence between state probabilities and energies no longer apply. However, this approach has some advantages over previous methods of training networks of complex units, e.g., (Benvenuto and Piazza, 1992; Widrow, McCool and Ball, 1975). Several interesting properties of DUBMs, such as the rotation-invariance and the interpretation of the magnitude of a unit as capturing the agreement of its inputs, are embodied in the activation function of the network and are hence retained.

We are currently exploring an important extension to the model which will expand its representational capabilities. Radford Neal (personal communication, 1992) has suggested a method for combining binary and directional units. This expanded representation may be

[5]Note that deterministic DUBMs share the problem of deterministic binary-unit Boltzmann machines, in that their performance drops off in deep networks (Galland, 1992). This is largely due to the fact that the mean-field approximation breaks down as multiple layers introduce increasing correlations among the units.

useful in domains with directional data that is not present everywhere. For example, it can be directly applied to the object labeling problem explored in MAGIC. The binary aspect of the unit can describe whether a particular image feature is present or absent. This may enable the system to handle various complications, particularly labeling across gaps along the contour of an object. Finally, we are applying a DUBM network to the interesting and challenging problem of time-series prediction of wind directions.

8 Acknowledgements

The authors thank Geoffrey Hinton for his generous support and guidance. We thank Radford Neal for many valuable discussions that significantly influenced this work. We also thank Peter Dayan, Conrad Galland, Sue Becker, Steve Nowlan, and the other members of the Connectionist Research Group at the University of Toronto for helpful comments regarding this work. This research was supported by a grant from the Information Technology Research Centre of Ontario to Geoffrey Hinton, and NSF Presidential Young Investigator award IRI–9058450 and grant 90–21 from the James S. McDonnell Foundation to MM.

References

Ackley, D. H., Hinton, G. E., and Sejnowski, T. J. (1985). A learning algorithm for Boltzmann machines. *Cognitive Science*, 9:147–169.

Almeida, L. B. (1987). A learning rule for asynchronous perceptrons with feedback in a combinatorial environment. In *Proceedings, 1st First International Conference on Neural Networks*, volume 2, pages 609–618, San Diego, CA. IEEE.

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

Baldi, P. and Pineda, F. (1991). Contrastive learning and neural oscillations. *Neural Computation*, 3(4):526–533.

Benvenuto, N. and Piazza, F. (1992). On the complex backpropagation algorithm. *IEEE Transactions On Signal Processing*, 40(4):967–969.

Eckhorn, R., Bauer, R., Jordan, W., Brosch, M., Kruse, W., Munk, M., and Reitboek, H. K. (1988). Coherent oscillations: A mechanism of feature linking in the visual cortex. *Biological Cybernetics*, 60:121–130.

Fradkin, E., Huberman, B. A., and Shenker, S. H. (1978). Gauge symmetries in random magnetic systems. *Physical Review B*, 18(9):4789–4814.

Galland, C. C. (1992). *Self-organization using Deterministic Boltzmann Machine learning.* PhD thesis, Physics Department of the University of Toronto.

Gislén, L., Peterson, C., and Söderberg, B. (1992). Rotor neurons: Basic formalism and dynamics. *Neural Computation*, 4(5):737–745.

Gray, C. M., Koenig, P., Engel, A. K., and Singer, W. (1989). Oscillatory responses in cat visual cortex exhibiting intercolumnar synchronization which reflects global properties. *Nature (London)*, 338:334–337.

Hinton, G. E. (1989). Deterministic Boltzmann learning performs steepest descent in weight-space. *Neural Computation*, 1(2):143–150.

Hopfield, J. J. (1982). Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences U.S.A.*, 79:2554–2558.

Hopfield, J. J. (1984). Neurons with graded response have collective computational properties like those of two-state neurons. *Proceedings of the National Academy of Sciences U.S.A.*, 81:3088–3092.

Kammen, D. M., Holmes, P. J., and Koch, C. (1989). Cortical oscillations in neuronal networks: Feed-back versus local coupling. In *Models of Brain Function*, pages 273–284. Cambridge University Press, Cambridge.

Mardia, K. V. (1972). *Statistics of Directional Data*. Academic Press, London.

Mozer, M. C., Zemel, R. S., and Behrmann, M. (1992). Learning to segment images using dynamic feature binding. In *Advances in Neural Information Processing Systems 4*, pages 436–443. Morgan Kaufmann, San Mateo, CA.

Mozer, M. C., Zemel, R. S., Behrmann, M., and Williams, C. K. I. (1992). Learning to segment images using dynamic feature binding. *Neural Computation*, 4(5):650–665.

Noest, A. J. (1988). Phasor neural networks. In *Neural Information Processing Systems*, pages 584–591, New York. AIP.

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

Pineda, F. J. (1987). Generalization of back propagation to recurrent and higher order neural networks. In *Proceedings of IEEE Conference on Neural Information Processing Systems*, Denver, Colorado. IEEE.

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

Widrow, B., McCool, J., and Ball, M. (1975). The complex LMS algorithm. *Proc. IEEE*, pages 719–720.

A Appendix

To show that F always decreases, we first differentiate E_{MF} with respect to time:

$$\begin{aligned} E_{MF} &= -1/2 \sum_{j,k} y_j y_k^* w_{jk} \\ \frac{dE_{MF}}{dt} &= -1/2 \sum_{j,k} \left[\frac{dy_j}{dt} y_k^* w_{jk} + y_j \frac{dy_k^*}{dt} w_{jk} \right] \\ &= -1/2 \sum_{j} \left[\frac{dy_j}{dt} \left(\sum_k y_k w_{jk}^* \right)^* + \frac{dy_j^*}{dt} \left(\sum_k y_k w_{jk}^* \right) \right] \end{aligned} \tag{15}$$

We then differentiate H_{MF} with respect to time:

$$\begin{aligned} H_{MF} &= \sum_j \left[-\beta a_j r_j + \log(2\pi I_0(\beta a_j)) \right] \\ \frac{dH_{MF}}{dt} &= \sum_j \left[-\beta \frac{da_j}{dt} r_j - \beta a_j \frac{dr_j}{dt} + \beta \frac{I_1(\beta a_j)}{I_0(\beta a_j)} \frac{da_j}{dt} \right] \\ &= \sum_j -\beta a_j \frac{dr_j}{dt} \end{aligned} \tag{16}$$

where we use the fact that $\frac{dI_0(m)}{dt} = I_1(m)\frac{dm}{dt}$, along with the definition of $r_j = \frac{I_1(\beta a_j)}{I_0(\beta a_j)}$.

Now consider (recalling that $x = ae^{i\alpha}$ and $y = re^{i\gamma} = re^{i\alpha}$):

$$\begin{aligned} x^* \frac{dy}{dt} &= ae^{-i\alpha} \left(\frac{dr}{dt} e^{i\alpha} + ire^{i\alpha} \frac{d\alpha}{dt} \right) \\ &= a\frac{dr}{dt} + ira\frac{d\alpha}{dt} \\ 1/2 \left[x^* \frac{dy}{dt} + x \frac{dy^*}{dt} \right] &= a\frac{dr}{dt} \end{aligned}$$

This last step follows from the fact that if $z = f + ig$, then $z + z^* = 2f$.

Substituting back into Equation 16, we find that:

$$\frac{dH_{MF}}{dt} = -1/2 \sum_j \beta \left[x_j^* \frac{dy_j}{dt} + x_j \frac{dy_j^*}{dt} \right] \tag{17}$$

Combining Equations 15 and 17, and using the definition of $\frac{dx_j}{dt} = -x_j + \sum_k y_k w_{jk}^*$ (see Equation 13):

$$\begin{aligned} \frac{dF_{MF}}{dt} &= \frac{dE_{MF}}{dt} - T\frac{dH_{MF}}{dt} \\ &= -1/2 \sum_j \left[\left((\sum_k y_k w_{jk}^*) - x_j \right) \frac{dy_j^*}{dt} + \left((\sum_k y_k w_{jk}^*)^* - x_j^* \right) \frac{dy_j}{dt} \right] \\ &= -1/2 \sum_j \left[\frac{dx_j}{dt} \frac{dy_j^*}{dt} + \frac{dx_j^*}{dt} \frac{dy_j}{dt} \right] \end{aligned}$$

We can expand these products as follows:

$$\begin{aligned}
\frac{dx}{dt} &= \frac{da}{dt}e^{i\alpha} + iae^{i\alpha}\frac{d\alpha}{dt} \\
\frac{dy^*}{dt} &= \frac{dr}{dt}e^{-i\alpha} - ire^{-i\alpha}\frac{d\alpha}{dt} \\
\\
\frac{dx}{dt}\frac{dy^*}{dt} &= \frac{dr}{dt}\frac{da}{dt} + ar\left(\frac{d\alpha}{dt}\right)^2 + i\left[a\frac{dr}{dt}\frac{d\alpha}{dt} - r\frac{da}{dt}\frac{d\alpha}{dt}\right] \\
&= \frac{dr}{da}\left(\frac{da}{dt}\right)^2 + ar\left(\frac{d\alpha}{dt}\right)^2 + i\left[a\frac{dr}{dt}\frac{d\alpha}{dt} - r\frac{da}{dt}\frac{d\alpha}{dt}\right]
\end{aligned}$$

where we have used the fact that since r depends solely on a, $\frac{dr}{dt} = \frac{dr}{da}\frac{da}{dt}$.

Finally,

$$\begin{aligned}
\frac{dF_{MF}}{dt} &= -1/2\sum_j\left[\frac{dx_j}{dt}\frac{dy_j^*}{dt} + \left(\frac{dx_j}{dt}\frac{dy_j^*}{dt}\right)^*\right] \\
&= -1/2\sum_j\left[\frac{dr_j}{da_j}\left(\frac{da_j}{dt}\right)^2 + a_j r_j\left(\frac{d\alpha_j}{dt}\right)^2\right] \\
&\leq 0
\end{aligned}$$

The final step in this derivation follows because r_j is a monotonically increasing function of a_j (see Figure 3), and because a_j and r_j are always non-negative.