Communicated by Robert Jacobs

Factor Analysis Using Delta-Rule Wake-Sleep Learning

Radford M. Neal
Department of Statistics and Department of Computer Science, University of Toronto, Toronto M5S 1A1, Canada

Peter Dayan
Department of Brain and Cognitive Sciences, Massachusetts Institute of Technology, Cambridge, MA, 02139 U.S.A.

We describe a linear network that models correlations between real-valued visible variables using one or more real-valued hidden variables—a factor analysis model. This model can be seen as a linear version of the Helmholtz machine, and its parameters can be learned using the wake-sleep method, in which learning of the primary generative model is assisted by a recognition model, whose role is to fill in the values of hidden variables based on the values of visible variables. The generative and recognition models are jointly learned in wake and sleep phases, using just the delta rule. This learning procedure is comparable in simplicity to Hebbian learning, which produces a somewhat different representation of correlations in terms of principal components. We argue that the simplicity of wake-sleep learning makes factor analysis a plausible alternative to Hebbian learning as a model of activity-dependent cortical plasticity.

1 Introduction

Statistical structure in a collection of inputs can be found using purely local Hebbian learning rules (Hebb, 1949), which capture second-order aspects of the data in terms of principal components (Linsker, 1988; Oja, 1989). This form of statistical analysis has therefore been used to provide a computational account of activity-dependent plasticity in the vertebrate brain (e.g., von der Malsburg, 1973; Linsker, 1986; Miller, Keller, & Stryker, 1989).

There are reasons, however, that principal component analysis may not be an adequate model for the generation of cortical receptive fields (e.g., Olshausen & Field, 1996). Furthermore, a Hebbian mechanism for performing principal component analysis would accord no role to the top-down (feedback) connections that always accompany bottom-up connections (Felleman & Van Essen, 1991). Hebbian learning must also be augmented by extra mechanisms in order to extract more than just the first principal com-

ponent and in order to prevent the synaptic weights from growing without bound.

In this article, we present the statistical technique of factor analysis as an alternative to principal component analysis and show how factor analysis models can be learned using an algorithm whose demands on synaptic plasticity are as local as those of the Hebb rule.

Our approach follows the suggestion of Hinton and Zemel (1994) (see also Grenander, 1976–1981; Mumford, 1994; Dayan, Hinton, Neal, & Zemel, 1995; Olshausen & Field, 1996) that the top-down connections in the cortex might be constructing a hierarchical, probabilistic "generative" model, in which dependencies between the activities of neurons reporting sensory data are seen as being due to the presence in the world of hidden or latent "factors." The role of the bottom-up connections is to implement an inverse "recognition" model, which takes low-level sensory input and instantiates in the activities of neurons in higher layers a set of likely values for the hidden factors, which are capable of explaining this input. These higher-level activities are meant to correspond to significant features of the external world, and thus to form an appropriate basis for behavior.

We refer to such a combination of generative and recognition models as a Helmholtz machine. The simplest form of Helmholtz machine, with just two layers, linear units, and gaussian noise, is equivalent to the factor analysis model, a statistical method that is used widely in psychology and the social sciences as a way of exploring whether observed patterns in data might be explainable in terms of a small number of unobserved factors. Everitt (1984) gives a good introduction to factor analysis and to other latent variable models.

Even though factor analysis involves only linear operations and gaussian noise, learning a factor analysis model is not computationally straightforward; practical algorithms for maximum likelihood factor analysis (Jöreskog, 1967, 1969, 1977) took many years to develop. Existing algorithms change the values of parameters based on complex and nonlocal calculations. A general approach to learning in Helmholtz machines that is attractive for its simplicity is the wake-sleep algorithm of Hinton, Dayan, Frey, and Neal (1995). We show empirically in this article that maximum likelihood factor analysis models can be learned by the wake-sleep method, which uses just the purely local delta rule in both of its two separate learning phases. The wake-sleep algorithm has previously been applied to the more difficult task of learning nonlinear models with binary latent variables, with mixed results. We have found that good results are obtained much more consistently when the wake-sleep algorithm is used to learn factor analysis models, perhaps because settings of the recognition model parameters that invert the generative model always exist.

In the factor analysis models we look at in this article, the factors are a priori independent, and this simplicity prevents them from reproducing interesting aspects of cortical structure. However, these results contribute to

the understanding of wake-sleep learning. They also point to possibilities for modeling cortical plasticity, since the wake-sleep approach avoids many of the criticisms that can be leveled at Hebbian learning.

2 Factor Analysis

Factor analysis with a single hidden factor is based on a generative model for the distribution of a vector of real-valued visible inputs, $\mathbf{x}$, given by

$$\mathbf{x} = \boldsymbol{\mu} + \mathbf{g}y + \boldsymbol{\epsilon}. \tag{2.1}$$

Here, y is the single real-valued factor, and is assumed to have a gaussian distribution with mean zero and variance one. The vector of "factor loadings," $\mathbf{g}$, which we refer to as the generative weights, expresses the way the visible variables are related to the hidden factor. The vector of overall means, $\boldsymbol{\mu}$, will, for simplicity of presentation, be taken to be zero in this article, unless otherwise stated. Finally, the noise, $\boldsymbol{\epsilon}$, is assumed to be gaussian, with a diagonal covariance matrix, Ψ, which we will write as

$$\Psi = \begin{pmatrix} \tau_1^2 & 0 & \dots & 0 \\ 0 & \tau_2^2 & \dots & 0 \\ & & \dots & \\ 0 & 0 & \dots & \tau_n^2 \end{pmatrix}. \tag{2.2}$$

The τ_j^2 are sometimes called the "uniquenesses," as they represent the portion of the variance in each visible variable that is unique to it rather than being explained by the common factor. We will refer to them as the generative variance parameters (not to be confused with the variance of the hidden factor in the generative model, which is fixed at one).

The model parameters, $\boldsymbol{\mu}$, $\mathbf{g}$, and Ψ, define a joint gaussian distribution for both the hidden factor, y, and the visible inputs, $\mathbf{x}$. In a Helmholtz machine, this generative model is accompanied by a recognition model, which represents the conditional distribution for the hidden factor, y, given a particular input vector, $\mathbf{x}$. This recognition model also has a simple form, which for $\boldsymbol{\mu} = 0$, is

$$y = \mathbf{r}^T\mathbf{x} + \nu, \tag{2.3}$$

where $\mathbf{r}$ is the vector of recognition weights and ν has a gaussian distribution with mean zero and variance σ^2. It is straightforward to show that the correct recognition model has $\mathbf{r} = [\mathbf{g}\mathbf{g}^T + \Psi]^{-1}\mathbf{g}$ and $\sigma^2 = 1 - \mathbf{g}^T[\mathbf{g}\mathbf{g}^T + \Psi]^{-1}\mathbf{g}$, but we presume that directly obtaining the recognition model's parameters in this way is not plausible in a neurobiological model.

The factor analysis model can be extended to include several hidden factors, which are usually assumed to have independent gaussian distributions

with mean zero and variance one (though we discuss other possibilities in section 5.2). These factors jointly produce a distribution for the visible variables, as follows:

$$\mathbf{x} = \boldsymbol{\mu} + \mathbf{G}\mathbf{y} + \boldsymbol{\epsilon}. \tag{2.4}$$

Here, $\mathbf{y}$ is the vector of hidden factor values, $\mathbf{G}$ is a matrix of generative weights (factor loadings), and $\boldsymbol{\epsilon}$ is again a noise vector with diagonal covariance. A linear recognition model can again be found to represent the conditional distribution of the hidden factors for given values of the visible variables. Note that there is redundancy in the definition of the multiple-factor model, caused by the symmetry of the distribution of $\mathbf{y}$ in the generative model. A model with generative weights $\mathbf{G}' = \mathbf{G}\mathbf{U}^T$, where $\mathbf{U}$ is any unitary matrix (for which $\mathbf{U}^T\mathbf{U} = \mathbf{I}$), will produce the same distribution for $\mathbf{x}$, with the hidden factors $\mathbf{y}' = \mathbf{U}\mathbf{y}$ again being independent, with mean zero and variance one. The presence of multiple solutions is sometimes seen as a problem with factor analysis in statistical applications, since it makes interpretation more difficult, but this is hardly an issue in neurobiological modeling, since easy interpretability of neural activities is not something that we are at liberty to require.

Although there are some special circumstances in which factor analysis is equivalent to principal component analysis, the techniques are in general quite different (Jolliffe, 1986). Loosely speaking, principal component analysis pays attention to both variance and covariance, whereas factor analysis looks only at covariance. In particular, if one of the components of $\mathbf{x}$ is corrupted by a large amount of noise, the principal eigenvector of the covariance matrix of the inputs will have a substantial component in the direction of that input. Hebbian learning will therefore result in the output, y, being dominated by this noise. In contrast, factor analysis uses ϵ_j to model any noise that affects only component j. A large amount of noise simply results in the corresponding τ_j^2 being large, with no effect on the output y. Principal component analysis is also unaffected by a rotation of the coordinate system of the input vectors, whereas factor analysis privileges the particular coordinate system in which the data are presented, because it is assumed in equation 2.2 that it is in this coordinate system that the components of the noise are independent.

3 Learning Factor Analysis Models with the Wake-Sleep Algorithm

In this section, we describe wake-sleep algorithms for learning factor analysis models, starting with the simplest such model, having a single hidden factor. We also provide an intuitive justification for believing that these algorithms might work, based on their resemblance to the Expectation-Maximization (EM) algorithm. We have not found a complete theoretical

proof that wake-sleep learning for factor analysis works, but we present empirical evidence that it usually does in section 4.

3.1 Maximum Likelihood and the EM Algorithm. In maximum likelihood learning, the parameters of the model are chosen to maximize the probability density assigned by the model to the data that were observed (the "likelihood"). For a factor analysis model with a single factor, the likelihood, L, based on the observed data in n independent cases, $\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}$, is obtained by integrating over the possible values that the hidden factors, $y^{(c)}$, might take on for each case:

$$L(\mathbf{g}, \Psi) = \prod_{c=1}^{n} p(\mathbf{x}^{(c)} \mid \mathbf{g}, \Psi) = \prod_{c=1}^{n} \int p(\mathbf{x}^{(c)} \mid y^{(c)}, \mathbf{g}, \Psi)\, p(y^{(c)})\, dy^{(c)} \quad (3.1)$$

where $\mathbf{g}$ and Ψ are the model parameters, as described previously, and $p(\cdot)$ is used to write probability densities and conditional probability densities. The prior distribution for the hidden factor, $p(y^{(c)})$, is gaussian with mean zero and variance one. The conditional distribution for $\mathbf{x}^{(c)}$ given $y^{(c)}$ is gaussian with mean $\boldsymbol{\mu} + \mathbf{g}y^{(c)}$ and covariance Ψ, as implied by equation 2.1.

Wake-sleep learning can be viewed as an approximate implementation of the EM algorithm, which Dempster, Laird, and Rubin (1977) present as a general approach to maximum likelihood estimation when some variables (in our case, the values of the hidden factors) are unobserved. Applications of EM to factor analysis are discussed by Dempster et al. (1977) and by Rubin and Thayer (1982), who find that EM produces results almost identical to those of Jöreskog's (1969) method, including getting stuck in essentially the same local maxima given the same starting values for the parameters.[1]

EM is an iterative method, in which each iteration consists of two steps. In the E-step, one finds the conditional distribution for the unobserved variables given the observed data, based on the current estimates for the model parameters. When the cases are independent, this conditional distribution factors into distributions for the unobserved variables in each case. For the single-factor model, the distribution for the hidden factor, $y^{(c)}$, in a case with observed data $\mathbf{x}^{(c)}$, is

$$p(y^{(c)} \mid \mathbf{x}^{(c)}, \mathbf{g}, \Psi) = \frac{p(y^{(c)}, \mathbf{x}^{(c)} \mid \mathbf{g}, \Psi)}{\int p(\mathbf{x}^{(c)} \mid y, \mathbf{g}, \Psi)\, p(y)\, dy}. \quad (3.2)$$

[1] It may seem surprising that maximum likelihood factor analysis can be troubled by local maxima, in view of the simple linear nature of the model, but this is in fact quite possible. For example, a local maximum can arise if a single-factor model is applied to data consisting of two pairs of correlated variables. The single factor can capture only one of these correlations. If the initial weights are appropriate for modeling the weaker of the two correlations, learning may never find the global maximum in which the single factor is used instead to model the stronger correlation.

In the M-step, one finds new parameter values, $\mathbf{g}'$ and Ψ', that maximize (or at least increase) the expected log likelihood of the complete data, with the unobserved variables filled in according to the conditional distribution found in the E-step:

$$\sum_{c=1}^{n} \int p(y^{(c)} \mid \mathbf{x}^{(c)}, \mathbf{g}, \Psi) \log \left[p(\mathbf{x}^{(c)} \mid y^{(c)}, \mathbf{g}', \Psi') \, p(y^{(c)}) \right] dy^{(c)}. \qquad (3.3)$$

For factor analysis, the conditional distribution for $y^{(c)}$ in equation 3.2 is gaussian, and the quantity whose expectation with respect to this distribution is required above is quadratic in $y^{(c)}$. This expectation is therefore easily found on a computer. However, the matrix calculations involved require nonlocal operations.

3.2 The Wake-Sleep Approach. To obtain a local learning procedure, we first eliminate the explicit maximization in the M-step of the quantity defined in equation 3.3 in terms of an expectation. We replace this by a gradient-following learning procedure in which the expectation is implicitly found by combining many updates based on values for the $y^{(c)}$ that are stochastically generated from the appropriate conditional distributions. We furthermore avoid the direct computation of these conditional distributions in the E-step by learning to produce them using a recognition model, trained in tandem with the generative model.

This approach results in the wake-sleep learning procedure, with the wake phase playing the role of the M-step in EM and the sleep phase playing the role of the E-step. The names for these phases are metaphorical; we are not proposing neurobiological correlates for the wake and sleep phases. Learning consists of interleaved iterations of these two phases, which in the context of factor analysis operate as follows:

Wake phase: From observed values for the visible variables, **x**, randomly fill in values for the hidden factors, **y**, using the conditional distribution defined by the current recognition model. Update the parameters of the generative model to make this filled-in case more likely.

Sleep phase: Without reference to any observation, randomly choose values for the hidden factors, **y**, from their fixed generative distribution, and then randomly choose "fantasy" values for the visible variables, **x**, from their conditional distribution given **y**, as defined by the current parameters of the generative model. Update the parameters of the recognition model to make this fantasy case more likely.

The wake phase of learning will correctly implement the M-step of EM (in a stochastic sense), provided that the recognition model produces the correct conditional distribution for the hidden factors. The aim of sleep phase learning is to improve the recognition model's ability to produce this

conditional distribution, which would be computed in the E-step of EM. If values for the recognition model parameters exist that reproduce this conditional distribution exactly and if learning of the recognition model proceeds at a much faster rate than changes to the generative model, so that this correct distribution can continually be tracked, then the sleep phase will effectively implement the E-step of EM, and the wake-sleep algorithm as a whole will be guaranteed to find a (local) maximum of the likelihood (in the limit of small learning rates, so that stochastic variation is averaged out).

These conditions for wake-sleep learning to mimic EM are too stringent for actual applications, however. At a minimum, we would like wake-sleep learning to work for a wide range of relative learning rates in the two phases, not just in the limit as the ratio of the generative learning rate to the recognition learning rate goes to zero. We would also like wake-sleep learning to do something sensible even when it is not possible for the recognition model to invert the generative model perfectly (as is typical in applications other than factor analysis), though one would not usually expect the method to produce exact maximum likelihood estimates in such a case.

We could guarantee that wake-sleep learning behaves well if we could find a cost function that is decreased (on average) by both the wake phase and the sleep phase updates. The cost would then be a Lyapunov function for learning, providing a guarantee of stability and allowing something to be said about the stable states. Unfortunately, no such cost function has been discovered, for either wake-sleep learning in general or wake-sleep learning applied to factor analysis. The wake and sleep phases each separately reduces a sensible cost function (for each phase, a Kullback-Leibler divergence), but these two cost functions are not compatible with a single global cost function. An algorithm that correctly performs stochastic gradient descent in the recognition parameters of a Helmholtz machine using an appropriate global cost function does exist (Dayan & Hinton, 1996), but it involves reinforcement learning methods for which convergence is usually extremely slow.

We have obtained some partial theoretical results concerning wake-sleep learning for factor analysis, which show that the maximum likelihood solutions are second-order Lyapunov stable and that updates of the weights (but not variances) for a single-factor model decrease an appropriate cost function. However, the primary reason for thinking that the wake-sleep algorithm generally works well for factor analysis is not these weak theoretical results but rather the empirical results in section 4. Before presenting these, we discuss in more detail how the general wake-sleep scheme is applied to factor analysis, first for a single-factor model and then for models with multiple factors.

3.3 Wake-Sleep Learning for a Single-Factor Model. A Helmholtz machine that implements factor analysis with a single hidden factor is shown

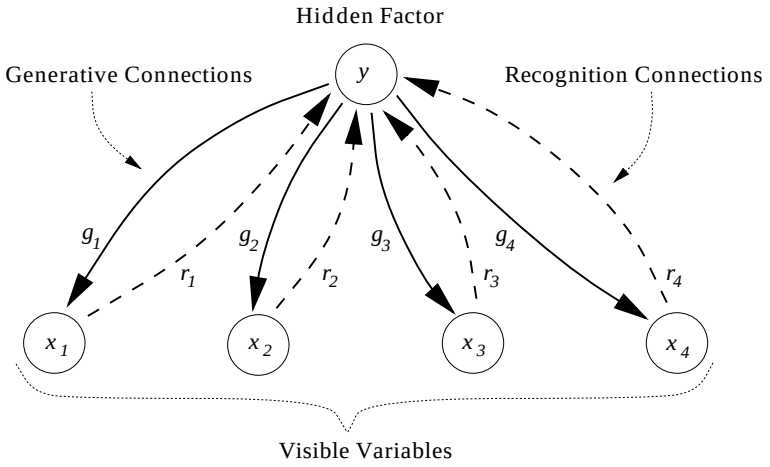


Figure 1: The one-factor linear Helmholtz machine. Connections of the generative model are shown using solid lines, those of the recognition model using dashed lines. The weights for these connections are given by the g_j and the r_j.

in Figure 1. The connections for the linear network that implements the generative model of equation 2.1 are shown in the figure using solid lines. Generation of data using this network starts with the selection of a random value for the hidden factor, y, from a gaussian distribution with mean zero and variance one. The value for the jth visible variable, x_j, is then found by multiplying y by a connection weight, g_j, and adding random noise with variance τ_j^2. (A bias value, μ_j, could be added as well, to produce nonzero means for the visible variables.)

The recognition model's connections are shown in Figure 1 using dashed lines. These connections implement equation 2.3. When presented with values for the visible input variables, x_j, the recognition network produces a value for the hidden factor, y, by forming a weighted sum of the inputs, with the weight for the jth input being r_j, and then adding gaussian noise with mean zero and variance σ^2. (If the inputs have nonzero means, the recognition model would include a bias for the hidden factor as well.)

The parameters of the generative model are learned in the wake phase, as follows. Values for the visible variables, $\mathbf{x}^{(c)}$, are obtained from the external world; they are set to a training case drawn from the distribution that we wish to model. The current version of the recognition network is then used to stochastically fill in a corresponding value for the hidden factor, $y^{(c)}$, using equation 2.3. The generative weights are then updated using the delta rule,

as follows:

$$g_j' = g_j + \eta\,(x_j^{(c)} - g_j y^{(c)})\,y^{(c)}, \tag{3.4}$$

where η is a small positive learning rate parameter. The generative variances, τ_j^2, which make up Ψ, are also updated, using an exponentially weighted moving average:

$$(\tau_j^2)' = \alpha\,\tau_j^2 \;+\; (1-\alpha)\,(x_j^{(c)} - g_j y^{(c)})^2, \tag{3.5}$$

where α is a learning rate parameter that is slightly less than one.

If the recognition model correctly inverted the generative model, these wake-phase updates would correctly implement the M-step of the EM algorithm (in the limit as $\eta \to 0$ and $\alpha \to 1$). The update for g_j in equation 3.4 improves the expected log likelihood because the increment to g_j is proportional to the derivative of the log likelihood for the training case with y filled in, which is as follows:

$$\frac{\partial}{\partial g_j}\left(\log p(\mathbf{x}^{(c)},\, y^{(c)} \mid \mathbf{g},\, \Psi)\right) = \frac{\partial}{\partial g_j}\left(-\frac{1}{2\tau_j^2}\,(x_j^{(c)} \;-\; g_j y^{(c)})^2\right) \tag{3.6}$$

$$= \frac{1}{\tau_j^2}\,(x_j^{(c)} \;-\; g_j y^{(c)})\,y^{(c)}. \tag{3.7}$$

The averaging operation by which the generative variances, τ_j^2, are learned will (for α close to one) also lead toward the maximum likelihood values based on the filled-in values for y.

The parameters of the recognition model are learned in the sleep phase, based not on real data but on "fantasy" cases, produced using the current version of the generative model. Values for the hidden factor, $y^{(f)}$, and the visible variables, $\mathbf{x}^{(f)}$, in a fantasy case are stochastically generated according to equation 2.1, as described at the beginning of this section. The connection weights for the recognition model are then updated using the delta rule, as follows:

$$r_j' = r_j + \eta\,(y^{(f)} - \mathbf{r}^T\mathbf{x}^{(f)})\,x_j^{(f)}, \tag{3.8}$$

where η is again a small positive learning rate parameter, which might or might not be the same as that used in the wake phase. The recognition variance, σ^2, is updated as follows:

$$(\sigma^2)' = \alpha\,\sigma^2 \;+\; (1-\alpha)\,(y^{(f)} - \mathbf{r}^T\mathbf{x}^{(f)})^2, \tag{3.9}$$

where α is again a learning rate parameter slightly less than one.

These updates are analogous to those made in the wake phase and have the effect of improving the recognition model's ability to produce the correct distribution for the hidden factor. However, as noted in section 3.2, the criterion by which the sleep phase updates "improve" the recognition model does not correspond to a global Lyapunov function for wake-sleep learning as a whole, and we therefore lack a theoretical guarantee that the wake-sleep procedure will be stable and will converge to a maximum of the likelihood, though the experiments in section 4 indicate that it usually does.

3.4 Wake-Sleep Learning for Multiple-Factor Models. A Helmholtz machine with more than one hidden factor can be trained using the wake-sleep algorithm in much the same way as described above for a single-factor model. The generative model is simply extended by including more than one hidden factor. In the most straightforward case, the values of these factors are chosen independently at random when a fantasy case is generated. However, a new issue arises with the recognition model. When there are k hidden factors and p visible variables, the recognition model has the following form (assuming zero means):

$$\mathbf{y} = \mathbf{R}\mathbf{x} + \boldsymbol{\nu} \tag{3.10}$$

where the $k \times p$ matrix $\mathbf{R}$ contains the weights on the recognition connections, and $\boldsymbol{\nu}$ is a k-dimensional gaussian random vector with mean $\mathbf{0}$ and covariance matrix Σ, which in general (i.e., for some arbitrary generative model) will not be diagonal. Generation of a random $\boldsymbol{\nu}$ with such covariance can easily be done on a computer using the Cholesky decomposition of Σ, but in a method intended for consideration as a neurobiological model, we would prefer a local implementation that produces the same effect.

One way of producing arbitrary covariances between the hidden factors is to include a set of connections in the recognition model that link each hidden factor to hidden factors that come earlier (in some arbitrary ordering). A Helmholtz machine with this architecture is shown in Figure 2. During the wake phase, values for the hidden factors are filled in sequentially by random generation from gaussian distributions. The mean of the distribution used in picking a value for factor y_i is the weighted sum of inputs along connections from the visible variables and from the hidden factors whose values were chosen earlier. The variance of the distribution for factor y_i is an additional recognition model parameter, σ_i^2. The $k(k-1)/2$ connections between the hidden factors and the k variances associated with these factors together make up the $k(k+1)/2$ independent degrees of freedom in Σ. Accordingly, a recognition model of this form exists that can perfectly invert any generative model.[2]

[2] Another way of seeing this is to note that the joint distribution for the visible vari-

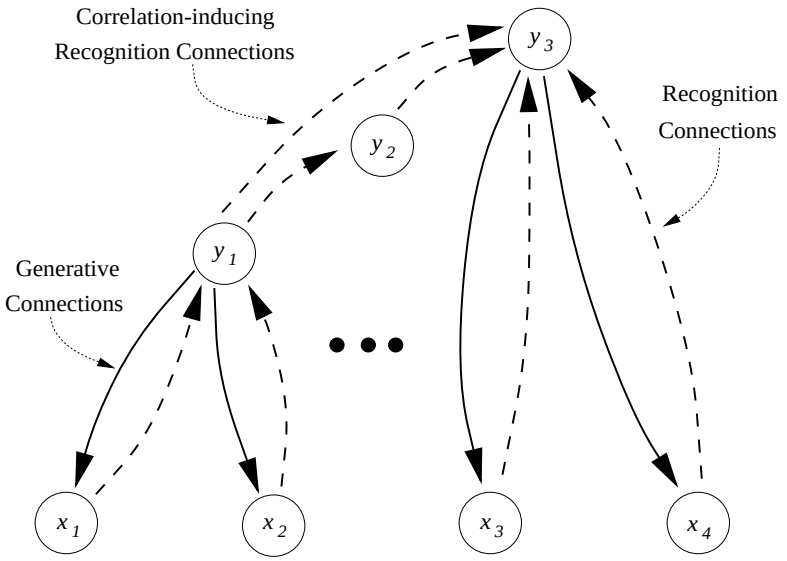


Figure 2: A Helmholtz machine implementing a model with three hidden factors, using a recognition model with a set of correlation-inducing connections. The figure omits some of the generative connections from hidden factors to visible variables and some of the recognition connections from visible variables to hidden factors (indicated by the ellipses). Note, however, that the only connections between hidden factors are those shown, which are part of the recognition model. There are no generative connections between hidden factors.

This method is reminiscent of some of the proposals to allow Hebbian learning to extract more than one principal component (e.g., Sanger, 1989; Plumbley, 1993); various of these also order the "hidden" units. In these proposals, the connections are used to remove correlations between the units so that they can represent different facets of the input. However, for the Helmholtz machine, the factors come to represent different facets of the input because they are jointly rather than separately engaged in capturing the statistical regularities in the input. The connections between the hidden units capture the correlations in the distribution for the hidden factors con-

ables and hidden factors is multivariate gaussian. From general properties of multivariate gaussians, the conditional distribution for a hidden factor given values for the visible variables and for the earlier hidden factors must also be gaussian, with some constant variance and with a mean given by some linear function of the other values.

ditional on given values for the visible variables that may be induced by this joint responsibility for modeling the visible variables.

Another approach is possible if the covariance matrix for the hidden factors, $\mathbf{y}$, in the generative model is rotationally symmetric, as is the case when it is the identity matrix, as we have assumed so far. We may then freely rotate the space of factors ($\mathbf{y}' = \mathbf{U}\mathbf{y}$, where $\mathbf{U}$ is a unitary matrix), making corresponding changes to the generative weights ($\mathbf{G}' = \mathbf{G}\mathbf{U}^T$), without changing the distribution of the visible variables. When the generative model is rotated, the corresponding recognition model will also rotate. There always exist rotations in which the recognition covariance matrix, Σ, is diagonal and can therefore be represented by just k variances, σ_i^2, one for each factor. This amounts to forcing the factors to be independent, or factorial, which has itself long been suggested as a goal for early cortical processing (Barlow, 1989) and has generally been assumed in nonlinear versions of the Helmholtz machine. We can therefore hope to learn multiple-factor models using wake-sleep learning in exactly the same way as we learn single-factor models, with equations 2.4 and 3.10 specifying how the values of all the factors $\mathbf{y}$ combine to predict the input $\mathbf{x}$, and vice versa. As seen in the next section, such a Helmholtz machine with only the capacity to represent k recognition variances, with no correlation-inducing connections, is usually able to find a rotation in which such a recognition model is sufficient to invert the generative model.

Note that there is no counterpart in Hebbian learning of this second approach, in which there are no connections between hidden factors. If Hebbian units are not connected in some manner, they will all extract the same single principal component of the inputs.

4 Empirical Results

We have run a number of experiments on synthetic data in order to test whether the wake-sleep algorithm applied to factor analysis finds parameter values that at least locally maximize the likelihood, in the limit of small values for the learning rates. These experiments also provide data on how small the learning rates must be in practice and reveal situations in which learning (particularly of the uniquenesses) can be relatively slow. We also report in section 4.4 the results of applying the wake-sleep algorithm to a real data set used by Everitt (1984). Away from conditions in which the maximum likelihood model is not well specified, the wake-sleep algorithm performs quite competently.

4.1 Experimental Procedure. All the systematic experiments described were done with randomly generated synthetic data. Models for various numbers (p) of visible variables, using various numbers (k) of hidden factors, were tested. For each such model, 10 sets of model parameters were generated, each of which was used to generate 2 sets of training data, the

first with 10 cases and the second with 500 cases. Models with the same p and k were then learned from these training sets using the wake-sleep algorithm.

The models used to generate the data were randomly constructed as follows. First, initial values for generative variances (the "uniquenesses") were drawn independently from exponential distributions with mean one, and initial values for the generative weights (the "factor loadings") were drawn independently from gaussian distributions with mean zero and variance one. These parameters were then rescaled so as to produce a variance of one for each visible variable; that is, for a single-factor model, the new generative variances were set to $\tau_j'^2 = f_j^2 \tau_j^2$ and the new generative weights to $g_j' = f_j g_j$, with the f_j chosen such that the new variances for the visible variables, $\tau_j'^2 + g_j'^2$, were equal to one.

In all experiments, the wake-sleep learning procedure was started with the generative and recognition variances set to one and the generative and recognition weights and biases set to zero. Symmetry is broken by the stochastic nature of the learning procedure. Learning was done online, with cases from the training set being presented one at a time, in a fixed sequence. The learning rates for both generative and recognition models were usually set to $\eta = 0.0002$ and $\alpha = 0.999$, but other values of η and α were investigated as well, as described below. In the models used to generate the data, the variables all had mean zero and variance one, but this knowledge was not built into the learning procedure. Bias parameters were therefore present, allowing the network to learn the means of the visible variables in the training data (which were close to zero, but not exactly zero, due to the use of a finite training set).

We compared the estimates produced by the wake-sleep algorithm with the maximum likelihood estimates based on the same training sets produced using the "factanal" function in S-Plus (Version 3.3, Release 1), which uses a modification of Jöreskog's (1977) method. We also implemented the EM factor analysis algorithm to examine in more detail cases in which wake-sleep disagreed with S-Plus. As with most stochastic learning algorithms, using fixed learning rates implies that convergence can at best be to a distribution of parameter values that is highly concentrated near some stable point. One would in general have to reduce the learning rates over time to ensure stronger forms of convergence. The software used in these experiments may be obtained over the Internet.[3]

4.2 Experiments with Single-Factor Models. We first tried using the wake-sleep algorithm to learn single-factor models for three ($p = 3$) and six ($p = 6$) visible variables. Figure 3 shows the progress of learning for a

[3] Follow the links from the first author's home page: `http://www.cs.utoronto.ca/~radford/`

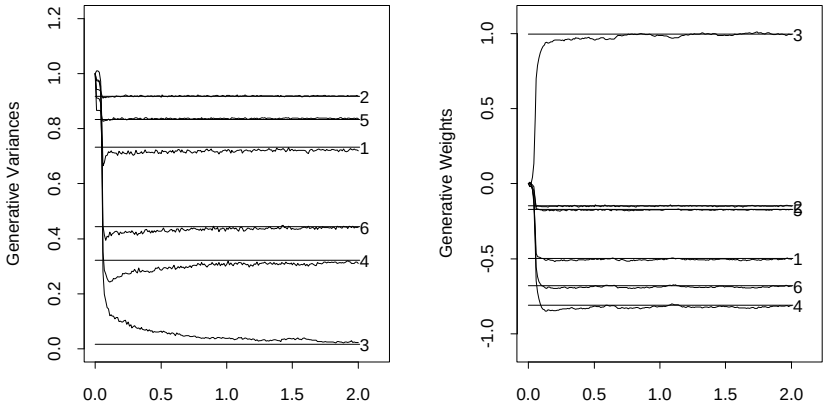


Figure 3: Wake-sleep learning of a single-factor model for six variables. The graphs show the progress of the generative variances and weights over the course of 2 million presentations of input vectors, drawn sequentially from a training set of size 500, with learning parameters of $\eta = 0.0002$ and $\alpha = 0.999$ being used for both the wake and the sleep phases. The training set was randomly generated from a single-factor model whose parameters were picked at random, as described in section 4.1. The maximum likelihood parameter estimates found by S-Plus are shown as horizontal lines.

typical run with $p = 6$, applied to a training set of size 500. The figure shows progress over 2 million presentations of input vectors (4000 presentations of each of the 500 training cases). Both the generative variances and the generative weights are seen to converge to the maximum likelihood estimates found by S-Plus, with a small amount of random variation, as is expected with a stochastic learning procedure. Of the 20 runs with $p = 6$ (based on data generated from 10 random models, with training sets of size 10 and 500), all but one showed similar convergence to the S-Plus estimates within 3 million presentations (and usually much earlier). The remaining run (on a training set of size 10) converged to a different local maximum, which S-Plus found when its maximization routine was started from the values found by wake-sleep.

Convergence to maximum likelihood estimates was sometimes slower when there were only three variables ($p = 3$). This is the minimum number of visible variables for which the single-factor model is identifiable (that is, for which the true values of the parameters can be found given enough data, apart from an ambiguity in the overall signs of the weights). Three of the 10 runs with training sets of size 10 and one of the 10 runs with training sets of size 500 failed to converge clearly within 3 million presentations,

but convergence was seen when these runs were extended to 10 million presentations (in one case to a different local maximum than initially found by S-Plus). The slowest convergence was for a training set of size 10 for which the maximum likelihood estimate for one of the generative weights was close to zero (0.038), making the parameters nearly unidentifiable.

All the runs were done with learning rates of $\eta = 0.0002$ and $\alpha = 0.999$, for both the generative and recognition models. Tests on the training sets with $p = 3$ were also done with learning for the recognition model slowed to $\eta = 0.00002$ and $\alpha = 0.9999$—a situation that one might speculate would cause problems, due to the recognition model's not being able to keep up with the generative model. No problems were seen (though the learning was, of course, slower).

4.3 Experiments with Multiple-Factor Models. We have also tried using the wake-sleep algorithm to learn models with two and three hidden factors ($k = 2$ and $k = 3$) from synthetic data generated as described in section 4.1. Systematic experiments were done for $k = 2$, $p = 5$, using models with and without correlation-inducing recognition weights, and for $k = 2$, $p = 8$ and $k = 3$, $p = 9$, in both cases using models with no correlation-inducing recognition weights. For most of the experiments, learning was done with $\eta = 0.0002$ and $\alpha = 0.999$.

The behavior of wake-sleep learning with $k = 2$ and $p = 5$ was very similar for the model with correlation-inducing recognition connections and the model without such connections. The runs on most of the 20 data sets converged within the 6 million presentations that were initially performed; a few required more iterations before convergence was apparent. For two data sets (both of size 10), wake-sleep learning converged to different local maxima than S-Plus. For one training set of size 500, the maximum likelihood estimate for one of the generative variances is very close to one, making the model almost unidentifiable ($p = 5$ being the minimum number of visible variables for identifiability with $k = 2$); this produced an understandable difficulty with convergence, though the wake-sleep estimates still agreed fairly closely with the maximum likelihood values found by S-Plus.

In contrast with these good results, a worrying discrepancy arose with one of the training sets of size 10, for which the two smallest generative variances found using wake-sleep learning differed somewhat from the maximum likelihood estimates found by S-Plus (one of which was very close to zero). When S-Plus was started at the values found using wake-sleep, it did *not* find a similar local maximum; it simply found the same estimates as it had found with its default initial values. However, when the full EM algorithm was started at the estimates found by wake-sleep, it barely changed the parameters for many iterations. One explanation could be that there is a local maximum in this vicinity, but it is for some reason not found by S-Plus. Another possible explanation could be that the likelihood is extremely flat in this region. In neither case would the discrepancy be cause

for much worry regarding the general ability of the wake-sleep method to learn these multiple-factor models. However, it is also possible that this is an instance of the problem that arose more clearly in connection with the Everitt crime data, as reported in section 4.4.

Runs using models without correlation-inducing recognition connections were also performed for data sets with $k = 2$ and $p = 8$. All runs converged, most within 6 million presentations, in one case to a different local maximum than S-Plus. We also tried runs with the same model and data sets using higher learning rates ($\eta = 0.002$ and $\alpha = 0.99$). The higher learning rates produced both higher variability and some bias in the parameter estimates, but for most data sets the results were still generally correct.

Finally, runs using models without correlation-inducing recognition connections were performed for data sets with $k = 3$ and $p = 9$. Most of these converged fine. However, for two data sets (both of size 500), a small but apparently real difference was seen between the estimates for the two smallest generative variances found using wake-sleep and the maximum likelihood estimates found using S-Plus. As was the case with the similar situation with $k = 2$, $p = 5$, S-Plus did not converge to a local maximum in this vicinity when started at the wake-sleep estimates. As before, however, the EM algorithm moved very slowly when started at the wake-sleep estimates, which is one possible explanation for wake-sleep having apparently converged to this point. However, it is also possible that something more fundamental prevented convergence to a local maximum of the likelihood, as discussed in connection with similar results in the next section.

4.4 Experiments with the Everitt Crime Data. We also tried learning a two-factor model for a data set used as an example by Everitt (1984), in which the visible variables are the rates for seven types of crime, with the cases being 16 American cities. The same learning procedure (with $\eta = 0.0002$ and $\alpha = 0.999$) was used as in the experiments above, except that the visible variables were normalized to have mean zero and variance one, and bias parameters were accordingly omitted from both the generative and recognition models.

Fifteen runs with different random number seeds were done, for both models with a correlation-inducing recognition connection between the two factors and without such a connection. All of these runs produced results fairly close to the maximum likelihood estimates found by S-Plus (which match the results of Everitt). In some runs, however, there were small, but clearly real, discrepancies, most notably in the smallest two generative variances, for which the wake-sleep estimates were sometimes nearly zero, whereas the maximum likelihood estimate is zero for only one of them. This behavior is similar to that seen in the three runs where discrepancies were found in the systematic experiments of section 4.3.

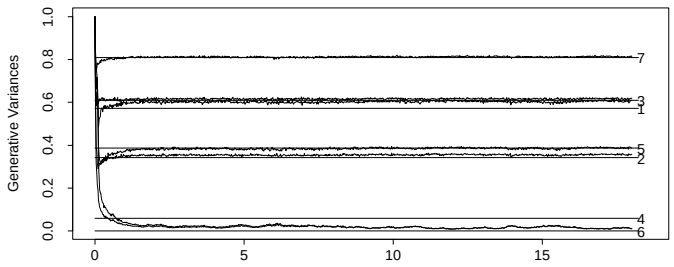
These discrepancies arose much more frequently when a correlation-inducing recognition connection was not present (14 out of 15 runs) than

when such a connection was included (6 out of 15 runs). Figure 4a shows one of the runs with discrepancies for a model with a correlation-inducing connection present. Figure 4b shows a run differing from that of 4a only in its random seed, but which did find the maximum likelihood estimates. The sole run in which the model without a correlation-inducing recognition connection converged to the maximum likelihood estimates is shown in Figure 4c. Close comparison of Figures 4b and 4c shows that even in this run, convergence was much more labored without the correlation-inducing connection. (In particular, the generative variance for variable 6 approaches zero rather more slowly.)

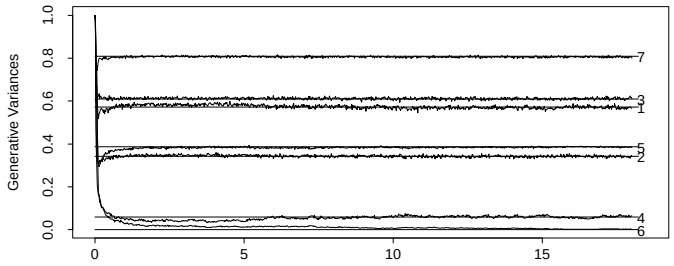
According to S-Plus, the solution found in the discrepant runs is *not* an alternative local maximum. Furthermore, extending the runs to many more iterations or reducing the learning rates by a large factor does not eliminate the problem. This makes it seem unlikely that the problem is merely slow convergence due to the likelihood of being nearly flat in this vicinity, although, on the other hand, when EM is started at the discrepant wake-sleep estimates, its movement toward the maximum likelihood estimates is rather slow (after 200 iterations, the estimate for the generative variance for variable 4 had moved only about a third of the way from the wake-sleep estimate to the maximum likelihood estimate). It seems most likely therefore that the runs with discrepancies result from a local "basin of attraction" for wake-sleep learning that does not lead to a local maximum of the likelihood.

The discrepancies can be eliminated for the model with a correlation-inducing connection in either of two ways. One way is to reduce the learning rate for the generative parameters (in the wake phase), while leaving the recognition learning rate unchanged. As discussed in section 3.2, there is a theoretical reason to think that this method will lead to the maximum likelihood estimates, since the recognition model will then have time to learn how to invert the generative model perfectly. When the generative learning rates are reduced by setting $\eta = 0.00005$ and $\alpha = 0.99975$, the maximum likelihood estimates are indeed found in eight of eight test runs. The second solution is to impose a constraint that prevents the generative variances from falling below 0.01. This also worked in eight of eight runs. However, these two methods produce little or no improvement when the correlation-inducing connection is omitted from the recognition model (no successes in eight runs with smaller generative learning rates; three successes in eight runs with a constraint on the generative variances). Thus, although models without correlation-inducing connections often work well, it appears that learning is sometimes easier and more reliable when they are present.

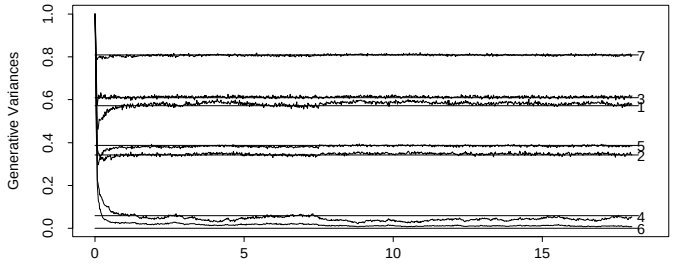
Finally, we tried learning a model for this data without first normalizing the visible variables to have mean zero and variance one, but with biases included in the generative and recognition models to handle the nonzero means. This is not a very sensible thing to do; the means of the variables in this data set are far from zero, so learning would at best take a long time, while the biases slowly adjusted. In fact, however, wake-sleep learning fails



(a)



(b)



(c)

Figure 4: Wake-sleep learning of two-factor models for the Everitt crime data. The horizontal axis shows the number of presentations in millions. The vertical axis shows the generative variances, with the maximum likelihood values indicated by horizontal lines. (a) One of the six runs in which a model with a correlation-inducing recognition connection did not converge to the maximum likelihood estimates. (b) One of the nine such runs that did find the maximum likelihood estimates. (c) The only run in which the maximum likelihood estimates were found using a model without a correlation-inducing connection.

rather spectacularly. The generative variances immediately become quite large. As soon as the recognition weights depart significantly from zero, the recognition variances also become quite large, at which point positive feedback ensues, and all the weights and variances diverge.

The instability that can occur once the generative and recognition weights and variances are too large appears to be a fundamental aspect of wake-sleep dynamics. Interestingly, however, it seems that the dynamics may operate to avoid this unstable region of parameter space, since the instability does not occur with the Everitt data if the learning rate is set very low. With a larger learning rate, however, the stochastic aspect of the learning can produce a jump into the unstable region.

5 Discussion

We have shown empirically that wake-sleep learning, which involves nothing more than two simple applications of the local delta rule, can be used to implement the statistical technique of maximum likelihood factor analysis. However, just as it is usually computationally more efficient to implement principal component analysis using a standard matrix technique such as singular-value decomposition rather than by using Hebbian learning, factor analysis is probably better implemented on a computer using either EM (Rubin & Thayer, 1982) or the second-order Newton methods of Jöreskog (1967, 1969, 1977) than by the wake-sleep algorithm. In our view, wake-sleep factor analysis is interesting as a simple and successful example of wake-sleep learning and as a possible model of activity-dependent plasticity in the cortex.

5.1 Implications for Wake-Sleep Learning. The experiments in section 4 show that, in most situations, wake-sleep learning applied to factor analysis leads to estimates that (locally) maximize the likelihood, when the learning rate is set to be small enough. In a few situations, the parameter estimates found using wake-sleep deviated slightly from the maximum likelihood values. More work is needed to determine exactly when and why this occurs, but even in these situations, the estimates found by wake-sleep are reasonably good and the likelihoods are close to their maxima. The sensitivity of learning to settings of parameters such as learning rates appears no greater than is typical for stochastic gradient descent methods. In particular, setting the learning rate for the generative model to be equal to or greater than that for the recognition model did not lead to instability, even though this is the situation in which one might worry that the algorithm would no longer mimic EM (as discussed in section 3.2). However, we have been unable to prove in general that the wake-sleep algorithm for factor analysis is guaranteed to converge to the maximum likelihood solution. Indeed, the empirical results show that any general theoretical proof of correctness would have to contain caveats regarding unstable regions of the parameter

space. Such a proof may be impossible if the discrepancies seen in the experiments are indeed due to a false basin of attraction (rather than being an effect of finite learning rates). Despite the lack so far of strong theoretical results, the relatively simple factor analysis model may yet provide a good starting point for a better theoretical understanding of wake-sleep learning for more complex models.

One empirical finding was that it is possible to learn multiple-factor models even when correlation-inducing recognition connections are absent, although the lack of such connections did cause difficulties in some cases. A better understanding of what is going on in this respect would provide insight into wake-sleep learning for more complex models, in which the recognition model will seldom be able to invert the generative model perfectly. Such a complex generative model may allow the data to be modeled in many nearly equivalent ways, for some of which the generative model is harder to invert than for others. Good performance may sometimes be possible only if wake-sleep learning favors the more easily inverted generative models. We have seen that this does usually happen for factor analysis.

5.2 Implications for Activity-Dependent Plasticity. Much progress has been made using Hebbian learning to model the development of structures in early cortical areas, such as topographic maps (Willshaw & von der Malsburg 1976, 1979; von der Malsburg, & Willshaw, 1977), ocular dominance stripes (Miller et al., 1989), and orientation domains (Linkser, 1988; Miller, 1994). However, Hebbian learning suffers from three problems that the equally-local wake-sleep algorithm avoids.

First, because of the positive feedback inherent in Hebbian learning, some form of synaptic normalization is required to make it work (Miller & MacKay, 1994). There is no evidence for synaptic normalization in the cortex. This is not an issue for Helmholtz machines because building a statistical model of the inputs involves negative feedback instead, as in the delta rule.

Second, in order for Hebbian learning to produce several outputs that represent more than just the first principal component of a collection of inputs, there must be connections of some sort between the output units, which force them to be decorrelated. Typically, some form of anti-Hebbian learning is required for these connections (see Sanger, 1989; Földiák, 1989; Plumbley, 1993). The common alternative of using fixed lateral connections between the output units is not informationally efficient. In the Helmholtz machine, the goal of modeling the distribution of the inputs forces output units to differentiate rather than perform the same task. This goal also supplies a clear interpretation in terms of finding the hidden causes of the input.

Third, Hebbian learning does not accord any role to the prominent cortical feature that top-down connections always accompany bottom-up connections. In the Helmholtz machine, these top-down connections play a

crucial role in wake-sleep learning. Furthermore, the top-down connections come to embody a hierarchical generative model of the inputs, some form of which appears necessary in any case to combine top-down and bottom-up processing during inference.

Of course, the Helmholtz machine suffers from various problems itself. Although learning rules equivalent to the delta rule are conventional in classical conditioning (Rescorla & Wagner, 1972; Sutton & Barto, 1981) and have also been suggested as underlying cortical plasticity (Montague & Sejnowski, 1994), it is not clear how cortical microcircuitry might construct the required predictions (such as $g_j y^{(c)}$ of equation 3.4) or prediction errors (such as $x_j^{(c)} - g_j y^{(c)}$ of the same equation). The wake-sleep algorithm also requires two phases of activation, with different connections being primarily responsible for driving the cells in each phase. Although there is some suggestive evidence of this (Hasselmo & Bower, 1993), it has not yet been demonstrated. There are also alternative developmental methods that construct top-down statistical models of their inputs using only one, slightly more complicated, phase of activation (Olshausen & Field, 1996; Rao & Ballard, 1995).

In Hebbian models of activity-dependent development, a key role is played by lateral local intracortical connections (longer-range excitatory connections are also present but develop later). Lateral connections are not present in the linear Helmholtz machines we have so far described, but they could play a role in inducing correlations between the hidden factors in the generative model, in the recognition model, or in both, perhaps replacing the correlation-inducing connections shown in Figure 2.

Purely linear models, such as those presented here, will not suffice to explain fully the intricacies of information processing in the cortical hierarchy. However, understanding how a factor analysis model can be learned using simple and local operations is a first step to understanding how more complex statistical models can be learned using more complex architectures involving nonlinear elements and multiple layers.

Acknowledgments

We thank Geoffrey Hinton for many useful discussions. This research was supported by the Natural Sciences and Engineering Research Council of Canada and by grant R29 MH55541-01 from the National Institute of Mental Health of the United States. All opinions expressed are those of the authors.

References

Barlow, H. B. (1989). Unsupervised learning. *Neural Computation, 1*, 295–311.

Dayan, P., & Hinton, G. E. (1996). Varieties of Helmholtz machine. *Neural Networks, 9*, 1385–1403.

Dayan, P., Hinton, G. E., Neal, R. M., & Zemel, R. S. (1995). The Helmholtz machine. *Neural Computation, 7*, 889–904.

Dempster, A. P., Laird, N. M, & Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Proceedings of the Royal Statistical Society, B39*, 1–38.

Everitt, B. S. (1984). *An introduction to latent variable models.* London: Chapman and Hall.

Felleman D. J., & Van Essen, D. C. (1991). Distributed hierarchical processing in the primate cerebral cortex. *Cerebral Cortex, 1*, 1–47.

Földiák, P. (1989). Adaptive network for optimal linear feature extraction. In *Proceedings of the International Joint Conference on Neural Networks, Washington, DC* (Vol. I, pp. 401–405).

Grenander, U. (1976–1981). *Lectures in pattern theory I, II and III: Pattern analysis, pattern synthesis and regular structures*. Berlin: Springer-Verlag.

Hasselmo, M. E., & Bower, J. M. (1993). Acetylcholine and memory. *Trends in Neurosciences, 16*, 218–222.

Hebb, D. O. (1949). *The organization of behavior: A neuropsychological theory.* New York: Wiley.

Hinton, G. E., Dayan, P., Frey, B., & Neal, R. M. (1995). The wake-sleep algorithm for self-organizing neural networks. *Science, 268*, 1158–1160.

Hinton, G. E., & Zemel, R. S. (1994). Autoencoders, minimum description length and Helmholtz free energy. In J. D. Cowan, G. Tesauro, & J. Alspector (Eds.), *Advances in neural information processing systems, 6* (pp. 3–10). San Mateo, CA: Morgan Kaufmann.

Jolliffe, I. T. (1986) *Principal component analysis*, New York: Springer-Verlag.

Jöreskog, K. G. (1967). Some contributions to maximum likelihood factor analysis, *Psychometrika, 32*, 443–482.

Jöreskog, K. G. (1969). A general approach to confirmatory maximum likelihood factor analysis, *Psychometrika, 34*, 183–202.

Jöreskog, K. G. (1977). Factor analysis by least-squares and maximum-likelihood methods. In K. Enslein, A. Ralston, & H. S. Wilf (Eds.), *Statistical methods for digital computers*. New York: Wiley.

Linsker, R. (1986). From basic network principles to neural architecture. *Proceedings of the National Academy of Sciences, 83*, 7508–7512, 8390–8394, 8779–9783.

Linsker, R. (1988). Self-organization in a perceptual network. *Computer, 21*, 105–128.

Miller, K. D. (1994). A model for the development of simple cell receptive fields and the ordered arrangement of orientation columns through activity-dependent competition between ON- and OFF-center inputs. *Journal of Neuroscience, 14*, 409–441.

Miller, K. D., Keller, J. B., & Stryker, M. P. (1989). Ocular dominance column development: Analysis and simulation. *Science, 245*, 605–615.

Miller, K. D., & MacKay, D. J. C. (1994). The role of constraints in Hebbian learning. *Neural Computation, 6*, 100–126.

Montague, P. R., & Sejnowski, T. J. (1994). The predictive brain: Temporal coincidence and temporal order in synaptic learning mechanisms. *Learning and Memory, 1*, 1–33.

Mumford, D. (1994). Neuronal architectures for pattern-theoretic problems. In C. Koch & J. Davis (Eds.), *Large-scale theories of the cortex* (pp. 125–152). Cambridge, MA: MIT Press.

Oja, E. (1989). Neural networks, principal components, and subspaces. *International Journal of Neural Systems, 1*, 61–68.

Olshausen, B. A., & Field, D. J. (1996). Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature, 381*, 607–609.

Plumbley, M. D. (1993). Efficient information transfer and anti-Hebbian neural networks. *Neural Networks, 6*, 823–833.

Rao, P. N. R., & Ballard, D. H. (1995). *Dynamic model of visual memory predicts neural response properties in the visual cortex* (Technical Rep. No. 95.4). Rochester, NY: Department of Computer Science, University of Rochester.

Rescorla, R. A., & Wagner, A. R. (1972). A theory of Pavlovian conditioning: The effectiveness of reinforcement and non-reinforcement. In A. H. Black & W. F. Prokasy (Eds.), *Classical conditioning II: Current research and theory* (pp. 64–69). New York: Appleton-Century-Crofts.

Rubin, D. B., & Thayer, D. T. (1982). EM algorithms for ML factor analysis. *Psychometrika, 47*, 69–76.

Sanger, T. D. (1989). Optimal unsupervised learning in a single-layer linear feedforward neural network. *Neural Networks, 2*, 459–473.

Sutton, R. S., & Barto, A. G. (1981). Toward a modern theory of adaptive networks: Expectation and prediction. *Psychological Review, 88*, 135–170.

von der Malsburg, C. (1973). Self-organization of orientation sensitive cells in the striate cortex. *Kybernetic, 14*, 85–100.

von der Malsburg, C., & Willshaw D. J. (1977). How to label nerve cells so that they can interconnect in an ordered fashion. *Proceedings of the National Academy of Sciences, 74*, 5176–5178.

Willshaw, D. J., & von der Malsburg, C. (1976). How patterned neural connections can be set up by self-organisation. *Proceedings of the Royal Society of London B, 194*, 431–445.

Willshaw, D. J., & von der Malsburg, C. (1979). A marker induction mechanism for the establishment of ordered neural mappings: Its application to the retinotectal problem. *Philosophical Transactions of the Royal Society B, 287*, 203–243.

Received August 7, 1996; accepted April 3, 1997.