Density Networks and their Application to Protein Modelling

David J.C. MacKay
Cavendish Laboratory, Cambridge, CB3 0HE. U.K.
mackay@mrao.cam.ac.uk

ABSTRACT. I define a latent variable model in the form of a neural network for which only target outputs are specified; the inputs are unspecified. Although the inputs are missing, it is still possible to train this model by placing a simple probability distribution on the unknown inputs and maximizing the probability of the data given the parameters. The model can then discover for itself a description of the data in terms of an underlying latent variable space of lower dimensionality. I present preliminary results of the application of these models to protein data.

1 Density Modelling

The most popular supervised neural networks, multilayer perceptrons (MLPs), are well established as probabilistic models for *regression* and *classification*, both of which are *conditional* modelling tasks: the *input* variables are assumed given, and we *condition* on their values when modelling the distribution over the *output* variables; no model of the density over input variables is constructed. Density modelling (or generative modelling), on the other hand, denotes modelling tasks in which a density over *all* the observable quantities is constructed. Multi-layer perceptrons have not conventionally been used to create density models (though belief networks and other neural networks such as the Boltzmann machine do define density models). Various interesting research problems in this field relate to the difficulty of defining a full probabilistic model with an MLP. For example, if some inputs in a regression problem are 'missing', then traditional methods offer no principled way of filling the gaps. This paper discusses how one can use an MLP as a density model.

Traditional density models

A popular class of density models are *mixture models*, which define the density as a sum of simpler densities. Mixture models might however be viewed as inappropriate models for high-dimensional data spaces such as images or genome sequences. The number of components in a mixture model has to scale exponentially as we add independent degrees of freedom. Consider, for example, a protein family in which there is a strong correlation between the amino acids in the first and second columns — they are either both hydrophobic, or both hydrophilic, say — and there is an independent correlation between two other amino acids elsewhere in the protein chain — when one of them has a large residue the other has a small residue, say. A mixture model would have to use four categories to capture all four combinations of these binary attributes, whereas only two independent degrees of freedom are really present. Thus a *combinatorial* representation of underlying variables would seem more appropriate. [Luttrell's (1994) partitioned mixture distribution is motivated similarly, but is a different form of quasi-probabilistic model.]

These observations motivate the development of density models that have *components* rather than *categories* as their 'latent variables' (Everitt 1984; Hinton and Zemel 1994). Let us denote the observables by $\mathbf{t}$. If a density is defined on the latent variables $\mathbf{x}$, and a parameterized mapping is defined from these latent variables to a probability distribution over the observables $P(\mathbf{t}|\mathbf{x}, \mathbf{w})$, then when we integrate over the unknowns $\mathbf{x}$, a non-trivial density over $\mathbf{t}$ is defined, $P(\mathbf{t}|\mathbf{w}) = \int d\mathbf{x}\, P(\mathbf{t}|\mathbf{x}, \mathbf{w})P(\mathbf{x})$. Simple linear models of this form in the statistics literature come under the label of 'factor analysis'. In a 'density network' (MacKay 1995) $P(\mathbf{t}|\mathbf{x}, \mathbf{w})$ is defined by a more general non-linear parameterized mapping, and interesting priors on $\mathbf{w}$ may be used.

The model

The 'latent inputs' of the model are a vector $\mathbf{x}$ indexed by $h = 1 \ldots H$ ('h' mnemonic for 'hidden'). The dimensionality of this hidden space is H but the effective dimensionality assigned by the model in the output space may be smaller, as some of the hidden dimensions may be effectively unused by the model. The relationship between the latent inputs and the observables, parameterized by $\mathbf{w}$, has the form of a mapping from inputs to outputs $\mathbf{y}(\mathbf{x}; \mathbf{w})$, and a probability of targets given outputs, $P(\mathbf{t}|\mathbf{y})$. The observed data are a set of target vectors $D = \{\mathbf{t}^{(n)}\}_{n=1}^{N}$. To complete the model we assign a prior $P(\mathbf{x})$ to the latent inputs (an independent prior for each vector $\mathbf{x}^{(n)}$) and a prior $P(\mathbf{w})$ to the unknown parameters. [In the applications that follow the priors over $\mathbf{w}$ and $\mathbf{x}^{(n)}$ are assumed to be spherical Gaussians; other distributions could easily be implemented and compared, if desired.] In summary, the probability of everything is:

$$P(D, \{\mathbf{x}^{(n)}\}, \mathbf{w}|\mathcal{H}) = \prod_n \left[P(\mathbf{t}^{(n)}|\mathbf{x}^{(n)}, \mathbf{w}, \mathcal{H})P(\mathbf{x}^{(n)}|\mathcal{H})\right] P(\mathbf{w}|\mathcal{H}) \tag{1}$$

It will be convenient to define 'error functions' $G^{(n)}(\mathbf{x}; \mathbf{w})$ as follows:

$$G^{(n)}(\mathbf{x}; \mathbf{w}) \equiv \log P(\mathbf{t}^{(n)}|\mathbf{x}^{(n)}, \mathbf{w}) \tag{2}$$

The function G depends on the nature of the problem. If $\mathbf{t}$ consists of real variables then G might be a sum-squared error between $\mathbf{t}$ and $\mathbf{y}$; in a 'softmax' classifier where the observations $\mathbf{t}$ are categorical, G is a 'cross entropy'. In general we may have many output groups of different types. The following derivation applies to all cases. Subsequently this paper concentrates on the following form of model, which may be useful to have in mind. The observable $\mathbf{t} = \{t_s\}_{s=1}^{S}$ (*e.g.*, a single protein sequence) consists of a number S of categorical attributes that are believed to be correlated (S will be the number of columns in the protein alignment). Each attribute can take one of a number I of discrete values, a probability over which is modelled with a softmax group (*e.g.*, $I = 20$).

$$P(\mathbf{t}|\mathbf{x}, \mathbf{w}) = \prod_{s=1}^{S} \{y_{t_s}^{s}(\mathbf{x}; \mathbf{w})\} \tag{3}$$

where

$$y_i^s(\mathbf{x}; \mathbf{w}) = \frac{e^{a_i^s(\mathbf{x};\mathbf{w})}}{\sum_{i'} e^{a_{i'}^s(\mathbf{x};\mathbf{w})}}. \tag{4}$$

The parameters $\mathbf{w}$ form a matrix of $(H+1) \times S \times I$ weights from the H latent inputs $\mathbf{x}$ (and one bias) to the $S \times I$ outputs:

$$a_i^s(\mathbf{x};\mathbf{w}) = w_{i0}^s + \sum_{h=1}^{H} w_{ih}^s x_h \tag{5}$$

The data items $\mathbf{t}$ are labelled by an index $n = 1 \ldots N$, not included in the above equations, and the error function $G^{(n)}$ is

$$G^{(n)} = \sum_s \log y_{t_s^{(n)}}. \tag{6}$$

Having written down the probability of everything (equation 1) we can now make any desired inferences by turning the handle of probability theory. Let us aim towards the inference of the parameters $\mathbf{w}$ given the data D, $P(\mathbf{w}|D,\mathcal{H})$. We can obtain this quantity conveniently by distinguishing two levels of inference.

Level 1: Given $\mathbf{w}$ and $\mathbf{t}^{(n)}$, infer $\mathbf{x}^{(n)}$. The posterior distribution of $\mathbf{x}^{(n)}$ is

$$P(\mathbf{x}^{(n)}|\mathbf{t}^{(n)},\mathbf{w},\mathcal{H}) = \frac{P(\mathbf{t}^{(n)}|\mathbf{x}^{(n)},\mathbf{w},\mathcal{H})P(\mathbf{x}^{(n)}|\mathcal{H})}{P(\mathbf{t}^{(n)}|\mathbf{w},\mathcal{H})}, \tag{7}$$

where the normalizing constant is:

$$P(\mathbf{t}^{(n)}|\mathbf{w},\mathcal{H}) = \int d^H\mathbf{x}^{(n)}\, P(\mathbf{t}^{(n)}|\mathbf{x}^{(n)},\mathbf{w},\mathcal{H})P(\mathbf{x}^{(n)}|\mathcal{H}). \tag{8}$$

Level 2: Given $D = \{\mathbf{t}^{(n)}\}$, infer $\mathbf{w}$.

$$P(\mathbf{w}|D,\mathcal{H}) = \frac{P(D|\mathbf{w},\mathcal{H})P(\mathbf{w}|\mathcal{H})}{P(D|\mathcal{H})} \tag{9}$$

The data-dependent term here is a product of the normalizing constants of the level 1 inferences:

$$P(D|\mathbf{w},\mathcal{H}) = \prod_{n=1}^{N} P(\mathbf{t}^{(n)}|\mathbf{w},\mathcal{H}). \tag{10}$$

The evaluation of the evidence $P(\mathbf{t}^{(n)}|\mathbf{w},\mathcal{H})$ for a particular n is a problem similar to the evaluation of the evidence for a supervised neural network (MacKay 1992). There, the inputs $\mathbf{x}$ are given, and the parameters $\mathbf{w}$ are unknown; we obtain the evidence by integrating over $\mathbf{w}$. In the present problem, on the other hand, the hidden vector $\mathbf{x}^{(n)}$ is unknown, and the parameters $\mathbf{w}$ are conditionally fixed. For each n, we wish to integrate over $\mathbf{x}^{(n)}$ to obtain the evidence.

Learning: the derivative of the evidence with respect to $\mathbf{w}$

The derivative of the log of the evidence (equation 8) is:

$$\begin{aligned}\frac{\partial}{\partial \mathbf{w}} \log P(\mathbf{t}^{(n)}|\mathbf{w},\mathcal{H}) &= \frac{1}{P(\mathbf{t}^{(n)}|\mathbf{w},\mathcal{H})} \int d^H\mathbf{x}\, \exp(G^{(n)}(\mathbf{x};\mathbf{w}))P(\mathbf{x}|\mathcal{H}) \frac{\partial}{\partial \mathbf{w}} G^{(n)}(\mathbf{x};\mathbf{w}) \\ &= \int d^H\mathbf{x}\, P(\mathbf{x}|\mathbf{t}^{(n)},\mathbf{w},\mathcal{H}) \frac{\partial}{\partial \mathbf{w}} G^{(n)}(\mathbf{x};\mathbf{w}). \end{aligned} \tag{11}$$

This gradient can thus be written as an expectation of the traditional 'backpropagation' gradient $\frac{\partial}{\partial \mathbf{w}} G^{(n)}(\mathbf{x};\mathbf{w})$, averaging over the posterior distribution of $\mathbf{x}^{(n)}$ found in equation (7).

Higher levels — priors on $\mathbf{w}$

We can continue up the hierarchical model, putting a prior on $\mathbf{w}$ with hyperparameters $\{\alpha\}$ which are inferred by integrating over $\mathbf{w}$. These priors are important from a practical point of view to limit overfitting of the data by the parameters $\mathbf{w}$. These priors will also be used to bias the solutions towards ones that are easier for humans to interpret.

Evaluation of the evidence and its derivatives using simple Monte Carlo

The evidence and its derivatives with respect to $\mathbf{w}$ both involve integrals over the hidden components $\mathbf{x}$. For a hidden vector of sufficiently small dimensionality, a simple Monte Carlo approach to the evaluation of these integrals can be effective.

Let $\{\mathbf{x}^{(r)}\}_{r=1}^{R}$ be random samples from $P(\mathbf{x})$. Then we can approximate the log evidence by:

$$\begin{aligned} \log P(\{\mathbf{t}^{(n)}\}|\mathbf{w},\mathcal{H}) &= \sum_n \log \int d^H\mathbf{x}\, \exp(G^n(\mathbf{x};\mathbf{w}))P(\mathbf{x}) \\ &\simeq \sum_n \log\left[\frac{1}{R}\sum_r \exp(G^n(\mathbf{x}^{(r)};\mathbf{w}))\right]. \end{aligned}$$

Similarly the derivative can be approximated by:

$$\frac{\partial}{\partial \mathbf{w}} \log P(\{\mathbf{t}^{(n)}\}|\mathbf{w},\mathcal{H}) \simeq \sum_n \frac{\sum_r \exp(G^n(\mathbf{x}^{(r)};\mathbf{w}))\frac{\partial}{\partial \mathbf{w}} G^n(\mathbf{x}^{(r)};\mathbf{w})}{\sum_r \exp(G^n(\mathbf{x}^{(r)};\mathbf{w})).} \tag{12}$$

This simple Monte Carlo approach loses the advantage that we gained when we rejected mixture models and turned to componential models; this implementation of the componential model requires a number of samples R that is exponential in the dimension of the hidden space H. More sophisticated methods using stochastic dynamics (Neal 1993) are currently under development.

Alternative implementations

An alternative approach to making such componential models scale well is the **free energy minimization** approximation of Hinton and Zemel (1994). They introduce a distribution $Q^n(\mathbf{x})$ that is intended to be similar to the posterior distribution $P(\mathbf{x}|\mathbf{t}^{(n)},\mathbf{w},\mathcal{H})$; Q is written as a nonlinear function of the observable $\mathbf{t}^{(n)}$; the parameters of this nonlinear function are then optimized so as to make $Q^n(\mathbf{x})$ the best possible approximation to $P(\mathbf{x}|\mathbf{t}^{(n)},\mathbf{w},\mathcal{H})$ (for all n) as measured by a free energy, $\sum_n \int d\mathbf{x}\, Q \log(Q/P)$. This method gives an approximate lower bound for the log evidence. If R random samples $\{\mathbf{x}^{(r)}\}_{r=1}^{R}$ from $Q(\mathbf{x})$ are made, then:

$$\begin{aligned} \log P(\mathbf{t}^{(n)}|\mathbf{w},\mathcal{H}) &= \log \int d^H\mathbf{x}\, \exp(G^n(\mathbf{x};\mathbf{w}))P(\mathbf{x}) \\ &\leq \int d\mathbf{x}\, Q(\mathbf{x}) \log \frac{\left[P(\mathbf{x})e^{G^{(n)}(\mathbf{x};\mathbf{w})}\right]}{Q(\mathbf{x})} \\ &\lesssim \frac{1}{R}\sum_r \left[G^{(n)}(\mathbf{x};\mathbf{w}) + \log P(\mathbf{x}) - \log Q(\mathbf{x})\right]. \end{aligned}$$

An alternative formula for estimating the evidence is given by **importance sampling**:

$$\log P(\mathbf{t}^{(n)}|\mathbf{w},\mathcal{H}) \simeq \log\left[\frac{1}{R}\sum_r \exp(G^n(\mathbf{x};\mathbf{w}))\frac{P(\mathbf{x})}{Q^n(\mathbf{x})}\right].$$

2 A componential density model for a protein family

A protein is a sequence of amino acids. A protein family is a set of proteins believed to have the same physical structure but not necessarily having the same sequence of amino acids. In a *multiple sequence alignment*, residues of the individual sequences which occupy structurally analogous positions are aligned into columns. There are twenty different amino acids, and columns can often be characterized by a predominance of particular amino acids. Lists of marginal frequencies over amino acids in different structural contexts are given in (Nakai *et al.* 1988).

The development of models for protein families is useful for two reasons. The first is that a good model might be used to identify new members of an existing family, and discover new families too, in data produced by genome sequencing projects. The second reason is that a sufficiently complex model might be able to give new insight into the properties of the protein family; for example, properties of the proteins' tertiary structure might be elucidated by a model capable of discovering suspicious inter-column correlations.

The only probabilistic model that has so far been applied to protein families is a hidden Markov model (Krogh *et al.* 1994). This model is not inherently capable of discovering long-range correlations, as Markov models, by definition, produce no correlations between the observables, given a hidden state sequence.

The next-door neighbour of proteins, RNA, has been modelled with a 'covariance model' capable of capturing correlations between base-pairs in anti-parallel RNA strands (Eddy and Durbin 1994). The aim of the present work is to develop a model capable of discovering general correlations between multiple arbitrary columns in a protein family. E. Steeg (personal communication) has developed an efficient statistical test for discovering correlated groups of residues. The present work is complementary to Steeg's in that (1) in the density network, a residue may be influenced by more than one latent variable; whereas Steeg's test is specialised for the case where the correlated groups are non-overlapping; (2) the density networks developed here define full probabilistic models rather than statistical tests.

Here I model the protein families using a density network containing one softmax group for each column (see equations 3–6). The network has only one layer of weights connecting the latent variables $\mathbf{x}$ directly to the softmax groups. I have optimized $\mathbf{w}$ by evaluating the evidence and its gradient and feeding them into a conjugate gradient routine. The random points $\{\mathbf{x}^{(r)}\}$ are kept fixed, so that the objective function and its gradient are deterministic functions during the optimization. This also has the advantage of allowing one to get away with a smaller number of samples R than might be thought necessary, as the parameters $\mathbf{w}$ can adapt to make the best use of the empirical distribution over $\mathbf{x}$.

Regularization schemes

A human prejudice towards comprehensible solutions gives an additional motivation for regularizing the model, beyond the usual reasons for having priors. Here I encourage the model to be comprehensible in two ways:

1. There is a redundancy in the model regarding where it gets its randomness from. Assume that a particular output is actually random and uncorrelated with other outputs. This could be modelled in two ways: its weights from the latent inputs could be set to zero, and the biases could be set to the log probabilities; or alternatively the biases could be fixed to arbitrary values, with appropriate connections to unused latent inputs being used to create the required probabilities, on marginalization over the latent variables. In predictive terms, these two models would be identical, but we prefer the first solution, finding it more intelligible. To encourage such solutions I use a prior which weakly regularizes the biases, so that they are 'cheap' relative to the other parameters.

2. If the distribution $P(\mathbf{x})$ is rotationally invariant, then the predictive distribution is invariant under corresponding transformations of the parameters $\mathbf{w}$. If a solution can be expressed in terms of parameter vectors aligned with some of the axes (*i.e.* so that some parameters are zero), then we would prefer that representation. Here I create a non-spherical prior on the parameters by using multiple undetermined regularization constants $\{\alpha_c\}$, each one associated with a class of weights (*c.f.* the automatic relevance determination model (MacKay and Neal 1994)). A weight class consists of all the weights from one latent input to one softmax group, so that for a protein with S columns modelled using H latent variables, I introduced SH regularization constants, each specifying whether a particular latent variable has an influence on a particular column. Given α_c, the prior on the parameters in class c is Gaussian with variance $1/\alpha_c$. This prior favours solutions in which one latent input has non-zero connections to all the units in some softmax groups (corresponding to small α_c), and negligible connections to other softmax groups (large α_c). The resulting solutions can easily be interpreted in terms of correlations between columns.

Method for optimization of hyperparameters

For given values of $\{\alpha_c\}$, the parameters $\mathbf{w}$ were optimized to maximize the posterior probability. No explicit Gaussian approximation was made to the posterior distribution of $\mathbf{w}$; rather, the hyperparameters $\{\alpha_c\}$ were adapted during the optimization of the parameters $\mathbf{w}$, using a cheap and cheerful method motivated by Gaussian approximations (MacKay 1992), thus:

$$\alpha_c := f \frac{k_c}{\sum_{i \in c} w_i^2}. \tag{13}$$

Here k_c is the number of parameters in class c and f is a 'fudge factor' incorporated to imitate the effect of integrating over $\mathbf{w}$ (set to a value between 0.1 and 1.0).

This algorithm could be converted to a correct 'stochastic dynamics' Monte Carlo method (Neal 1993) by adding an appropriate amount of noise to gradient descent on $\mathbf{w}$ and setting $f = 1$.

Toy data

A toy data set was created imitating a protein family with four columns each containing one of five amino acids. The 27 data (table 1) were constructed to exhibit two correlations between the columns: the first and second columns have a tendency both to be amino acid E together. The third and fourth columns are correlated such that if one is amino acid B,

EEAB	EECB	EEBC	EECC	EEAA	EEBA	EEBB	EECD
EEDC	EEDD	AACD	DDDC	CBDD	CCAB	BDCB	ABBC
CBCC	EDAA	ABBA	BCBB	DBAB	AECB	EBBC	BDCC
BCAA	DABA	BCBB					

Table 1: Toy data for a protein family

then the other is likely to be A, B or C; if one is C, then the other is likely to be B, C or D; and so forth, with an underlying single dimension running through the amino acids A,B,C,D. The model is given no prior knowledge of the 'spatial relationship' of the columns, or of the ordering of the amino acids. A model that can identify the two correlations in the data is what we are hoping for.

Both regularized and unregularized density networks having four latent inputs were adapted to this data. Unregularized density networks give solutions that successfully predict the two correlations, but the parameters of those models are hard to interpret (figure 1a). There is also evidence of overfitting of the data leading to overconfident predictions by the model. The regularized models, in which all the parameters connecting one input to one softmax group are put in a regularization class with an unknown hyperparameter α_c, give interpretable solutions that clearly identify the two correlated groups of columns. Figure 1b shows the hyperparameters and parameters inferred in a typical solution using a regularized density network. Notice that two of the latent inputs are unused in this solution. Of the other two inputs, one has an influence on columns 1 and 2 only, and the other has an influence on columns 3 and 4 only. Thus this model has successfully revealed the underlying 'structure' of the proteins in this family.

Results on real data: beta sheets

Beta sheets are structures in which two parts of the protein engage in a particular hydrogen–bonding interaction. It would greatly help in the solution of the protein folding problem if we could distinguish correct from incorrect alignments of beta strands.

Data on aligned antiparallel beta strands was provided by Tim Hubbard. $N = 1000$ examples were taken. Density networks with $H = 6$ latent inputs were used to model the joint distribution of the twelve residues surrounding a beta sheet hydrogen bond. Our prior expectation is that if there is any correlation among these residues, it is likely to reflect the spatial arrangement of the residues, with nearby residues being correlated. But this prior expectation was not included in the model. The hope was that meaningful physical properties such as this would be learned from the data.

Analysis

The parameters of a typical optimized density network are shown in figure 2.

The parameter vectors were compared, column by column, with a large number of published amino acid indices (Nakai *et al.* 1988) to see if they corresponded to established physical properties of amino acids. Each index was normalized by subtracting the mean from each vector and scaling it to unit length. The similarity of a parameter vector to an index was then measured by the magnitude of their inner product.

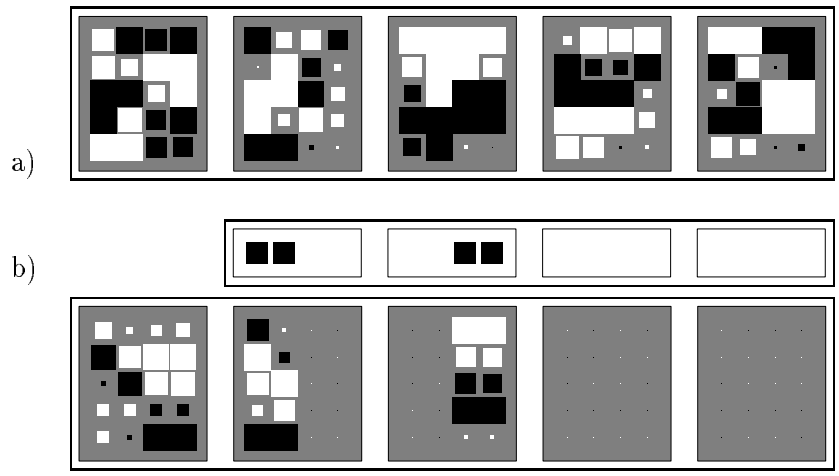

Figure 1: Parameters and Hyperparameters inferred for the toy protein family
a) Hinton diagram showing parameters **w** of model optimized without adaptive regularizers. Positive parameters are shown by black squares, negative by white. Magnitude of parameter is proportional to square area. This diagram shows, in the five grey rectangles, the *projective fields* from the bias and the four latent variables to the outputs. In each grey rectangle the influences of one latent variable on the twenty outputs are arranged in a 5×4 grid: in each column the 5 output units correspond to the 5 amino acids. It is hard to interpret these optimized parameters.
b) The hyperparameters and parameters of a hierarchical model with adaptive regularizers. The results are more intelligible and show a model that has discovered the two underlying dimensions of the data. Hyperparameters: Each hyperparameter controls all the influences of one latent variable on one column. Square size denotes the value of $\sigma_w^2 = 1/\alpha$ on a log scale from 0.001 to 1.0. The model has discovered that columns 1 and 2 are correlated with each other but not with columns 3 and 4, and vice versa. Parameters: same conventions as (a). Note the sparsity of the connections, making clear the two distinct underlying dimensions of this protein family.

Two distinctive patterns reliably emerged in most adapted models, both having a meaningful physical interpretation. First, an alternating pattern can be seen in the influences in the third rectangle from the left. The influences on columns 2, 4, 9 and 11 are similar to each other, and opposite in sign to the influences on columns 3, 5, 10 and 12. This dichotomy between the residues is physically meaningful: residues 2, 4, 9 and 11 are on the opposite side of the beta sheet plane from residues 3, 5, 10 and 12; when these influence vectors were compared with the published amino acid indices, they showed the greatest similarity to Nakai *et al.*'s (1988) indices 57, 17, 7 and 42, which respectively describe the amino acids' polarity, the proportion of residues 100% buried, the transfer free energy to surface, and the consensus normalized hydrophobicity scale. This latent variable has clearly discovered the inside–outside characteristics of the beta sheet structure: either one face of sheet is exposed to the solvent (high polarity) or the other face, but not both.

Second, a different pattern is apparent in the second rectangle from the right. Here the influences on residues 4, 5, 6, 7, 8 are similar and opposite to the influences on 11, 12, 1, 2. For five of these residues the influence vector shows greatest similarity with index number 21, the normalized frequency of beta-turn. What this latent variable has discovered,

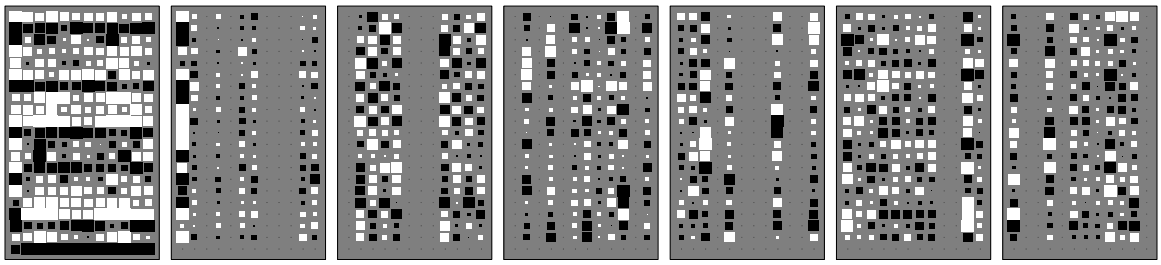

Figure 2: **Parameters w of an optimized density network modelling aligned antiparallel beta strands.**
In each grey rectangle the twelve columns represent the twelve residues surrounding a beta hydrogen bond. The twenty rows represent the twenty amino acids, in alphabetical order (A,C,D,...). Each rectangle shows the influences of one latent variable on the 12×20 probabilities.
The top left rectangle shows the biases of all the output units. There is an additional 21st row in this rectangle for the biases of the output units corresponding to 'no amino acid'. The latent variables were defined to have no influence on these outputs to inhibit the wasting of latent variables on the modelling of dull correlations. The other six rectangles contain the influences of the 6 latent variables on the output units, of which the second and fifth are discussed in the text.

therefore, is that a beta turn may happen at one end or the other of two anti-parallel beta strands, but not both.

Both of these patterns have the character of an 'exclusive-or' problem. One might imagine that an alternative way to model aligned beta sheets would be to train a *discriminative* model such as a neural network binary classifier to distinguish 'aligned beta sheet' from not aligned beta sheet. However, such a model would have difficulty learning these exclusive-or patterns. Exclusive-or *can* be learnt by a neural network with one hidden layer and two layers of weights, but it is not a natural function readily produced by such a network. In contrast these patterns are easily captured by the density networks presented here, which have only one layer of weights.

It is interesting to note that the two effects discovered above involve competing correlations between large numbers of residues. The inside-outside latent variable produces a positive correlation between columns 4 and 11, for example, while the beta turn latent variable produces a negative correlation between those two columns. These results, although they do not constitute new discoveries, suggest that this technique shows considerable promise.

Future work

More complex models under development will include additional layers of processing between the latent variables and the observables. If some of the parameters of a second layer were communal to all columns of the protein, the model would be able to generalize amino acid equivalences from one column to another.

It would be interesting to attempt to represent protein evolution as taking place in the latent variable space of a density network.

It is hoped that a density network adapted to beta sheet data will eventually be useful

for discriminating correct from incorrect alignements of beta strands. The present work is not of sufficient numerical accuracy to achieve this, but possibly by introducing superior sampling methods in tandem with free energy minimization (Hinton and Zemel 1994), these models may make a contribution to the protein folding problem.

References

EDDY, S. R., and DURBIN, R., (1994) RNA sequence analysis using covariance models. NAR, in press.

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

HINTON, G. E., and ZEMEL, R. S. (1994) Autoencoders, minimum description length and Helmholtz free energy. In *Advances in Neural Information Processing Systems 6*, ed. by J. D. Cowan, G. Tesauro, and J. Alspector, San Mateo, California. Morgan Kaufmann.

KROGH, A., BROWN, M., MIAN, I. S., SJOLANDER, K., and HAUSSLER, D. (1994) Hidden Markov models in computational biology: Applications to protein modeling. *Journal of Molecular Biology* **235**: 1501–1531.

LUTTRELL, S. P., (1994) The partitioned mixture distribution: an adaptive Bayesian network for low-level image processing. to appear.

MACKAY, D. J. C. (1992) A practical Bayesian framework for backpropagation networks. *Neural Computation* **4** (3): 448–472.

MACKAY, D. J. C. (1995) Bayesian neural networks and density networks. *Nuclear Instruments and Methods in Physics Research, Section A*.

MACKAY, D. J. C., and NEAL, R. M. (1994) Automatic relevance determination for neural networks. Technical Report in preparation, Cambridge University.

NAKAI, K., KIDERA, A., and KANEHISA, M. (1988) Cluster analysis of amino acid indices for prediction of protein structure and function. *Prot. Eng.* **2**: 93–100.

NEAL, R. M. (1993) Bayesian learning via stochastic dynamics. In *Advances in Neural Information Processing Systems 5*, ed. by C. L. Giles, S. J. Hanson, and J. D. Cowan, pp. 475–482, San Mateo, California. Morgan Kaufmann.

I thank Radford Neal, Geoff Hinton, Sean Eddy, Richard Durbin, Tim Hubbard and Graeme Mitchison for invaluable discussions. I gratefully acknowledge the support of this work by the Royal Society Smithson Research Fellowship.