Technical Report No. 9607, Department of Statistics, University of Toronto

# Factor Analysis Using Delta-Rule Wake-Sleep Learning

Radford M. Neal
Department of Statistics and Department of Computer Science
University of Toronto
radford@stat.utoronto.ca

Peter Dayan
Department of Brain and Cognitive Sciences
Massachusetts Institute of Technology
dayan@ai.mit.edu

24 July 1996

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 Oja's version of 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

Activity-dependent plasticity in the vertebrate brain has typically been modeled in terms of Hebbian learning (Hebb 1959), in which weight changes are based on the covariance of pre-synaptic and post-synaptic activity (eg, von der Malsburg 1973; Linsker 1986; Miller, Keller, and Stryker 1989). These models derive support from neurobiological evidence of long-term potentiation (see, for example, Collingridge and Bliss (1987), and for a recent review, Baudry and Davis (1994)). They have also been seen as performing a reasonable function, namely extracting the statistical structure amongst a collection of inputs in terms of principal components (Linkser 1988). In this paper, we suggest the statistical technique of factor analysis as an interesting alternative to principal components analysis, and show how to implement it using an algorithm whose demands on synaptic plasticity are as local as those of the Hebb rule.

Factor analysis is a model for real-valued data in which correlations are "explained" by postulating the presence of one or more underlying "factors". These factors play the role of "latent" or "hidden" variables, which are not directly observable, but which allow the dependencies between the "visible" variables to be expressed in a convenient way. Everitt (1984) gives a good introduction to latent variable models in general, and to factor analysis in particular. These models are widely used 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. Our interest in these models stems from their potential as a way of building high-level representations from sensory data.

Oja's version of Hebbian learning (Oja and Karhunen 1985; Oja 1989, 1992) is a particularly convenient counterpoint. This rule applies to a linear unit with weight vector $\mathbf{w}$ that computes an output $y = \mathbf{w}^T\mathbf{x}$ when presented with a real-valued input vector $\mathbf{x}$ (which, for convenience, is assumed to have mean zero). After each presentation of an input vector, the weights for the unit are changed by an amount given by the following proportionality:

$$\Delta\mathbf{w} \;\; \propto \;\; y\,(\mathbf{x} - y\mathbf{w}) \;\; = \;\; y\mathbf{x} \; - \; y^2\mathbf{w}. \qquad (1)$$

The first term in this weight increment, $y\mathbf{x}$, is of Hebbian form. The second term, $-y^2\mathbf{w}$, tends to push the weights towards zero, balancing the positive feedback in plain Hebbian learning, which would otherwise increase the magnitude of the weights without bound. Wyatt and Elfadel (1995) give an explicit analysis of learning based on equation (1), showing that with reasonable starting conditions, $\mathbf{w}$ converges to the principal eigenvector of the covariance matrix of the inputs — that is, it converges to a unit vector pointing in the direction of highest variance in the input space. Extracting the subsidiary eigenvectors of the covariance matrix of the inputs is somewhat more challenging, requiring some form of inhibition between successive output units (Sanger 1989; Földiák 1989; Plumbley 1993).

Linsker (1988) views Hebbian learning as a way of maximising the information retained by $y$ about $\mathbf{x}$. Under the simplifying assumption that the distribution of the inputs is Gaussian, setting the output of a unit to the projection of its input onto the first principal component of the input covariance matrix conveys as much information as possible on average (see also Plumbley 1993). This goal seems reasonable for the very early stages of sensory processing, where information bottlenecks such as the optic nerve may plausibly be present. Note, however, that it implicitly assumes that all information is equally important. Maximizing information transfer seems less compelling as a goal for subsequent levels of processing, once sensory signals have reached cortex. Several other computational goals have been suggested from this stage upwards, including factorial coding (Barlow 1989), sparsification (Olshausen and Field 1995), and various methods for encouraging the cortex to respect reasonable invariances, such as translation or scale invariance for visual processing (Li and Atick 1994).

In this paper, we pursue the suggestion of Hinton and Zemel (1994) (see also Grenander 1976-1981; Mumford 1994; Dayan, Hinton, Neal, and Zemel 1995) that the cortex might be constructing a hierarchical stochastic "generative" model of its input in the top-down connections, while implementing in the bottom-up connections a "recognition" model that in a sense is the inverse of the generative model. The recognition model provides high-level interpretations of sensory data that can serve as a basis for behavior, and also plays a role in the learning of the generative model. If this suggestion is correct, the activity in successive cortical areas would represent the input in terms of a hierarchy of latent variables, which might be interpretable as encoding the *hidden causes* of the observed input. The purpose of this code would not be to preserve as much raw information as possible about the input, but rather to support a probabilistic model for the inputs that matches as closely as possible their actual distribution. We refer to such a combination of generative and recognition models as a "Helmholtz machine". One attractive method of learning such models is the "wake-sleep" algorithm of Hinton, Dayan, Frey, and Neal (1995).

The simplest form of Helmholtz machine has just two layers, linear units, and Gaussian noise models. Such a simple Helmholtz machine is equivalent to a factor analysis model. We show empirically that these models can be learned using the wake-sleep algorithm. The "wake" and "sleep" phases of this learning procedure both use the simple and local delta rule. The wake-sleep algorithm has previously been applied to the more difficult task of learning non-linear 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. One reason for this is that, unlike the typical situation with binary latent variables, the recognition model used with factor analysis can perfectly invert the generative model.

In the next section, we describe the factor analysis model. Section 3 explains the wake-sleep algorithm and its roots in the expectation-maximisation (EM) algorithm (Dempster, Laird, and Rubin 1977). Section 4 presents experimental evidence that factor analysis models can indeed be learned by the wake-sleep method, though there appear to be some situations in which the algorithm does not operate perfectly. Finally, we discuss the implications of this work in Section 5.

## 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, $\mathrm{x}$, that is given by

$$\mathrm{x} \;=\; \boldsymbol{\mu} + \mathrm{g}y + \boldsymbol{\epsilon} \tag{2}$$

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", $\mathrm{g}$, which we refer to as the generative weights, expresses the relationships of each visible variable to the hidden factor. The vector of overall means, $\boldsymbol{\mu}$, will, for simplicity of presentation, be taken to be zero in this paper, unless otherwise stated. Finally, the noise, $\boldsymbol{\epsilon}$, is assumed to be Gaussian, with a diagonal covariance matrix, $\Psi$, which we will write as

$$\Psi \;=\; \begin{pmatrix} \tau_1^2 & 0 & \dots & 0 \\ 0 & \tau_2^2 & \dots & 0 \\ & & \dots & \\ 0 & 0 & \dots & \tau_n^2 \end{pmatrix} \tag{3}$$

The $\tau_j^2$ 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 $\Psi$, define a joint 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 of 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

$$y \;=\; \mathbf{r}^T\mathbf{x} + \nu \tag{4}$$

where $\mathbf{r}$ is the vector of recognition weights, and $\nu$ has a Gaussian distribution with mean zero and some variance, $\sigma^2$.

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 5.3). These factors jointly produce a distribution for the visible variables, as follows:

$$\mathbf{x} \;=\; \boldsymbol{\mu} + \mathbf{G}\mathbf{y} + \boldsymbol{\epsilon} \tag{5}$$

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, will produce the same distribution for $\mathbf{x}$, using hidden factors given by $\mathbf{y}' = \mathbf{U}\mathbf{y}$. The presence of such multiple solutions causes difficulties in interpreting the results of factor analysis in statistical applications, but is probably not a problem in neurobiological modeling.

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 models any noise that affects only component $j$ by means of $\epsilon_j$. A large amount of noise simply results in the corresponding $\tau_j^2$ being large, with no effect on the output $y$. Furthermore, principal component analysis is essentially 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 (3) that it is in this coordinate system that the components of the noise are independent.

## 3 Factor analysis 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 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

Many methods for performing factor analysis have been proposed since the origins of the technique early in this century. In the 1960's, computationally feasible algorithms were developed for performing factor analysis by the statistically attractive method of maximum likelihood (Jöreskog 1967, 1969, 1977). In maximum likelihood learning, the parameters of the model are chosen so as 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:

$$L \;=\; L(\mathbf{g}, \Psi) \;=\; \prod_{c=1}^{n} p(\mathbf{x}^{(c)} \mid \mathbf{g},\, \Psi) \tag{6}$$

$$= \prod_{c=1}^{n} \int p(\mathbf{x}^{(c)},\, y^{(c)} \mid \mathbf{g},\, \Psi)\, dy^{(c)} \tag{7}$$

$$= \prod_{c=1}^{n} \int p(\mathbf{x}^{(c)} \mid y^{(c)},\, \mathbf{g},\, \Psi)\, p(y^{(c)})\, dy^{(c)} \tag{8}$$

where g and $\Psi$ are the model parameters, as described previously, and $p(\cdot)$ is used to write probability densities and conditional probability densities. In the last equation above, the 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 $\Psi$, as implied by equation (2).

Wake-sleep learning can be viewed as an approximate implementation of the Expectation-Maximization (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.* 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 minima given the same starting values for the parameters).

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

$$p(y^{(c)} \mid \mathbf{x}^{(c)}, \mathbf{g}, \Psi) \;=\; \frac{p(y^{(c)}, \mathbf{x}^{(c)} \mid \mathbf{g}, \Psi)}{\int p(\mathbf{x}^{(c)} \mid y, \mathbf{g}, \Psi)\, p(y)\, dy} \tag{9}$$

In the M-step, one finds new parameters values, $\mathbf{g}'$ and $\Psi'$, that maximise the expected log-likelihood of the complete data, with the unobserved variables filled in according to the conditional distribution found in the E-step:

$$\sum_{c=1}^{n} \int p(y^{(c)} \mid \mathbf{x}^{(c)}, \mathbf{g}, \Psi) \log \Big[ p(\mathbf{x}^{(c)} \mid y^{(c)}, \mathbf{g}', \Psi')\, p(y^{(c)}) \Big] \, dy^{(c)} \tag{10}$$

For factor analysis, the conditional distribution for $y^{(c)}$ in equation (9) 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 operations involved do not appear particularly plausible as a model of learning in the brain.

### 3.2 The wake-sleep approach

To obtain a learning procedure of greater potential neurobiological interest, we first eliminate the explicit maximization in the M-step of a quantity defined in terms of an expectation, replacing it 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 operate as follows:

**Wake phase:** From observed values for the visible variables, $\mathbf{x}$, randomly fill in values for the hidden factors, $\mathbf{y}$, 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, $\mathbf{y}$, from their fixed generative distribution, and then randomly choose "fantasy" values for the visible variables, $\mathbf{x}$, from their conditional distribution given $\mathbf{y}$, as defined by the current parameters of the generative model. Update the parameters of the recognition model to make this fantasy case more likely.

We now work out how this general scheme can be applied to factor analysis, first for a single-factor model, and then for models with multiple factors.

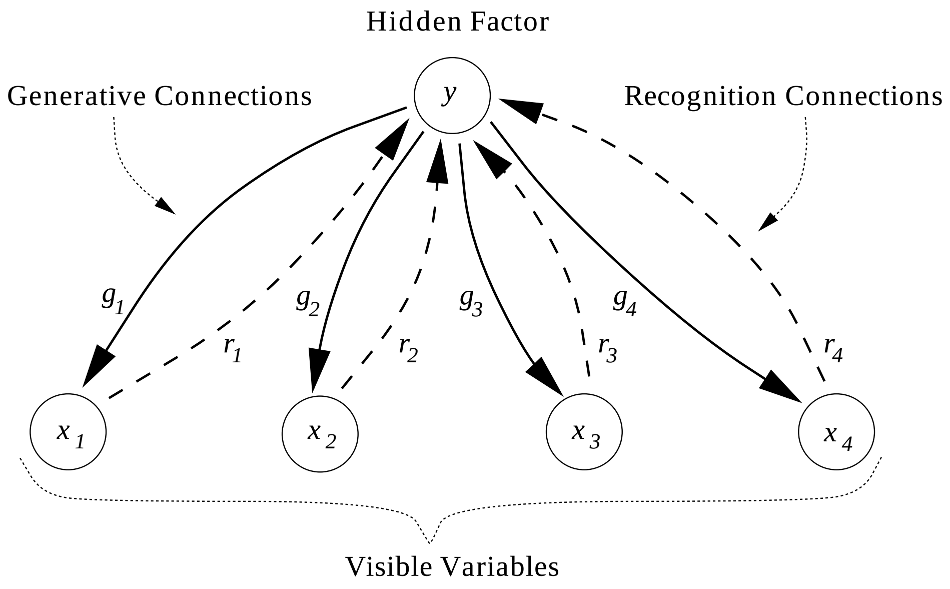

Figure 1: The one-factor linear Helmholtz machine. Connections of the generative model are shown using solid lines, those of the recognition model using dashed lines. The weights for these connections are given by the $g_j$ and the $r_j$.

## 3.3 Wake-sleep learning for a single-factor model

A Helmholtz Machine that implements factor analysis with a single hidden factor is shown in Figure 1. The connections for the linear network that implements the generative model of equation (2) 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 $j$'th visible variable, $x_j$, is then found by multiplying $y$ by a connection weight, $g_j$, and adding random noise with variance $\tau_j^2$. (A bias value, $\mu_j$, could be added as well, to produce non-zero means for the visible variables.)

The recognition model's connections are shown in Figure 1 using dashed lines. These connections implement equation (4). 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 $j$'th input being $r_j$, and then adding Gaussian noise with mean zero and variance $\sigma^2$. (If the inputs have non-zero 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 — ie, they are set to a "training case" drawn from the distribution which 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 (4). The generative weights are then updated using the delta rule, as follows:

$$g_j' \;=\; g_j \;+\; \eta\,(x_j^{(c)} - g_j y^{(c)})\, y^{(c)} \tag{11}$$

where $\eta$ is a small positive learning rate parameter. The generative variances, $\tau_j^2$, which make up $\Psi$, are also updated, using an exponentially-weighted moving average:

$$(\tau_j^2)' \;=\; \alpha\,\tau_j^2 \;+\; (1-\alpha)\,(x_j^{(c)} - g_j y^{(c)})^2 \tag{12}$$

where $\alpha$ 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 (11) improves the 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:

$$\frac{\partial}{\partial g_j}\Big(\log p(\mathbf{x}^{(c)},\, y^{(c)} \,|\, \mathbf{g},\, \Psi)\Big) \;=\; \frac{\partial}{\partial g_j}\left(-\frac{1}{2\tau_j^2}\,(x_j^{(c)} \;-\; g_j y^{(c)})^2\right) \tag{13}$$

$$\;=\; \frac{1}{\tau_j^2}\,(x_j^{(c)} \;-\; g_j y^{(c)})\, y^{(c)} \tag{14}$$

The averaging operation by which the generative variances, $\tau_j^2$, are learned will (for $\alpha$ 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 this fantasy case are stochastically generated according to equation (2), as described at the beginning of this section. The connection weights for the recognition model are then updated using the delta rule, as follows:

$$r_i' \;=\; r_i \;+\; \eta\,(y^{(f)} - \mathbf{r}^T\mathbf{x}^{(f)})\, x_i \tag{15}$$

where $\eta$ 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, $\sigma^2$, is updated as follows:

$$(\sigma^2)' \;=\; \alpha\,\sigma^2 \;+\; (1-\alpha)\,(y^{(f)} - \mathbf{r}^T\mathbf{x}^{(f)})^2 \tag{16}$$

where $\alpha$ is again a learning rate parameter slightly less than one.

The aim of sleep phase learning is to improve the recognition model's ability to produce the conditional distribution that would be computed in the E-step of EM. However, in contrast to the exact correspondence between the wake phase and the M-step of EM, the operation of the sleep phase does not correspond in any obvious way to an exact implementation of the E-step. This problem with justifying the wake-sleep procedure is discussed further in Section 5.1.

In the experiments reported in Section 4, wake-phase learning steps (based on cases taken in turn from the training set) alternated with sleep-phase learning steps, and the same learning rate was usually used in both phases. Such balanced operation of the two phases is not essential for convergence to the correct solution, however. This contrasts with the "wake" and "sleep" phases of learning in Boltzmann Machines (Hinton and Sejnowski 1986), in which the two phases must be exactly balanced for the learning to follow an appropriate gradient.

### 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 means of zero):

$$\mathbf{y} \;=\; \mathbf{R}\mathbf{x} + \boldsymbol{\nu} \tag{17}$$

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 with covariance matrix $\Sigma$, which in general (ie, 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 $\Sigma$, 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 chain 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, $\sigma_i^2$. 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 $\Sigma$. Accordingly, a recognition model of this form exists that can perfectly invert any generative model.

This method is reminiscent of some of the proposals to allow Hebbian learning to extract more than one principal component (eg, Sanger 1989; Plumbley 1993); various of these also order the "hidden" units. However, in these proposals, the connections are use to *remove* correlations between the units so that they can represent different facets of the input. 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 conditional on given values for the visible variables that may be induced by this joint responsibility for modeling the visible variables.

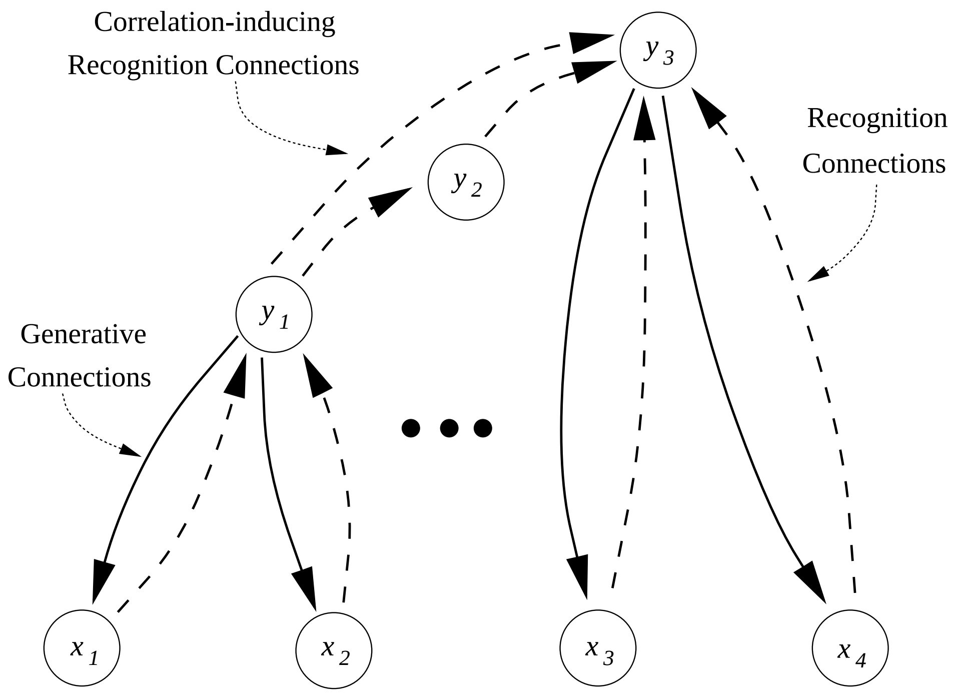

Figure 2: A Helmholtz machine implementing a model with three hidden factors, using a recognition model with a chain of correlation-inducing connections. The figure omits some of the generative connections from hidden factors to visible variables, and some of the recognition connections from visible variables to hidden factors (they are 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.

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 is always at least one rotation in which the recognition covariance matrix, $\Sigma$, is diagonal, and can therefore be represented by just $k$ variances, $\sigma_i^2$, 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 non-linear 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 (5) and (17) 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 input.

## 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 can be relatively slow. We also report in Section 4.4 the results of applying the wake-sleep algorithm to a real dataset used by Everitt (1984).

### 4.1 Experimental procedure

All the systematic experiments described below 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, ten sets of model parameters were generated, each of which was used to generate two sets of training data, the first with 10 cases, 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 re-scaled so as to produce a variance of one for each visible variable — ie, 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. (Note that symmetry is broken by the stochastic nature of the procedure.) Learning was done "on-line", 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 $\eta$ and $\alpha$ 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 knowledge of this 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 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

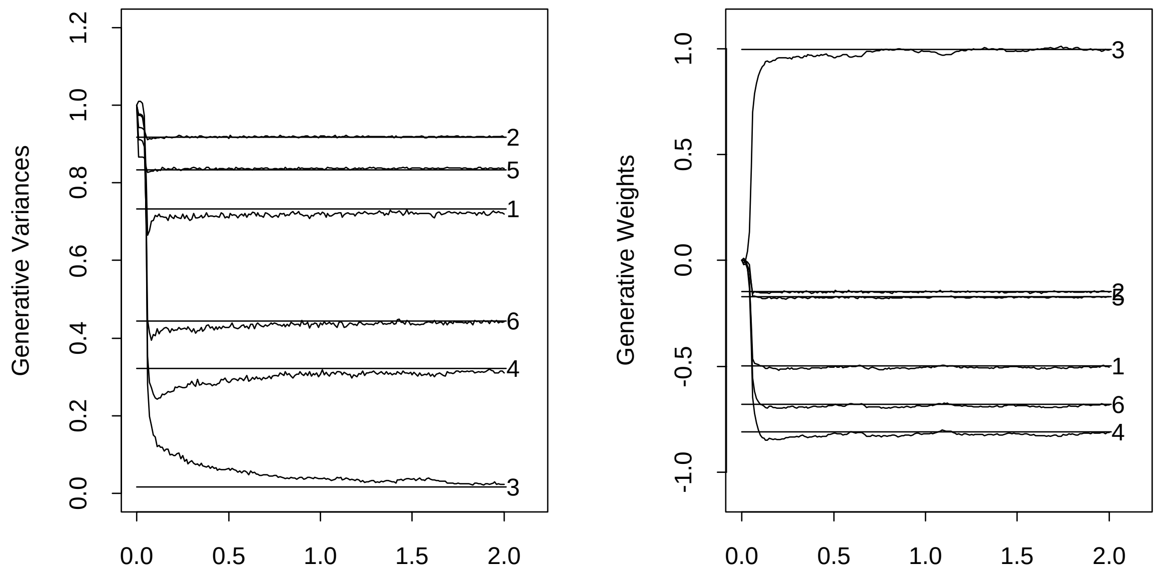

Figure 3: Wake-sleep learning of a single-factor model for six variables. The graphs show the progress of the generative variances and weights over the course of two million presentations of input vectors, drawn sequentially from a training set of size 500, with learning parameters of $\eta = 0.0002$ 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.

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.[1]

## 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 typical run with $p = 6$, applied to a training set of size 500. The figure shows progress over two million presentations of input vectors (ie, 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

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

procedure. Of the twenty runs with $p = 6$ (based on data generated from ten random models, with training sets of size 10 and 500), all but one showed similar convergence to the S-Plus estimates within three 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 (ie, 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 ten runs with training sets of size 10 and one of the ten runs with training sets of size 500 failed to clearly converge within three million presentations, but convergence was seen when these runs were extended to ten 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 above 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 which one might speculate would cause problems, due to the recognition model 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 (ie, $k = 2$ 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 twenty datasets converged within the six million presentations that were initially performed; a few required more iterations before convergence was apparent. For two datasets (both of size 10), wake-sleep learning converged to different local maxima than S-Plus (ones which S-Plus also found when started at the parameter estimates found by wake-sleep). 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.

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 datasets with $k = 2$ and $p = 8$. All runs converged, most within six million presentations. One run (on a training set of size 10) converged to a different local maximum than S-Plus. We also tried runs with the same model and datasets 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 datasets the results were still generally correct.

Finally, runs using models without correlation-inducing recognition connections were performed for datasets with $k = 3$ and $p = 9$. Most of these converged fine, in a few cases to alternative local maxima. However, for two datasets (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 dataset 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, both for models with a correlation-inducing recognition connection between the two factors, and for models without such a connection. All 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 4(a) shows one of the runs with discrepancies, for a model with a correlation-inducing connection present. Figure 4(b) shows a run differing from that of (a) 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 4(c). Close comparison of (b) and (c) shows that even in this run, convergence was much more laboured 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 and / 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 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 (ie, in the wake phase), while leaving the recognition learning rate unchanged. As discussed in Section 5.1, there is a theoretical reason to think that this method will lead to the maximum likelihood estimates. 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 non-zero means. This is not a very sensible thing to do — the means of the variables in this dataset are far from zero, so learning would at best take

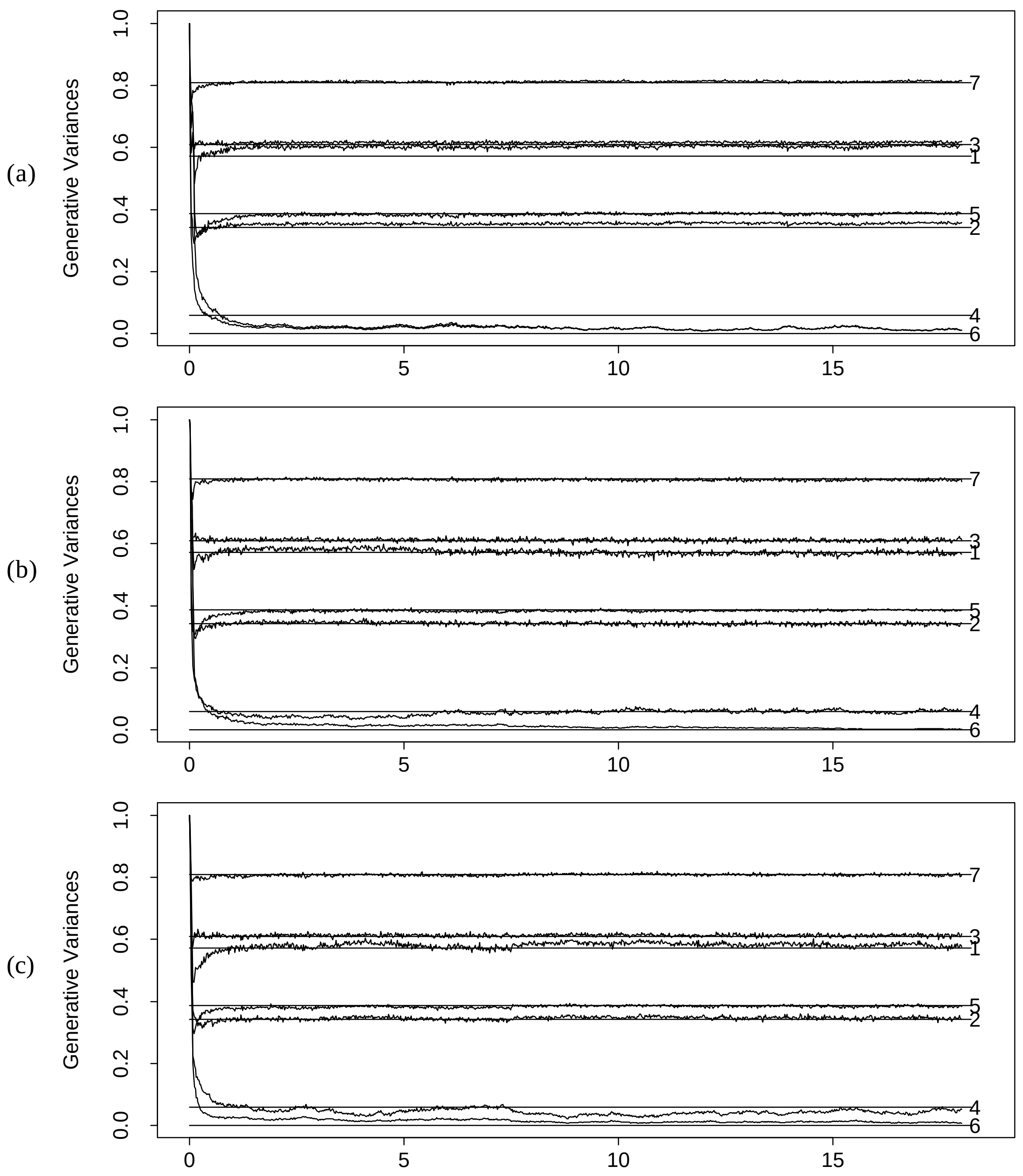

Figure 4: Wake-sleep learning of two-factor models for the Everitt crime data. The horizontal axis shows the number of presentations, in millions. The vertical axis shows the generative variances, with the maximum likelihood values indicated by horizontal lines. (a) One of the six runs in which a model 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.

a long time, while the biases slowly adjusted. In fact, however, wake-sleep learning fails 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. In most situations, the wake-sleep method 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 one should note that the estimates found by wake-sleep are reasonably good even in these situations.

Just as it is usually computationally more efficient to implement principal components 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 and Thayer 1982) or the second order Newton methods of Jöreskog (1967, 1969, 1977) than by the wake-sleep algorithm. In our view, the application of the wake-sleep algorithm to factor analysis is interesting as a possible model of activity-dependent plasticity in the cortex, and as a simple and successful example of wake-sleep learning. We discuss below some possible future work relating to these areas.

### 5.1 Theoretical analysis of the wake-sleep algorithm

A straightforward analysis of the wake-sleep algorithm would be possible if we could show that wake-sleep learning changes the parameters of the generative and the recognition models in a fashion that leads to a minimum of a function whose minima correspond to local maxima of the likelihood for the generative model. A candidate for such a function is the cost, $C(d)$, defined by Hinton, *et al.* (1995) (their equation (5)), which is derived by considering the number of bits it takes to code a visible vector using a procedure that employs the distribution for hidden variables defined by the recognition model. Indeed, one can show that updates of the generative parameters made during the wake phase do produce a decrease in this cost function. Difficulties arise during sleep, as the learning rule during this phase was derived by imagining that the network is reversed. This is intuitively reasonable, since the goal of sleep learning is to make the recognition model invert the generative model. Formally, however, the learning that occurs during sleep corresponds to minimizing a Kullback-Leibler divergence in which the two distributions appear in the wrong order, and in which the distribution over visible variables is that produced by the generative model rather than the external world (Hinton *et al.* 1995). An algorithm that correctly performs stochastic gradient descent in the recognition parameters using a correct cost function does exist (Dayan and Hinton 1996), but unfortunately, it involves reinforcement learning methods for which convergence is usually extremely slow.

We have nevertheless derived two convergence results. The first relies on the fact that when wake-sleep learning is applied to a factor analysis model with a single factor, or with multiple factors having correlation-inducing recognition connections, adaptation of the recognition parameters will ultimately lead to a recognition model that exactly inverts the generative model, if the generative model is kept fixed during the process. It follows that wake-sleep learning will work if the recognition learning rate is much larger than the generative learning rate (and both are sufficiently small), as the method will then be equivalent to EM.

The second convergence result is that wake-sleep learning is second-order Lyapunov stable near to any stable point at which the recognition weights are correct. This offers a very weak guarantee that the actual solution is not unstable, without providing an assurance that this solution can be found.

The empirical results reported in Section 4 indicate, however, that any general theoretical proof of correctness would have to come with some caveats regarding unstable regions of the parameter space. A proof of convergence may not be possible at all, if the discrepancies seen in the experiments are indeed to due to a “false” basin of attraction (rather than being an effect of finite learning rates). One empirical finding was that it is often more difficult to learn multiple-factor models when correlation-inducing recognition connections are not included. A possible reason for this is that without such connections, an arbitrary generative model may not be perfectly invertible, even though an equivalent generative model that can be perfectly inverted always exists.

Another possible avenue for theoretical analysis would be to show that wake-sleep learning leads to “consistent” estimates (ie, estimates that converge to the true parameters when based on unlimited amounts of data generated from the model being used). This could well be true even if wake-sleep learning does not always produce the exact maximum likelihood estimates. This criterion for correctness may in any case be more relevant in a true “on-line” context, in which each training case is seen only once.

## 5.2 Possible hierarchical extensions

There are limits to what a linear factor analysis model can do to capture dependencies between observed variables. One possible extension is to use a mixture of factor analysers (perhaps implemented using an EM-style algorithm). This has been applied successfully in the domain of character recognition (Hinton, Revow, and Dayan 1995), although mixtures of factor analysis models did not perform much better than mixtures of principal component models. One could also augment such a mixture model by allowing the mixing proportions to vary under the control of a gating network (Jacobs, Jordan, Nowlan, and Hinton 1991).

Another possibility is to build a hierarchical model (Ghahramani and Hinton, personal communication; Rao and Ballard 1995). Willsky and his colleagues (Chou, Willsky, and Benveniste 1994; Chou, Willsky, and Nikoukhah 1994; Krim, Willsky, and Karl 1994; Luettgen and Willsky 1995) have built a sophisticated multi-resolution tree architecture for images that combines interconnected factor analysers at different spatial resolutions. The advantage of the tree is that the E-step of EM can be done using a single bottom-up pass followed by a single top-down pass. This is closely related to the Rauch-Tung-Striebel filtering / smoothing algorithm. Of course, since the whole system is linear, it remains incapable of capturing more than second-order correlations between inputs, even though the prior covariance matrix over the image can have a very sophisticated structure. The single bottom-up and top-down passes for calculating the true recognition distribution is a good place to start in designing more local schemes for propagating information up and down a hierarchical factor analyser. This is then suggestive of mean field methods (Saul *et al* 1995; Jaakkola *et al.* 1995; Rao and Ballard 1995), which would be exact in such a linear version. Combining bottom-up and top-down information is very important for many applications, and the generative model should rightly police this combination.

Extracting more than second order structure requires using non-linearities somewhere in the system. The price to be paid is that there are no guarantees that the recognition distribution will be tractable, and approximations will typically have to be made. In this realm, the Helmholtz machine (Hinton *et al.* 1995), mean field methods for Bayesian networks (Saul *et al.* 1995; Jaakkola *et al.* 1995), and the sparse coding proposal of Olshausen and Field (1995) show promise, though their worth has yet to be proven empirically.

## 5.3 Implications for activity-dependent plasticity

A main motivation for the original development of the Helmholtz machine was to understand activity-dependent development in hierarchical structures such as the cortex, using activation and learning rules that are no less local than the Hebb rule. Of course, much progress has been made in modeling activity-dependent plasticity using simple Hebbian learning rules embedded in linear networks, with limited nonlinearities such as synaptic normalisation to prevent weights from growing without bound. Networks like these have been used to model the formation of observable anatomical structures in primary sensory cortices, such as topographic maps (Willshaw and von der Malsburg 1976, 1979; von der Malsburg and Willshaw 1977), ocular dominance stripes, in which nearby groups of cells respond preferentially to one eye rather than the other (Miller, Keller, and Stryker 1989), and orientation domains, in which groups of nearby cells respond to bars of light of nearby position and orientation on the retina (Linkser 1988; Miller 1994).

However, Hebbian learning is theoretically unsatisfying in two respects. First, it is justified as maximising the transfer of Shannon information through a network. This may be undesirable in situations where some of this information is really noise. Maximising information is also an appropriate goal only for the first stage of processing, as there can be no more information in the second layer of a network in cortex than in the first layer, if that first layer forms its entire input. Of course, developmental models generally impose restrictions in the connectivity, allowing higher layers to extract information by combining cells in lower layers whose receptive fields do not completely overlap. A second unsatisfying characteristic of Hebbian learning models is that they accord no role to prominent features of the cortex, notably the top-down connections that ubiquitously follow the bottom-up projections.

The Helmholtz machine and its associated wake-sleep learning algorithm are a potential alternative theory for cortical self-organisation, which may be able to avoid some of these problems. Because its goal is to build a good statistical model of its inputs, rather than simply communicate information, the Helmholtz machine does not pay undue attention to noise, and does not consign higher layers to an intrinsically futile role. Furthermore, the Helmholtz machine's generative model, embodied in top-down connections, parallels its recognition model, embodied in bottom-up connections. Weights for both sets of connections can be learned by the purely local delta rule, although cortical microcircuitry is required to construct the predictions (such as $g_j y^{(c)}$ of equation (11)) or prediction errors (such as $x_j^{(c)} - g_j y^{(c)}$ of the same equation). Rules equivalent to the delta rule are conventional in classical conditioning (Rescorla and Wagner 1972; Sutton and Barto, 1981) and have also been suggested as underlying cortical plasticity (Montague and Sejnowski 1994). Of course, the wake-sleep learning rule 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 and Bower 1993), further work on the implementational microcircuitry is clearly required.

In Hebbian models of activity dependent development, a key role is played by lateral local intra-cortical connections (longer range excitatory connections are also present, but develop later). These are usually chosen to have the form of a "Mexican hat", with nearby neurons tending to excite, and distant ones tending to inhibit each other. This encourages nearby cells to have similar tuning properties and distant ones different properties. For instance, Miller, Keller and Stryker (1989) constructed a model for the development of ocular dominance stripes (bands in layer IV of striate cortex containing cells that are tuned to just one eye). In their model, the position of the peak of the Fourier transform of the local interconnections largely determines the width of the stripes. Lateral connections have also been used with generative models similar to those of Helmholtz machines for which inference is done by mean field methods (Olshausen and Field, 1996; Rao and Ballard, 1995).

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. Lateral generative connections would be active only during the sleep phase, and would force a correlation structure on the hidden factors ($\mathbf{y}$), perhaps of the Mexican hat form. The recognition model should be able to adapt to accommodate this correlation structure, although recognition models that assume independence of factors might no longer be viable, since this model lacks the rotational symmetry needed to ensure that a model showing such independence exits. Structures such as ocular dominance stripes should form in such a network architecture.

Lateral connections in the recognition model could replace the chain of correlation-inducing connections shown in Figure 2. It would be interesting to investigate whether the same connections might be able to support correlations in both the generative and the recognition models.

Purely linear models, such as those presented here, will not suffice to fully explain 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 the first step to understanding how more complex statistical models can be learned using more complex architectures involving non-linear elements.

## Acknowledgements

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.

Baudry, M. and Davis, J. L. (1994) *Long-term Potentiation: A Debate of Current Issues, Volume 2* Cambridge, Massachusetts: MIT Press.

Chou, K. C.,Willsky, A. S. and Benveniste, A. (1994) Multiscale recursive estimation, data fusion, and regularization, *IEEE Transactions on Automatic Control,* **39**, 464-478.

Chou, K. C., Willsky, A. S. and Nikoukhah, R. (1994) Multiscale systems, Kalman filters, and Riccati equations, *IEEE Transactions on Automatic Control,* **39**, 479-492.

Collingridge, G. L. and Bliss, T. V. (1987) NMDA receptors: Their role in long-term potentiation, *Trends in Neurosciences,* **10**, 288-293.

Dayan, P. and Hinton, G. E. (1996) Varieties of Helmholtz Machine, *Neural Networks,* in press.

Dayan, P., Hinton, G. E., Neal, R. M., and Zemel, R.S. (1995) The Helmholtz machine, *Neural Computation,* **7**, 229-904.

Dempster, A. P., Laird, N. M, and Rubin, D. B. (1977) Maximum likelihood from incomplete data via the EM algorithm, *Proceedings of the Royal Statistical Society,* **B 39**, 1-38.

Everitt, B. S. (1984) *An Introduction to Latent Variable Models,* London: Chapman and Hall.

Földiák, P. (1989) Adaptive network for optimal linear feature extraction, *Proceedings of the International Joint Conference on Neural Networks, Washington, DC,* **I**, 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. and Bower, J. M. (1993) Acetylcholine and memory, *Trends in Neurosciences,* **16**, 218-222.

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

Hinton, G. E., Revow, M. and Dayan, P. (1995) Recognizing handwritten digits using mixtures of linear models, in G. Tesauro, D. S. Touretzky, and T. K. Leen (editors), *Advances in Neural Information Processing Systems, 7,* 1015-1022. Cambridge, MA: MIT Press.

Hinton, G. E. and Sejnowski, T. J. (1986) Learning and relearning in Boltzmann machines, in D. E. Rumelhart, J. L. McClelland, and the PDP research group, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Volume 1: Foundations,* Cambridge, Massachusetts: MIT Press, 282-317.

Hinton, G. E. and Zemel, R. S. (1994) Autoencoders, minimum description length and Helmholtz free energy, in J. D. Cowan, G. Tesauro and J. Alspector (editors), *Advances in Neural Information Processing Systems 6,* San Mateo, California: Morgan Kaufmann, 3-10.

Jacobs, R. A., Jordan, M. I., Nowlan, S. J., and Hinton, G. E. (1991) Adaptive mixtures of local experts, *Neural Computation,* **3**, 79-87.

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, and H. S. Wilf (editors), *Statistical Methods for Digital Computers* (volume 3 of Mathematical Methods for Digital Computers), New York: Wiley.

Krim, H., Willsky, A. S. and Karl, W.C. (1994) Multiresolution models for random fields and their use in statistical image processing, *Proceedings of 1994 Workshop on Information Theory and Statistics,* New York: IEEE, 56.

Li, Z. and Atick, J. J. (1994) Toward a theory of the striate cortex, *Neural Computation,* **6**, 127-146.

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.

Luettgen, M. R. and Willsky, A.S. (1995) Likelihood calculation for a class of multiscale stochastic models, with application to texture discrimination, *IEEE Transactions on Image Processing,* **4**, 194-207.

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. and Stryker, M. P. (1989) Ocular dominance column development: Analysis and simulation, *Science*, **245**, pp 605-615.

Montague, P. R. and 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 and J. Davis (editors), *Large-Scale Theories of the Cortex*, Cambridge, MA: MIT Press, 125-152.

Olshausen, B. A. and Field, D. J. (1996) Sparse coding of natural images produces localized, oriented, bandpass receptive fields, *Nature*, in press.

Oja, E. (1989) Neural networks, principal components, and subspaces, *International Journal of Neural Systems*, **1**, 61-8.

Oja, E. (1992) Principal components, minor components, and linear neural networks, *Neural Networks*, **5**, 927-35.

Oja, E. and Karhunen, J. (1985) On stochastic approximation of the eigenvectors and eigenvalues of the expectation of a random matrix, *Journal of Mathematical Analysis and Applications*, **106**, 69-84.

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

Rao, P. N. R. and Ballard, D. H. (1995) Dynamic model of visual memory predicts neural response properties in the visual cortex, Technical report 95.4, Department of Computer Science, Rochester, New York.

Rescorla, R. A. and Wagner, A.R. (1972) A theory of Pavlovian conditioning: The effectiveness of reinforcement and non-reinforcement, in A. H. Black and W. F. Prokasy (editors), *Classical Conditioning II: Current Research and Theory*, pp 64-69. New York: Appleton-Century-Crofts.

Rubin, D. B. and 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.

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

von der Malsburg, C. and 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. and 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. and 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**, pp 203-243.

Wyatt, J. L., Jr and Elfadel, I. M. (1985) Time-domain solutions of Oja's equations, *Neural Computation*, **7**, 915-22.