

Information Theory and Neural Networks

John G. Taylor^a and Mark D. Plumbley^a

^aCentre for Neural Networks, Department of Mathematics, King's College London, Strand, London WC2R 2LS, UK

1 Introduction

Ever since Shannon's "Mathematical Theory of Communication" [40] first appeared, information theory has been of interest to psychologists and physiologists, to try to provide an explanation for the process of perception. Attneave [8] proposed that visual perception is the construction of an economical description of a scene from a very redundant initial representation. Barlow [9] suggested that lateral inhibition in the visual pathway may reduce the redundancy of an image, so information can be represented more efficiently. More recently, Linsker with his 'Infomax' principle [21, 22], Atick and Redlich [5, 6], and Plumbley and Fallside [30, 32] have continued with this approach with considerable success.

There have also been important advances in data compression techniques associated with principal component analysis. The original work of Oja [23] has now been extended to the analysis of higher-order statistics by Taylor and Coombes [45], and these techniques are presently attracting a great amount of interest. They have implications for the rapid feature analysis of patterns which are not linearly separable, and for higher order curve and surface fitting.

Finally there is a very general approach to the information-theoretic analysis of neural networks, using differential geometric ideas, by Amari [2]. This appears to be the deepest usage of information theory in neural networks, although only very preliminary results are available at present.

2 Information Theory and Learning

2.1 Information Theory

Let us briefly introduce some concepts from information theory. In particular, we are interested in the *entropy* of a random variable, and the *mutual information* between two

random variables [40].

Consider an experiment with n possible outcomes x , $1 \leq x \leq n$ with respective probabilities p_x . The *entropy*, $H(X)$, of this system is defined by the formula

$$H(X) = - \sum_{x=1}^n p_x \log p_x. \quad (1)$$

Entropy gives us a measure of the *uncertainty* in a system. If the probabilities of the various outcomes are equal, then the entropy will be maximised. On the other hand, if one of the outcomes is certain to happen, so its probability is one and the other outcomes have probability zero, then the entropy will be zero. (For this to work properly in the limit, we take $p_x \log p_x \rightarrow 0$ in the limit $p_x \rightarrow 0$.)

As an example, for a fair coin toss X , with $n = 2$ and $p_{\text{head}} = p_{\text{tail}} = 1/2$, we have

$$\begin{aligned} H(\Omega) &= - \left(\frac{1}{2} \log \frac{1}{2} + \frac{1}{2} \log \frac{1}{2} \right) \\ &= \log 2. \end{aligned}$$

However, suppose we have seen the coin land with e.g. heads uppermost: call this observation Y . We now know what the outcome of the toss is, so we have $p_{\text{head}} = 1$ and $p_{\text{tail}} = 0$. If we measure the entropy now, we get

$$\begin{aligned} H(X|Y) &= -(1 \log 1 + 0 \log 0) \\ &= 0 + 0 = 0 \end{aligned}$$

i.e. no uncertainty, since we know that the coin landed with heads uppermost. We use $H(X|Y)$ to denote the entropy in X given that we know Y . We shall see shortly that the *information* in the observation X about the coin toss Y is the difference between these two entropies.

More formally, if we have two random variables X and Y which take values x and y in the range $1 \leq x \leq n$ and $1 \leq y \leq m$, and have joint probabilities p_{xy} , their *joint entropy* is defined to be

$$H(X, Y) = - \sum_{x,y} p_{xy} \log p_{xy} \quad (2)$$

and the conditional entropy of X given Y is

$$H(X|Y) = - \sum_{x,y} p_{xy} \log \frac{p_{xy}}{p_y} = H(X, Y) - H(Y) \quad (3)$$

If X and Y are independent, then $p_{xy} = p_x p_y$, so

$$H(X|Y) = - \sum_{x,y} p_{xy} \log \frac{p_x p_y}{p_y} = - \sum_{x,y} p_x \log p_x = H(X).$$

The *mutual information* $I(X, Y)$ between X and Y is defined by the formula

$$I(X, Y) = H(X) - H(X|Y) = \sum_{x,y} p_{xy} \log \frac{p_{xy}}{p_x p_y} \quad (4)$$

which is zero if X and Y are independent. Note that $I(X, Y) = I(Y, X)$, so the information in Y about X is the same as the information in X about Y .

Another information-theoretic measure is the *I-divergence* or *cross-entropy* distortion measure D_{CE} from one probability distribution to another. The cross-entropy from a ‘true’ probability distribution p to an ‘estimated’ probability distribution q is defined by the expression

$$D_{CE}(p, q) = \sum_i p_i \log \frac{p_i}{q_i}.$$

which is zero if p and q are identical, but positive otherwise. Cross-entropy is asymmetric, so that in general $D_{CE}(p, q) \neq D_{CE}(q, p)$. By substituting $p_i = p_{xy}$ and $q_i = p_x p_y$ in the expression for D_{CE} above, it can be seen that mutual information between X and Y is the cross-entropy from the true probability distribution p_{xy} governing X and Y , to the probability distribution $p_x p_y$ which would govern them if they were independent (but had the same marginal distributions).

If the random variables X and Y are continuous, we use probability density $p(x, y)$ instead of probability distribution p_{xy} an integral instead of a sum in our expressions for entropy and mutual information:

$$H(X) = - \int_{-\infty}^{\infty} p(x) \log p(x) dx \quad (5)$$

$$I(X, Y) = \int_{-\infty}^{\infty} p(x, y) \log \frac{p(x, y)}{p(x)p(y)} dx dy. \quad (6)$$

For example, if X has an N -dimensional Gaussian probability density with covariance matrix C_X , its entropy will be given by

$$H(X) = \log \left((2\pi e)^{N/2} (\det C_X)^{1/2} \right). \quad (7)$$

For the special case where Y is a function of X with some additive noise Φ , i.e.

$$Y = f(X) + \Phi$$

then the mutual information between Y and X is given by

$$I(Y, X) = H(Y) - H(\Phi) \quad (8)$$

i.e. the entropy of the ‘signal plus noise’ less the entropy of the ‘noise’. If both Y and Φ are Gaussian with covariance matrices C_Y and C_Φ respectively, we get

$$I(Y, X) = \log \left((\det C_Y)^{1/2} \right) - \log \left((\det C_\Phi)^{1/2} \right).$$

2.2 Supervised Learning

2.2.1 BackProp with Mean Squared Error Distortion

One of the major uses for information theory has been in interpretation and guidance for *unsupervised* neural networks: networks which are not provided with a teacher or *target* output which they are to emulate. However, we can see how information relates

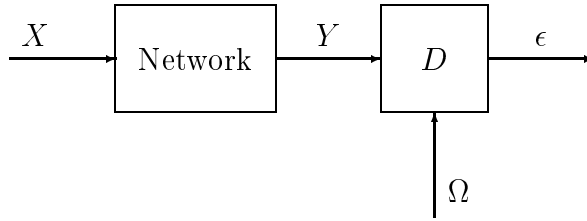


Figure 1: Supervised neural network with distortion measure D

to the more familiar supervised learning schemes, and in particular the use of Error Back Propagation (‘BackProp’) to minimize mean squared error (MSE) in a multi-layer perceptron (MLP).

Given a particular input X , supervised learning attempts to produce an output Y as ‘close’ as possible to some target output Ω , where closeness is measured with some distortion measure D (Fig. 1). Typically, the MSE distortion measure $\epsilon = D_{MSE}(\Omega, Y) = E(|Y - \Omega|^2)$ is used. BackProp calculates the derivative of ϵ with respect to each of the *weights* (the adjustable parameters) in the network, and adapts each weight to move ‘downhill’ in this error landscape towards the minimum distortion position [37].

Consider for the moment an alternative, information-theoretic, approach. Suppose we would like the output Y to contain as much information as possible about the target Ω . We can show that the two approaches are related, in that using BackProp to minimise MSE represents an approximation to maximising the information $I(\Omega, Y)$ in Y about Ω .

If we write $\Phi = Y - \Omega$, minimising MSE amounts to minimising the term

$$E(|\Phi|^2) = \text{Trace}(C_\Phi)$$

where $C_\Phi = E(\Phi\Phi^T)$ is the correlation matrix of Φ . On the other hand, maximising the information $I(\Omega, Y) = H(\Omega) - H(\Omega|Y)$ is equivalent to minimising the entropy of Ω given Y , $H(\Omega|Y)$, since $H(\Omega)$ is outside our control, and is independent of the changes made in the network. It is simple to show [28] that

$$H(\Omega|Y) = H(\Phi|Y) \leq H(\Phi)$$

and

$$H(\Phi) \leq \log\left((2\pi e)^{N/2}(\det C_\Phi)^{1/2}\right)$$

where N is the dimensionality of the output. Now, since

$$(\det C_\Phi)^{1/N} \leq \text{Trace}(C_\Phi)/N$$

by the arithmetic/geometric mean inequality, minimising the MSE $\text{Trace}(C_\Phi)$ will minimise an upper bound on $H(\Omega|Y)$, and thus maximise a lower bound on the information $I(\Omega, Y)$ in Y about Ω .

Due to the number of inequalities appearing in the process, this bound may not be particularly tight, especially when the error are ‘unbalanced’ in some way. See [28] for a discussion of some of the implications of this.

2.2.2 Cross-Entropy Distortion

Other distortion measures are possible in place of MSE. In particular, the information-theoretic *cross-entropy distortion* [43] has been suggested. The use of this distortion measure relies on the both the output and the target representing probability distributions directly. Thus a single output must be restricted to lie between 0 and 1: the sigmoid function $\sigma(u) = 1/(1 + \exp(-u))$ as the final stage in the output unit will conveniently achieve this.

With more than one output unit, they can either be treated as independent features, in which case the total distortion measure is given by

$$\epsilon = \sum_i D_{CE}(\Omega_i, Y_i) = \sum_i \Omega_i \log \frac{\Omega_i}{Y_i} + (1 - \Omega_i) \log \frac{1 - \Omega_i}{1 - Y_i}$$

or they can be treated as a single probability distribution, in which case the total distortion measure is given by

$$\epsilon = D_{CE}(\Omega, Y) = \sum_i \Omega_i \log \frac{\Omega_i}{Y_i}.$$

In the latter case, the additional restriction that the outputs must sum to one must be enforced, to ensure the outputs form a true probability distribution. Bridle's *Softmax* function [13]

$$y_i = \frac{\exp(u_i)}{\sum_j \exp(u_j)}$$

can be used to enforce this condition.

2.3 Unsupervised Learning: Direct Approaches

2.3.1 G-Max: Maximising Cross-Entropy

An early unsupervised learning algorithm derived from an information-theoretic approach was the *G-Maximization* algorithm suggested by Pearlmutter and Hinton [27].

They used single binary neuron with N binary inputs x_i and corresponding weights w_i , $i = 1, \dots, N$ and a single output y , which can be either '0' or '1'. The probability that the output state is '1' is

$$p_1 = 1 - p_0 = \sigma \left(\sum_{i=1}^N w_i x_i \right) \quad (9)$$

where the $\sigma(\cdot)$ is the logistic sigmoid function

$$\sigma(x) = \frac{1}{1 + \exp(-x)}. \quad (10)$$

They wanted the unit to discover regularities in the input patterns, i.e. patterns that occur more often than would be expected if the activities of the individual input lines were assumed to be independent.

To this end, they chose as their target function the cross-entropy distortion

$$G = p_0 \log \frac{p_0}{q_0} + p_1 \log \frac{p_1}{q_1}$$

between the true output probability distribution p , and an ‘independent input activity’ probability distribution q of the output.

The algorithm is run in two phases. The first phase is run with with the real input data to accumulate the output statistics which define the output probability distribution p_y . The probabilities of each of the inputs is also measured for use in the second phase. In the second phase, the inputs x_i are set randomly to ‘1’ or ‘0’ with the same probabilities that they had in the first phase, but independently of each other. The output statistics are measured in this second phase: this defines the probability distribution q_y . Since the cross-entropy measure used can be differentiated with respect to each of the weights w_i , a steepest descent search can then be used to modify the weights to maximise the G measure.

When this algorithm was tested on a 10×10 ‘retina’ exposed to randomly positioned and oriented edges, the unit typically developed a centre-on surround-off response (or vice versa). This type of response is typical of retinal ganglion cells. Unfortunately, it was rather awkward to extend the G-max algorithm to more than one output unit, since some additional mechanism has to be employed to prevent all the units learning to respond to the same features of the input.

2.3.2 I-Max: Maximising Mutual Information between Output Units

More recently, Becker and Hinton [11] suggested that the information *between* output units could be used as the objective function for an unsupervised learning technique (Fig. 2). In a visual system, this scheme would attempt to extract higher-order features of the visual scene which are coherent over space (or time). For example, if two networks each produce a single output from two separate but neighbouring *patches* of a retina, the objective of their algorithm is to maximise the mutual information $I(Y_1, Y_2)$ between these two outputs. A steepest-ascent procedure can be used to find the maximum of this mutual information function, both for binary- and real-valued units.

One application of this principle is the extraction of depth from random-dot stereograms. Nearby patches in an image usually view objects of a similar depth, so if the mutual information between neighbouring patches is to be maximised, the outputs from both output units y_1 and y_2 should correspond to the information which is common between the patches, rather than that which is different. In other words the outputs should both learn to extract the common depth information rather than any other property of the random dot patterns.

For binary-valued units, with each unit similar to that used by the G-max scheme described above, the mutual information $I(Y_1, Y_2)$ between the two output units is

$$I(Y_1, Y_2) = H(Y_1) + H(Y_2) - H(Y_1, Y_2) \quad (11)$$

so if the probability distributions $P(y_1)$, $P(y_2)$ and $P(y_1, y_2)$ are measured, this mutual information can be calculated directly. Of course, it is sufficient to measure $P(y_1, y_2)$ only,

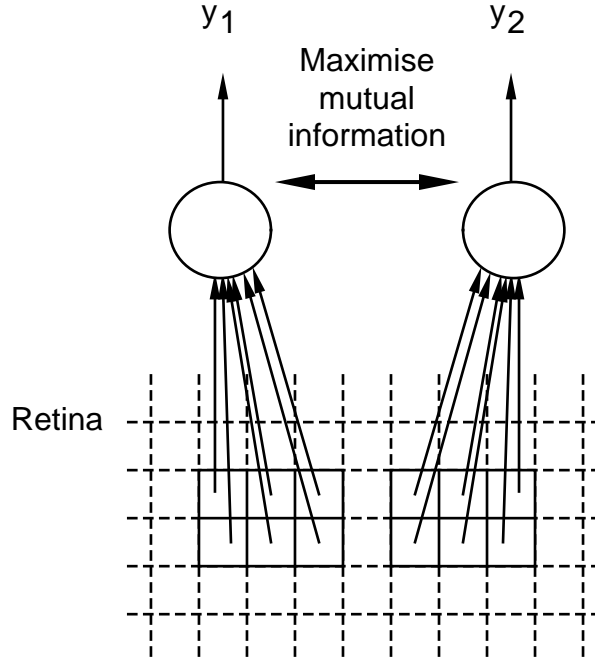


Figure 2: Maximising Mutual Information between Output Units

since

$$P(y_1) = \sum_{y_2=0}^1 P(y_1, y_2)$$

and similarly for $P(y_2)$. The derivative of (11) can be taken with respect to the weights in the network for each different input pattern, so enabling the steepest-ascent procedure to be used.

For real-valued outputs it would be impossible to measure the entire probability distribution $P(Y_1, Y_2)$, so instead it is assumed that the two outputs have a Gaussian probability distribution, and that one of the outputs is a noisy version of the other, with independent additive Gaussian noise. In this case, the information $I(Y_1, Y_2)$ between the two outputs can be calculated from the variances of one of the outputs (the ‘signal’) and the variance of the difference between the outputs (the ‘noise’) as

$$I(Y_1, Y_2) = \frac{1}{2} \log \frac{\sigma_{Y_1}^2}{\sigma_{Y_1 - Y_2}^2} \quad (12)$$

where $\sigma_{Y_1}^2$ is the variance of the output of the first unit, and $\sigma_{Y_1 - Y_2}^2$ is the variance of the difference between the two outputs.

If we accumulate the mean and variance of both Y_1 and $Y_1 - Y_2$, it is possible to find the derivative of (12) for each input pattern, with respect to each weight value. Thus the weights in the network can be updated in a steepest-ascent procedure to maximise $I(Y_1, Y_2)$, or at least the approximation to $I(Y_1, Y_2)$ given by (12).

Becker and Hinton found that unsupervised networks using this principle could learn to extract depth information from random-dot stereograms with either binary- or continuous-valued shifts, as appropriate for the type of outputs used, although in some cases it helped to force the units to share weight values, further enforcing the idea that the units should calculate the same function of the input. They generalised their scheme to allow networks with hidden layers, and also to allow multiple output units, with each unit maximising the mutual information between itself and a value predicted from its neighbouring units. This latter scheme allowed the system to discover an interpolation for curved surfaces.

Subsequently, Zemel and Hinton [49] generalised this procedure to allow for more than one output per module. In their network, they use 4 outputs per unit to attempt identify four degrees of freedom in 2-dimensional objects: horizontal and vertical position, orientation, and size. The mutual information measure is now

$$I(Y_1, Y_2) = \frac{1}{2} \log \frac{\det(C_{Y_1+Y_2})}{\det(C_{Y_1-Y_2})} \quad (13)$$

where $C_{Y_1+Y_2}$ is the covariance matrix of the sum of Y_1 and Y_2 (now vectors, of course), and $C_{Y_1-Y_2}$ is the covariance matrix of their difference. By measuring the degree of mismatch between the two representations, the model can tell roughly how much one half of an object is perturbed away from the position and orientation which is consistent with the other half of the object.

3 Predictive Coding

3.1 Redundancy and Visual Scenes

Soon after the introduction of his Information Theory, Shannon [42] used a prediction ‘game’ to investigate the information content of English text. He showed that the entropy of text is just over 1 bit per letter, much less than the 4.7 bits ($= \log_2 26$) which would be needed if the 26 letters of english text were independent and occurred with equal frequency. This is itself much less than the 8 bits per letter used to represent text in a typical computer. This shows that there is a significant amount of *redundancy* in a typical passage of text: redundancy is the ratio of the information capacity used to represent a signal to the information actually present in the signal. If a communication system could take advantage of this redundancy, text could be transmitted more economically than before, since each bit used costs a certain amount of resources to transmit it.

Attneave [8] suggested that the process of perception may be creating an economical description of sensory inputs, by reducing the redundancy in the stimulus. As a concrete example, he considered a picture of an ink bottle on a desk, similar to Fig. 3. If a subject is asked to predict the colours in the picture, scanning along the rows from left to right, taking successive rows down the picture, most mistakes are made at the edges and corners in the picture. Applying this ‘Shannon game’ on the scene suggests that most information is concentrated around the edges and corners of a scene, which fits well with psychological suggestions that these features are important in a visual scene. It is also strengthened by the later observations by Hubel and Wiesel [20] of ‘line detector’ cells in the visual cortex.

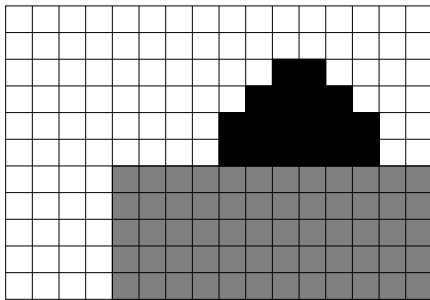


Figure 3: A redundant visual stimulus (After Attneave [8])

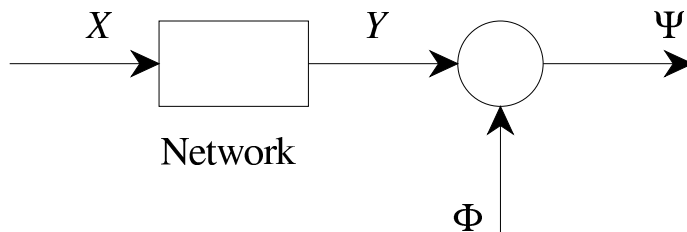


Figure 4: Network output signal Y is corrupted by noise Φ .

Using this redundancy reduction hypothesis, Barlow [9] suggested that *lateral inhibition* may be producing an economical description of the scene. By subtracting a weighted sum of surrounding sensor responses, the resulting activations are more independent (less redundant) than the original stimulus, while the total information content remains unchanged. Using lateral inhibition, the visual scene can be represented more economically.

3.2 Whitening filters and predictive coding

The original justification for the redundancy reduction hypothesis considered for example signals with a finite number of levels, such as *just noticeable differences* (JNDs). However, a related approach was formulated by Shannon for economical transmission of signals through a noisy channel [41].

Consider a system (a neural network in this case) which is transmitting its real-valued output signal Y through a channel where it will be corrupted by additive noise Φ (Fig. 4). If there were no restrictions on Y , we could simply amplify it until we had overcome as much of the noise as we like. However, suppose that there is a power cost

$$S_T = \int_0^B S(f)^2 df \quad (14)$$

associated with transmitting the signal Y through the channel, where $S(f)$ is the power

spectral density of Y (the ‘signal’) at frequency f , and B is a bandwidth limit. Then we wish to maximise the transmitted information (assuming both signal and noise are Gaussian)

$$I(\Psi, X) = \int_0^B \log \frac{S(f) + N(f)}{N(f)} df \quad (15)$$

where $N(f)$ is the power spectral density of Φ (the ‘noise’), for a given power cost S_T .

Using the Lagrange multiplier technique, we attempt to maximise

$$J = I(\Psi, X) - \lambda S_T = \int_0^B \left(\log \frac{S(f) + N(f)}{N(f)} - \lambda S(f) \right) df \quad (16)$$

as a function of $S(f)$ for every $0 \leq f \leq B$. This is the case when

$$S(f) + N(f) = \text{constant} \quad (17)$$

so if $N(f)$ is *white* noise, i.e. the power spectral density is uniform (or *flat*), the power spectral density $S(f)$ should also be uniform [41]. A filter which performs this flattening is called a *whitening* filter. It is well known that a signal with flat power spectral density has an autocorrelation function $R_y y(\tau) = E(Y(t), Y(t + \tau))$ which is proportional to a delta function $\delta(\tau)$. In other words, the time-varying output signal $Y(t_1)$ at any time t_1 should be uncorrelated with the signal $Y(t_2)$ at any other time $t_2 \neq t_1$.

This approach leads to an equivalent condition where the signal is represented over a regular grid of units in *space* instead of *time*, such as a signal from a grid of visual receptors. In this case the signal should be transformed so that the outputs of the transform are uncorrelated from each other.

One way of achieving this *decorrelation* of outputs is to use the technique of *linear predictive coding* (See e.g [26]). To see how this works, consider a time-sequence of input values x_i, x_{i-1}, \dots where x_i is the most recent value. We can form the least mean square (LMS) linear prediction \hat{x}_i of x_i from the previous values as follows:

$$\hat{x}_i = a_1 x_{i-1} + a_2 x_{i-2} + \dots \quad (18)$$

where the coefficients a_j are chosen to minimise the expected squared error

$$\epsilon = E (x_i - \hat{x}_i)^2. \quad (19)$$

Taking the derivative of this with respect to each coefficient a_j , the condition for this minimum is

$$E [(x_i - \hat{x}_i) x_{i-j}] = 0 \quad (20)$$

for all $j > 0$. If we take the residual $y_i = x_i - \hat{x}_i$ to be the output of our transform, the LMS linear prediction gives us

$$E [y_i x_{i-j}] = 0 \quad (21)$$

for all $j > 0$, and therefore

$$E [y_i y_k] = 0 \quad (22)$$

for all $k < i$, since

$$y_k = x_k - (a_1 x_{k-1} + a_2 x_{k-2} + \dots).$$

Thus linear predictive coding has given us the uncorrelated outputs we need.

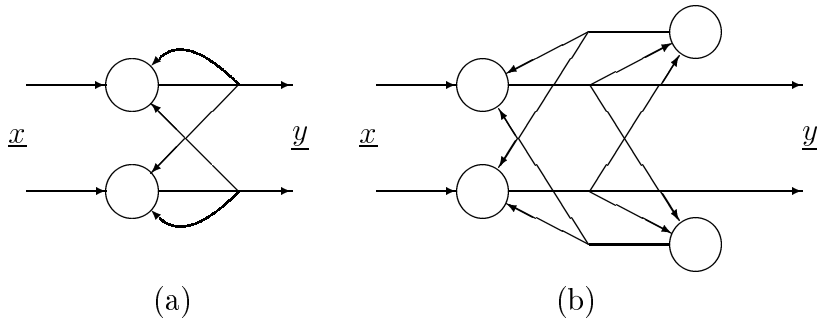


Figure 5: Linear decorrelating networks ($M = 2$).

3.3 Local Decorrelating Algorithms

One of the early suggestions for learning in neural networks was Hebb's [19] principle, that the effectiveness of the connection between two units should be increased when they are both active at the same time. This has been used as the basis of a number of artificial neural network learning algorithms, so-called *Hebbian* algorithms, which increase a connection weight in proportion to the product of the unit activations at each end of the connection. If the connection weight *decreases* (or increases its *inhibition*) in proportion to the product of the unit activations, this is called *anti-Hebbian* learning.

A number of anti-Hebbian algorithms have been proposed to perform decorrelation of output units. For example, Barlow and Földiák [10] have suggested a network with linear recurrent lateral inhibitory connections (Fig. 5(a)) with an anti-Hebbian local learning algorithm.

In vector notation, we have an M -dimensional input vector \underline{x} , an M -dimensional output vector \underline{y} , and an $M \times M$ lateral connection matrix V . For a fixed input, the lateral connections cause the output values to evolve according to the expression

$$(y_i)_{t+1} = x_i - \sum_j v_{ij}(y_j)_t \quad \text{i.e.} \quad \underline{y}_{t+1} = \underline{x} - V\underline{y}_t \quad (23)$$

at time step t , which settles to an equilibrium when $\underline{y} = \underline{x} - V\underline{y}$, which we can write as

$$\underline{y} = (I_M + V)^{-1}\underline{x} \quad (24)$$

provided $(I_M + V)$ is positive definite. We assume that this settling happens virtually instantaneously. The matrix V is assumed to be symmetrical so that the inhibition from unit i to unit j is the same as the inhibition from j to i , and for the moment we assume that there are no connections from a unit back to itself, so the diagonal entries of V are zero.

Barlow and Földiák [10] suggested that for each input \underline{x} , the weights v_{ij} between different units should be altered by a small change

$$\Delta v_{ij} = \eta y_i y_j \quad i \neq j \quad (25)$$

where η is a small update factor. In vector notation this is

$$\Delta V = \eta \text{offdiag}(\underline{y} \underline{y}^T) \quad (26)$$

since the diagonal entries of V remain fixed at zero. This algorithm converges when $E(y_i y_j) = 0$ for all $i \neq j$, and thus causes the outputs to become decorrelated [10].

Atick and Redlich [7] considered a similar network, but with an integrating output $dy/dt = \underline{x} - V\underline{y}$ leading to $\underline{y} = V^{-1}\underline{x}$ when it has settled. They show that a similar algorithm for the lateral inhibitory connections between different output units leads to decorrelated outputs, while reducing an information-theoretic redundancy measure.

The algorithms considered so far simply decorrelate their outputs, but ignore what happens to the diagonal entries of the covariance matrix. For a signal with statistics which are position-independent, such as images on a regularly-spaced grid of receptors, we can consider the problem in the spatial frequency domain. Decorrelation is optimal, as we have seen above, and the variance of all the outputs will happen to be equal.

If we do not have position-independent statistics, we can go back to the power-limited noisy channel argument, but use the actual output covariance matrix instead of working in the frequency domain. For small output noise, we can express the transmitted information as

$$I(\Psi, X) = 1/2 \log \det C_Y - 1/2 \log \det C_\Phi \quad (27)$$

and the power cost as

$$S_T = \text{Trace}(C_Y). \quad (28)$$

Using the Lagrange multiplier technique again, we wish to maximise

$$J = I(\Psi, X) - 1/2\lambda S_T \quad (29)$$

which leads to the condition [30]

$$C_Y = 1/\lambda I_M. \quad (30)$$

In other words, not only should the outputs be decorrelated, but they should all have the same variance, $E(y_i^2) = 1/\lambda$.

The Barlow and Földiák [10] algorithm can be modified to achieve this, if self-inhibitory connections from each unit back to itself are allowed [30]. The algorithm becomes

$$\Delta v_{ij} = \eta y_i y_j - (1/\lambda) \delta_{ij} \quad \text{i.e.} \quad \Delta V = \eta(\underline{y} \underline{y}^T - (1/\lambda) I_M) \quad (31)$$

which monotonically increases J as it progresses.

This is perhaps a little awkward, since the self-inhibitory connections have a different update algorithm to the normal lateral inhibitory connections. As an alternative, a linear network with inhibitory interneurons (Fig. 5(b)) can be used. After an initial transient, this network settles to

$$\underline{y} = \underline{x} - V\underline{z} \quad \text{and} \quad \underline{z} = V^T \underline{y} \quad (32)$$

i.e.

$$\underline{y} = (I + VV^T)^{-1} \underline{x} \quad (33)$$

where v_{ij} is now the weight of the excitatory (positive) connection from y_i to z_j , and also the weight of the inhibitory (negative) connection back from z_j to y_i .

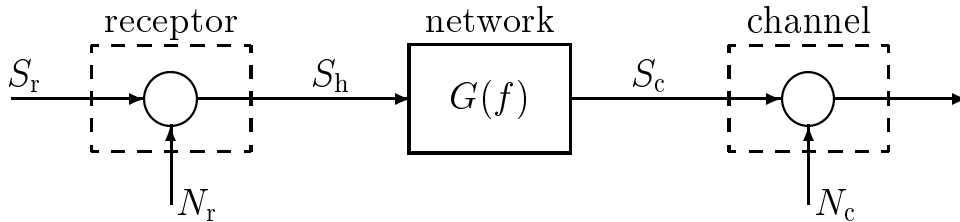


Figure 6: The signal is corrupted by both input noise before the network (or filter), and output noise after the network.

Suppose that the weights in this network are updated according to the algorithm

$$\Delta v_{ij} = \eta(y_i z_j - 1/\lambda v_{ij}) \quad (34)$$

which is a Hebbian (or anti-Hebbian) algorithm with weight decay, and is

$$\Delta V = \eta(C_Y - 1/\lambda I_M)V \quad (35)$$

in vector notation. Then the algorithm will converge when $C_Y = 1/\lambda I_M$, which is precisely what we need to maximise J . In fact, this algorithm will also monotonically increase J as it progresses.

This network suggests that inhibitory interneurons, which are found in many places in sensory systems, may be performing some sort of decorrelation task. Not only does the condition of decorrelated equal variance output optimize information transmission for a given power cost, but it can be achieved by various biologically-plausible Hebb-like algorithms.

3.4 Optimal filtering

Srinivasan, Laughlin and Dubs [44] suggested that predictive coding is used in the fly's visual system to perform decorrelation. They compared measurements from the fly with theoretical results based on predictive coding of typical scenes, and found reasonably good agreement at both high and low light levels. However, they did find a slight mismatch, in that the surrounding inhibition was a little more diffuse than the theory predicted.

A possible problem with the original predictive coding approach is that only the *output* noise is considered in the calculation of information: the input noise is assumed to be part of the signal. At low light levels, where the input noise is a significant proportion of the input, the noise is simply considered to change the input power spectrum, making it flatter [44]. This assumption means that the predictive coding is an approximation to a true optimal filter: the approximation is likely to be worse for either high frequency components, where the original signal power spectral density is small, or for low light conditions, where all signal components are small.

In fact, it is possible to analyse the system for both input *and* output noise (Fig. 6). We can take a similar Lagrange multiplier approach as before, and attempt to maximise transmitted information for a fixed power cost. Omitting the details, we get the following

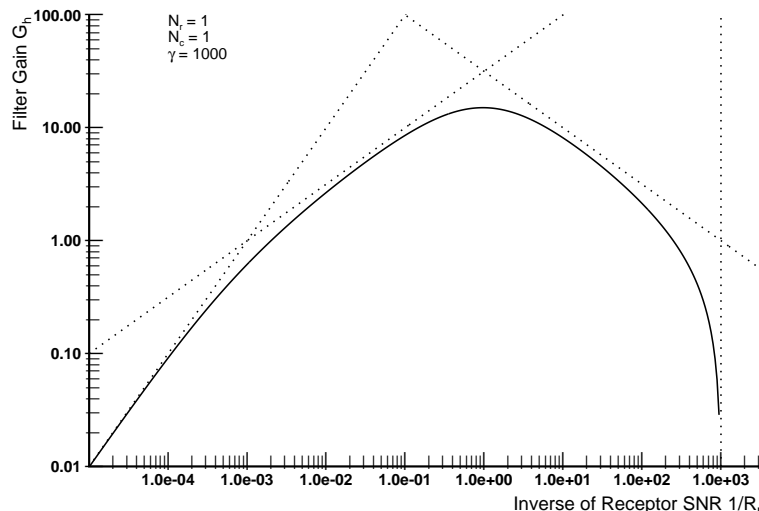


Figure 7: Typical optimal filter solution, for equal white receptor and channel noise.

quadratic equation to solve for this optimal filter at every frequency f [33]

$$(R_c + R_r + 1)(R_c + 1) - \frac{\gamma}{N_c} R_r = 0 \quad (36)$$

where R_c is the channel signal to noise power spectral density ratio S_c/N_c , and R_r is the receptor signal to noise power spectral density ratio S_r/N_r , and γ is a Lagrange multiplier which determines the particular optimal curve to be used.

This leads to a non-zero filter gain G_h whenever $R_r > [(\gamma/N_c) - 1]^{-1}$. For constant N_c (corresponding to a flat channel noise spectrum) there is therefore a certain cut-off point below which noisy input signals will be suppressed. Fig. 7 shows a typical optimal solution, together with its asymptotes. Plumbley [29, 31] has been investigating modifications to the decorrelating algorithms mentioned above which may learn to approximate this optimal filtering behaviour.

Atick and Redlich [5] used a similar optimal filtering approach in their consideration of the mammalian visual system, minimising redundancy for fixed information rather than maximising information for fixed power. They compared their theory with the spatiotemporal response of the human visual system, and found a very good match [4]. These results suggest very strongly that economical transmission of information is a major factor in the organization of the visual system, and perhaps other sensory systems as well.

4 Principal Component Analysis and Infomax

Principal component analysis (PCA) is widely used for dimension reduction in data analysis and pre-processing, and is used under a variety of names such as the (discrete)

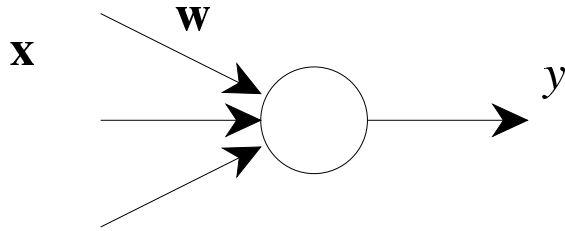


Figure 8: The Oja Neuron.

Karhunen Loève Transform (KLT), factor analysis, or the Hotelling Transform in image processing. Its primary use is to provide a reduction in the number of parameters used to represent a quantity, while minimising the error introduced by so doing. In the case of PCA, a purely linear transform is used to reduce the dimensionality of the data, and it is the transform which minimises the mean squared reconstruction error. This is the error which we get if we transform the output y back into the input domain to try to reconstruct the input \underline{x} so that the error is minimised.

Linsker’s principal of maximum information preservation, “Infomax”, can be applied to a number of different forms of neural network. The analysis, however, is much simpler when we are dealing with simple networks, such as binary or linear systems. It is instructive to look at the linear case of PCA in some detail, since much effort in other fields has been directed at linear systems. We should not be too surprised to find a neural network system which can perform KLT and PCA.

From one point of view, these conventional data processing methods let us know what to expect from a linear unsupervised neural network. However, the information theoretic approach to the neural network system can help us with the conventional data processing methods. In particular, we shall find that a dilemma in the use of PCA, known as the *scaling problem*, can be clarified with the help of information theory.

4.1 The Linear Neuron

Arguably the simplest form of unsupervised neural network is an N -input, single-output linear neuron (Fig. 8). Its output response y is simply the sum of the inputs x_i multiplied by their respective weights w_i , i.e.

$$y = \sum_{i=1}^N w_i x_i \quad (37)$$

or, in vector notation,

$$y = \underline{w}^T \underline{x} \quad (38)$$

where $\underline{w} = [w_1, \dots, w_N]^T$ and $\underline{x} = [x_1, \dots, x_N]^T$ are column vectors. The output y is thus the dot product $\underline{x} \cdot \underline{w}$ of the input \underline{x} with the weight vector \underline{w} . If \underline{w} is a unit vector, i.e.

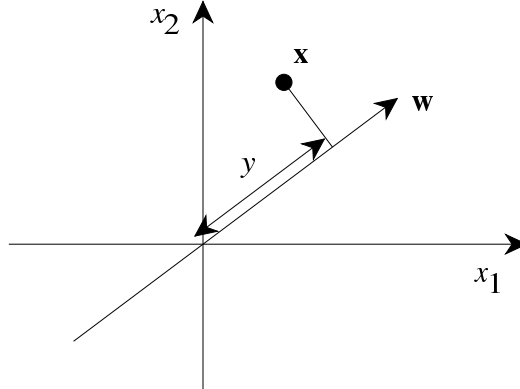


Figure 9: Output y as a component of \underline{x} , with unit weight vector.

$|\underline{w}| = 1$, y is the component of \underline{x} in the direction of \underline{w} (Fig. 9).

We thus have a simple neuron which finds the component of the input \underline{x} in a particular direction. We would now like to have a neural network learning rule for this system, which will modify the weight vector depending on the inputs which are presented to the neuron.

4.2 The Oja Principle Component Finder

A very simple form of Hebbian learning rule would be to update each weight by the product of the activations of the units at either end of the weight. For the single linear neuron (Fig. 8), this would result in a learning algorithm of the form

$$\Delta w_i = \eta x_i y \quad (39)$$

or in vector notation

$$\Delta \underline{w} = \eta \underline{x} y. \quad (40)$$

Unfortunately, this learning algorithm alone would cause any weight to increase without bound, so some modification has to be used to prevent the weights from becoming too large.

One possible solution is to limit the absolute values that each weight w_i can take [46], while another is to renormalise the weight vector \underline{w} to have unit length after each update [23]. An alternative is to use a weight decay term which causes the weight vector to tend to have unit length as the algorithm progresses, without explicitly normalising it. To see how this works, consider the following weight update algorithm, due to Oja [23]:

$$\begin{aligned} \Delta \underline{w} &= \eta (\underline{x} y - \underline{w} y^2) \\ &= \eta (\underline{x} \underline{x}^T \underline{w} - \underline{w} (\underline{w}^T \underline{x} \underline{x}^T \underline{w})). \end{aligned} \quad (41)$$

When the weight vector is small, the update algorithm is dominated by the first term on the right hand side, which causes the weight to increase as for the unmodified Hebbian

algorithm. However, as the weight vector increases, the second term (the ‘weight decay’ term) on the right hand side becomes more significant, and this tends to keep the weight vector from becoming too large.

To find the convergence conditions of the Oja algorithm, let us consider the average weight update over some number of input presentations. We shall assume the input vectors \underline{x} have zero mean, and we shall also assume that the weight update factor is so small that the weight itself can be regarded as approximately constant over this number of presentations. Thus the mean update is given by

$$\begin{aligned}\overline{\Delta \underline{w}} &= \eta \left(\overline{\underline{x} \underline{x}^T \underline{w}} - \overline{\underline{w} \underline{w}^T \underline{x} \underline{x}^T \underline{w}} \right) \\ &\approx \eta \left(\overline{\underline{x} \underline{x}^T \underline{w}} - \underline{w} \left(\overline{\underline{w}^T \underline{x} \underline{x}^T \underline{w}} \right) \right) \\ &\approx \eta (C_x \underline{w} - \underline{w} \lambda)\end{aligned}\tag{42}$$

where $\lambda = \underline{w}^T C_x \underline{w}$, and $C_x = E(\underline{x} \underline{x}^T)$ is the covariance matrix of the input data \underline{x} .

When the algorithm has converged, the average value of $\overline{\Delta \underline{w}}$ will be zero, so we have

$$C_x \underline{w} = \underline{w} \lambda\tag{43}$$

i.e. the weight vector \underline{w} is an eigenvector of the input covariance matrix C_x . A perturbation analysis confirms that the only stable solution is for \underline{w} to be the *principal* eigenvector of C_x . To find the eventual length of \underline{w} we simply substitute (43) into the expression for λ , and we find that

$$\lambda = \underline{w}^T (C_x \underline{w}) = \underline{w}^T (\underline{w} \lambda)\tag{44}$$

i.e. provided λ is non-zero, $\underline{w}^T \underline{w} = 1$ so the final weight vector has unit length.

We have therefore seen that as the Oja algorithm progresses, the weight vector will converge to the normalised principal eigenvector of the input covariance matrix (or its negative) [23]. The component of the input which is extracted by this neuron, to be transmitted through its output y , is called the *principal component* of the input, and is the component with largest variance for any unit length weight vector.

4.3 Reconstruction Error

For our single-output system, suppose we wish to find the best estimate $\hat{\underline{x}}$ of the input \underline{x} from the single output $y = \underline{w}^T \underline{x}$. We form our reconstruction using the vector \underline{u} as follows:

$$\hat{\underline{x}} = \underline{u} y\tag{45}$$

where \underline{u} is to be adjusted to minimise the mean squared error

$$\epsilon = E \left[|\underline{x} - \hat{\underline{x}}|^2 \right] = E \left[\left| (I - \underline{u} \underline{w}^T) \underline{x} \right|^2 \right].\tag{46}$$

If we minimise ϵ with respect to \underline{u} for a given weight vector \underline{w} , we get a minimum for ϵ at

$$\underline{u}_{\underline{w}} = \arg \min_{\underline{u}} (\epsilon) = \frac{C_x \underline{w}}{\underline{w}^T C_x \underline{w}}\tag{47}$$

where $C_x = E[\underline{x}\underline{x}^T]$ as before (assuming that \underline{x} has zero mean). Our best estimate of \underline{x} is then given by

$$\hat{\underline{x}}_{\underline{w}} = \underline{u}_{\underline{w}}\underline{y} = \frac{C_x \underline{w} \underline{w}^T}{\underline{w}^T C_x \underline{w}} \underline{x} = P \underline{x} \quad (48)$$

where the matrix

$$Q = \frac{C_x \underline{w} \underline{w}^T}{\underline{w}^T C_x \underline{w}} \quad (49)$$

is a *projection* operator, a matrix operator which has the property that $Q^2 = Q$. This means that the best estimate of the reconstruction vector $\hat{\underline{x}}_{\underline{w}}$ from the output $\hat{\underline{y}}_{\underline{w}} = \underline{w} \hat{\underline{x}}_{\underline{w}}$ is $\hat{\underline{x}}_{\underline{w}}$ itself. Once this is established, it is possible to minimise ϵ with respect to the original weight vector \underline{w} . Provided the input covariance matrix C_x is positive definite, this minimum occurs when the weight vector is the principal eigenvector of C_x . Thus PCA minimises mean squared reconstruction error.

4.4 The Scaling Problem

Users of PCA are sometimes presented with a problem known as the *scaling problem*. The result of PCA, and related transforms such as KLT, is dependent on the scaling of the individual input components x_i . When all of the input components come from a related source, such as light level receptors in an image processing system, then it is obvious that all the inputs should have the same scaling. However, when different inputs represent unrelated quantities, then the relative scaling which each input should be given is not so apparent. As an extreme example of this problem, consider two uncorrelated inputs which initially have equal variance. Whichever input has the largest scaling will become the principal component. While this extreme situation is unusual, the scaling problem does cause PCA to produce scaling-dependent results, which is rather unsatisfactory.

Typically, this dilemma is solved by scaling each input to have the same variance as each other [47]. However, there is also a related problem which arises when multiple readings of the same quantity are available. These readings can either be averaged to form a single reading, or they can be used individually as separate inputs. If same-variance scaling is used, these two options again produce inconsistent results.

Thus although PCA is used in many problem areas, these scaling problems may lead us not to trust it to give us a consistent result in an unsupervised learning system.

4.5 Information Maximization

We have seen that the Oja neuron learns to perform a principal component analysis of its input, but that principal component analysis itself suffers from an inconsistency problem when the scaling of the input components is not well defined. In order to gain some insight to this problem, we shall apply Linsker's *Infomax* principle [21] to this situation.

Consider a system with input X and output Y . Linsker's Infomax principle states that a network should adjust itself so that the information $I(X, Y)$ transmitted to its output Y about its input X should be maximised. This is equivalent to the information in the input X about the output Y , since $I(X, Y)$ is symmetric in X and Y .

However, if Y is a noiseless function of X , as is the case for our linear neuron

$$y = \underline{w}^T \underline{x}$$

then there will be an *infinite* amount of information in the output Y about X , because Y represents X infinitely accurately. In order to proceed, we must assume that the input contains some noise $\underline{\phi}$ which prevents any of the input from being measured too accurately.

Consider the case where the input signal \underline{x} is zero mean Gaussian with covariance matrix C_x , and the noise $\underline{\phi}$ is also zero mean Gaussian, with covariance matrix $C_\phi = \sigma^2 I$, so that the noise on each input component is uncorrelated with equal variance. The output of the neuron is then the weighted sum

$$y = \underline{w}^T (\underline{x} + \underline{\phi}) = \underline{w}^T \underline{x} + \underline{w}^T \underline{\phi}. \quad (50)$$

Writing down the information in the output y about the input signal \underline{x} , we get

$$I(Y; X) = \frac{1}{2} \log \frac{S + N}{N} \quad (51)$$

where

$$S = E(|\underline{w}^T \underline{x}|^2) = \underline{w}^T C_x \underline{w}$$

and

$$N = E(|\underline{w}^T \underline{\phi}|^2) = \sigma^2 |\underline{w}|^2.$$

Since (51) is monotonically increasing in S/N , $I(X, Y)$ is maximised when \underline{w} is the principal eigenvector of C_x , i.e. it is the principal component of the input.

This is the same condition for minimising the mean squared reconstruction error considered above, but now we have an explicit condition on the noise on the input. The condition is that the noise on each input should be uncorrelated, and each input should have the same noise variance.

The scaling problem of principal component analysis is now changed to one of guessing the noise on each of the inputs, and scaling them so that this noise is approximately equal. The standard approach of scaling all inputs so that their signal variance is equal is therefore equivalent to assuming that the signal to noise ratio of all inputs is equal [28].

Of course, the assumptions that the input signal and noise are Gaussian and zero mean are very strong, but can be relaxed somewhat if *information loss* is considered rather than *transmitted information*. However, the result in each case is the same: the Oja algorithm, which finds the principal component of the input, maximises information capacity on condition that the noise on the input components is equal.

4.6 Multi-dimensional PCA

There are a number of algorithms which extend Oja's algorithm to more than one output neuron. For these we need an output vector \underline{y} and a weight matrix W , such that

$$\underline{y} = W^T \underline{x}. \quad (52)$$

If the Oja algorithm was used for each output neuron with no modification, each would find the same principal component of the input data. Some mechanism must be used to force the outputs to learn something different from each other.

One possibility is to use a lateral inhibition network between the output neurons, which forces their outputs to be decorrelated [16]. An alternative is to modify the weight decay term of the Oja algorithm: this approach is used by William’s Symmetric Error Correction (SEC) algorithm [48], Oja and Karhunen’s M -output PCA algorithm [24], and Sanger’s Generalised Hebbian Algorithm (GHA) [39].

In fact, these algorithms have much in common. Although the weight vectors themselves have different algorithms, the subspace defined by the orthogonal projection

$$P = W(W^T W)^{-1}W^T$$

which is the subspace spanned by the weight vectors to each of the outputs, moves in exactly the same way for each of these algorithms [28]. Since this subspace, rather than the weight vector itself, determines the change in information transmitted through the network, these three algorithms tend to increase the transmitted information in exactly the same way. All three lead to a set of weight vectors which spans the same space as the largest principal eigenvectors of the input covariance matrix, which is sufficient to maximise the transmitted information (under the equal noise conditions which we outlined above) [28].

The three algorithms differ only in the behaviour of the weight vectors themselves. In particular, the SEC algorithm [48] leads to weight vectors which are orthogonal and unit length, but have no particular relationship to the eigenvectors of the input covariance matrix. Oja and Karhunen’s algorithm [24] uses a Gramm-Schmidt Orthogonalisation (GSO) approach to find the principal components themselves, in order. Sanger’s algorithm [39] uses GSO in a slightly different way, but also finds the principal components in order.

4.7 Discussion

We have seen that linear neurons, with a modified form of Hebbian learning algorithm, can learn to find the principal component or principal subspace of its input data. We have also seen that this principal subspace maximises information capacity of the system, under the condition that the input components have uncorrelated, independent, equal-variance Gaussian noise on all of the components.

When we perform principal component analysis in practice, we also tend to use a set of output which are decorrelated, or at the very least not highly correlated with each other. The algorithms considered here also do this, but this does not seem to be required to maximise information capacity. We should only have to find the principal subspace of the input data: correlation between the output components should be irrelevant.

The puzzle here arises because we have neglected noise which may occur *after* the network which we are currently considering. As we have already seen in §3.2, decorrelated outputs tend to be better protected against later noise (which may be added noise or calculation errors) than outputs which are highly correlated [10]. One of the authors has recently investigated combination algorithms which may be suitable with both input and

output noise [29, 31]. It may be that real perceptual systems are able to take account of both noise sources at the same time.

5 Non-linear Principal Component Analysis

5.1 Introduction

In the previous section various algorithms were suggested so as to achieve principal component analysis on a set of input patterns and thereby allow for dimension reduction and data compression. At the same time it was pointed out in §4.5 that such analysis leads to maximisation of information capacity. All of this analysis was performed in the context of linear neurons, with outputs given by (37) or (38). The statistics of the input data being used was only up to second order, whilst it is suspected that higher order statistics may be being analysed by higher layers in visual cortex, each of which has at least a quadratic output from complex cells [35]. As such, at each layer visual cortex may be working on up to fourth order statistics. This could rapidly build up to quite high order analysis in several layers, as is clearly appropriate for an effective object recognition system.

5.2 Non-linear PCA

The extension of the Oja principal component analyser described in §4 to non-linear neurons has been performed in [45]. This uses, in the case of the linear and quadratic neuron, the output rule

$$y = W_i x_i + W_{ij} x_i x_j \quad (53)$$

(where the summation convention is being used, with summation on any twice repeated index) and extension of the Hebbian update rule (41) to

$$\Delta W_{ij} = \eta(y x_i x_j - y^2 W_{ij}) \quad (54)$$

On averaging over the input patterns it is possible to write an extension of (42) which preserves its form. This if the two-component object $\Omega = \begin{pmatrix} W_i \\ W_{ij} \end{pmatrix}$ and the $(N + N^2) \times (N + N^2)$ matrix $C = \begin{pmatrix} C_2 & C_3 \\ C_3 & C_4 \end{pmatrix}$ be introduced, where $(C_2)_{ij} = E(x_i x_j)$ ($C_2 = C_x$ of §4), $(C_3)_{ijk} = E(x_i x_j x_k)$, $(C_4)_{ijkl} = E(x_i x_j x_k x_l)$, then (41) and (54), with y given by (53), become

$$\Delta \Omega = \eta(C\Omega - \lambda\Omega) \quad (55)$$

where $\lambda = \Omega^T C \Omega$.

The equation (55) has identical form to equation (42), but now involves up to fourth order statistics of the input patterns. It is evident that this process can be continued by adding higher order terms still into the right hand side of (53). If terms of order n are included then the object Ω^T is extended to $\Omega^T = (W_i, W_{ij}, \dots, W_{i_1 \dots i_n})$, whilst C is enlarged to

$$\begin{pmatrix} C_2 & C_3 & \dots & C_{n+1} \\ C_3 & \ddots & & \vdots \\ \vdots & & \ddots & \\ C_{n+1} & \dots & \dots & C_{2n} \end{pmatrix}.$$

Equation (55) remains unchanged.

Extensions may be given along the lines of §4.6 to learning eigenvectors corresponding to lower eigenvalues of C . These will be of importance in giving a better reconstructed image. However we will not discuss these in any detail here.

5.3 Pattern Non-linear Reconstruction

In all of these cases the learning proceeds till Ω becomes the normalised eigenvector of C with maximal eigenvalue. As in the linear case, principal component analysis may be shown to minimise the mean-squared reconstruction error, so leading again to important data compression. The reduction is not completely trivial, so we will give a brief demonstration here which extends that of §4.3 for the linear case [15].

The optimal reconstruction is expected to be a non-linear extension of (45) [34], so taking the form

$$\hat{x}_i^\alpha = y^\alpha G_i + y^\alpha y^\beta G_i^\beta + \dots \quad (56)$$

where α is the pattern label. The value of \hat{x}^α can be expressed in vector form using the notation

$$\vec{Y}^\alpha(y)^T = (y^\alpha, y^\alpha y^\beta, y^\alpha y^\beta y^\gamma, \dots) \quad (57a)$$

$$\begin{aligned} \vec{G}^T &= (\underline{G}, \underline{G}^\beta, \underline{G}^{\beta\gamma} \dots) \\ &= (\underline{a}, \underline{a}_j x_j^\beta, \underline{a}_{jk} x_j^\beta x_k^\gamma, \dots). \end{aligned} \quad (57b)$$

In terms of (57), \hat{x}^α may be written as

$$\hat{x}^\alpha = \vec{G} \cdot \vec{Y}^\alpha(y) \quad (58)$$

where the dot product in (58) is over the vector indices β, γ etc in (57a,b). The mean square reconstruction error, extending (46), is

$$\varepsilon = E(\|\underline{x} - \hat{x}\|^2) = E(\|\underline{x} - \vec{G} \cdot \vec{Y}(y)\|^2) \quad (59)$$

where y is given by the non-linear expression, extending (53), as

$$y = W_i x_i + W_{ij} x_i x_j + \dots \quad (60)$$

If we denote combinations of indices $j_1 \dots j_n$ by \underline{j} , of patterns $\gamma_1, \dots, \gamma_m$ by $\underline{\gamma}$, and $x_{j_1}^{\beta_1} \dots x_{j_n}^{\beta_n}$ by $x_{\underline{j}}^\beta$ then (59) may be written

$$\varepsilon = \sum_{\alpha, i} \left[x_i^\alpha - y^\alpha y^\beta a_{i\underline{j}} x_{\underline{j}}^\beta \right]^2 \quad (61)$$

Variations of ε in (61) with respect to $a_{i\underline{j}}$ leads to the matrix equation

$$a_{i\underline{j}} \Lambda_{\underline{j}\underline{k}} = \Lambda_{i\underline{k}} \quad (62)$$

where

$$\Lambda_{i\mathbf{k}} = (C\Omega)_i(C\Omega)_{\mathbf{k}}$$

and

$$\Lambda_{\mathbf{k}\ell} = (\Omega^T C\Omega)(C\Omega)_{\mathbf{k}}(C\Omega)_{\ell}$$

where

$$(C\Omega)_{\mathbf{k}} = \prod_{r=1}^n (C\Omega)_{k_r}.$$

Then a solution of (62) is

$$a_{i\mathbf{k}} = (C\Omega)_i(C\Omega)_{\mathbf{k}} \left[\sum_{\mathbf{k}} (\underline{\Omega} C^2 \underline{\Omega})^{|\mathbf{k}|} \right]^{-1} (\Omega^T C\Omega)^{-1} \quad (63)$$

which is a non-linear extension of (47) (and reduces to it in the linear case). The minimum error for this solution may be obtained from (59) and (63) as

$$\frac{1}{2} \left[\sum_{\alpha} (\underline{x}^{\alpha})^2 - (\Omega^T C^2 \Omega) / (\Omega^T C\Omega) \right] \quad (64)$$

This is minimised on the original weight vector when Ω is chosen as the principle eigenvector of C . In fact there are more general solutions $a_{i\mathbf{k}} = (C\Omega)_i f_{\mathbf{k}}(\Omega C^2 \Omega)$ to equation (62), but they all lead to the same minimum error (64), and hence also to the principle component choice. As in the linear case, the scaling problem enters, albeit now in a non-linear fashion. It is not as easy to resolve as in the linear case by considering additive input noise, and scaling so that input noise variances are equal. This is because it is no longer possible to write down the analogue of (51) in the non-linear case. It may be possible to give an approximation to this for small additive noise, although no definite results are available on this yet.

5.4 Non-linear Pattern Reconstruction

An important question to be resolved is as to the quantitative benefit of using higher order statistics in pattern reconstruction. This was analysed above using only the mean square error (MSE) ε of equation (59). The MSE can be reduced to zero by using all of the eigenvectors in the linear case [38]. This case can only take account of C_2 , so cannot be expected to reproduce higher order statistics correctly. To assess these higher orders a suitable extension of the MSE must be introduced.

One such is the Kullback-Liebler distance $D(P, \hat{P})$ between the input probability distribution $P(\underline{x})$ and the reconstruction distribution $P(\hat{\underline{x}})$ of §2.2.2, defined as

$$D(P, \hat{P}) = \int d^n \underline{x} P(\underline{x}) \ln[P(\underline{x})/\hat{P}(\underline{x})]. \quad (65)$$

In the linear case one has, from (45)

$$\hat{P}(\hat{\underline{x}}) = \int \delta^n(\hat{\underline{x}} - \underline{u}(\underline{w}, \underline{x})) P(\underline{x}) d^n \underline{x} \quad (66)$$

Regularisation of the δ -function and use of a Gaussian distribution for $P(\underline{x})$ allows a Gaussian distribution to be obtained for \hat{P} , and the result that $d(P, \hat{P})$ is a minimum when \underline{u} is the principal eigenvector of C_2 , as is \underline{w} . The extension of (66) to be non-linear case is

$$\hat{P}(\hat{x}) = \int \delta^n(\hat{x} - \vec{G} \cdot \vec{Y}(y(\underline{x}))) P(\underline{x}) d^n \underline{x} \quad (67)$$

where we are using the notation of the previous sub-section. This approach appears to be the most natural one in the context of the differential geometric approach to estimation theory [2] to be outlined in the next section; it has still to be pursued much further in this context.

It is possible to indicate the power of the non-linear PCA approach outlined above if an extension is made to the error term (59). This modification is designed so error minimisation is achieved for *all* of the statistics of the input, and not just the quadratic part. In terms of the notation of §5.3 the new error function is

$$\sum_{\underline{i}} \frac{1}{2} E[|x_{\underline{i}} - \vec{G}_{\underline{i}} \cdot \vec{Y}(y)|^2] \quad (68)$$

where $x_{\underline{i}} = x_{i_1} x_{i_2} \dots x_{i_n}$, and $\vec{G}_{\underline{i}}^T = (a_{\underline{i}}, a_{\underline{i}\underline{j}} x_{\underline{j}}^\beta, a_{\underline{i}\underline{j}\underline{k}} x_{\underline{j}}^\beta x_{\underline{k}}^\gamma, \dots)$. On variations with respect to the free parameters of (57b), which are now $a_{\underline{i}\underline{j}}$ (where the subscripts \underline{i} run over the same set of labels as in the summation in (68) but the indices \underline{j} are multi-vectoral as described below), the natural extension of (62) is obtained as

$$a_{\underline{i}\underline{j}} \Omega_{\underline{j}\underline{k}} \quad (69)$$

The indices $\underline{j}, \underline{k}$ in (69) are now multi-vectoral, with $\underline{j} = (j_1, \dots, j_\ell)$, $\underline{k} = (k_1, \dots, k_m)$ for some ℓ and m , but \underline{i} only denotes a single vector index, and

$$\Omega_{\underline{j}\underline{k}} = (\Omega^T \Omega) \prod_s (C\Omega)_{\underline{j}_s} \prod_r (C\Omega)_{\underline{k}_r} \quad (70a)$$

$$\Omega_{\underline{i}\underline{k}} = (C\Omega)_{\underline{i}} \prod_r (C\Omega)_{\underline{k}_r} \quad (70b)$$

Then (69) now has solution

$$a_{\underline{i}\underline{j}} = (C\Omega)_{\underline{i}} f_{\underline{j}} (\Omega^T C\Omega)^{-1} \quad (71a)$$

where

$$\sum_{\underline{j}} f_{\underline{j}} \prod (C\Omega)_{\underline{j}} = 1 \quad (71b)$$

As in the case of (64), the minimised error from (68) becomes

$$\frac{1}{2} \left[\text{Trace} C - \sum_{\underline{i}} \left[(c\Omega)_{\underline{i}} \right]^2 (\Omega^T C\Omega)^{-1} \right] \quad (72)$$

This has minimum when Ω is along the eigenvector direction with largest eigenvalue which is, exactly the principle component of C being learnt by the rule (55). thus this

rule enables minimum reconstruction error (in the MSE sense) of the input patterns and their higher statistics (up to the order contained in C).

The above analysis can be extended immediately to the case of learning lower components, by means of extending the single non-linear neuron (60) to a set of M of them. The resulting set of $\underline{\Omega}^T = (\Omega(1), \dots, \Omega(M))$ is a multi-tensor, and the learning rule, to one or other of the PCA-subspace rules, is of the form

$$\Delta \underline{\Omega} = (I - \underline{\Omega} \underline{\Omega}^T) \underline{X}(x) \underline{X}^T(x) \underline{\Omega} \quad (73a)$$

where

$$\underline{X}^T(x) = (x_i, x_i x_j, x_i x_j x_k, \dots) \quad (73b)$$

Numerous extensions of (73) are possible to obtain orthogonal decompositions of the PCA subspace; they were briefly considered in §4.6, and will not be discussed here further.

5.5 Discussion

It is relevant to note here that neurons in general are non-linear, and the effect of a sigmoid non-linearity on the neuron output on determination of the principal components of the second order input statistics have already been studied [25]. In general improved stability of the PCA algorithms were found to result from this non-linearity. In this section we have not discussed this aspect of non-linear neurons in the PCA context, but focussed on the ability of such neurons, with suitably adaptive higher-order weights, to learn higher than second order statistics of their inputs. This was first presented in [45], and practical aspects will be discussed more fully in [15]. We have indicated here briefly how to achieve the non-linear extension of much of the linear PCA analysis of the previous section. MSE approach to the reconstruction problem indicates how to achieve better assessment of the importance of the higher order statistics in the pattern reconstruction process. It will be explored more detail elsewhere [15].

6 Differential Geometry of the Manifold of Networks

6.1 Introduction

Any neural network of a given architecture is a parameterised form of mapping

$$\underline{x} \rightarrow \underline{F}(\underline{w}, \underline{x}) = \underline{y} \quad (74)$$

from the space of inputs \underline{x} to that of the outputs \underline{y} , where the parameters \underline{w} are a finite set of real valued quantities, usually the weights and thresholds of the separate neurons. As \underline{w} varies, the set of such parameterised maps \underline{F} forms a space of maps N characterising all neural networks with the given architecture. It is highly relevant to discover suitable tools for describing the intrinsic structure of N , and also the manner in which it is embedded in the space S of all mappings from \underline{x} to \underline{y} . Such structure is of importance in describing how training algorithms change the network along a trajectory on N , or more general architecture optimisation strategies modify the network inside the more general space S of all maps.

An important approach to these questions has been developed by Amari [2] in terms of differential geometric concepts associated with statistical estimation theory. This approach uses suitable parametrisations of N , so that will be considered first. We will then, in the following sub-sections discuss the appropriate differential geometry and consider neural networks in that framework.

6.2 Network Parametrizations

The simplest and most complete parametrization is for the case of stochastic neurons with n binary inputs \underline{u} and a single output y . In that case the probability of emitting a one is a quantity which we can denote $\alpha_{\underline{u}}$:

$$\text{prob}(y = 1|\underline{u}) = \alpha_{\underline{u}} \quad (75a)$$

$$\text{prob}(y = 0|\underline{u}) = 1 - \alpha_{\underline{u}} \quad (75b)$$

The 2^n set of quantities $\underline{\alpha} = \{\alpha_{\underline{u}}\}$ give a complete description of the neuron. Such a neuron has been termed a *pRAM* [45] and discussed extensively in a series of papers (see [17] or [18] for a review). The pRAM has the particularly attractive features of (a) having a simple hardware realisability; (b) possessing continuously variable ‘‘connection weights’’ $\alpha_{\underline{u}}$, which can be trained by reward learning in a hardware-realizable manner [14]; and (c) representing the stochastic response arising from noisy quantal synaptic transmission in living neurons. As such, pRAMs completely fill the space S_{pRAM} of binary input and output stochastic neurons. Subspaces N_k of S can be formed by those pRAMs which only have non-zero memory contents $\alpha_{\underline{u}}$ for $|\underline{u}| \leq k$, so that

$$N_1 \subset N_2 \subset \dots \subset N_n = S_{pRAM} \quad (76)$$

Networks of pRAMs can be built [14], and are parametrised by the $\underline{\alpha}$ ’s of the respective pRAM components; they also will in general be subsets of S_{pRAM} . Other subspaces of S_{pRAM} exist, such as that formed by linear weighted summed neurons with

$$\text{prob}(y = 1|\underline{u}) = (1 + e^{(-\underline{u} \cdot \underline{w} - t)})^{-1} \quad (77)$$

where \underline{w}, t are connection weights and threshold respectively.

Parametrization of graded input or output neurons can be developed in a similar manner. The neurons of §4.1 are parametrised by a single vector, whilst the non-linear neuron of §5 by the multi-tensor quantity $\Omega = (w_i, w_{ij}, w_{ijk}, \dots)$. In general a neuron may have an input-output transform which is parametrised by any number of parameters, so that for such cases N can be infinite-dimensional. However it would be usual to have a bound on the number of adaptive parameters in a neural net, so only finite dimensional subspaces of S would arise.

6.3 Differential Geometry

The output of the stochastic neuron of the previous subsection can be interpreted as the random variable $r_{\underline{u}}$ for binary input \underline{u} , so that a 2^n -component random variable $\underline{r} = \{r_{\underline{u}}\}$

ensues, with probability

$$p(\underline{x}, \underline{\alpha}) = \prod_{\underline{u}} [r_{\underline{u}} \alpha_{\underline{u}} + (1 - r_{\underline{u}})(1 - \alpha_{\underline{u}})] \quad (78)$$

The above expression gives a family of probability distributions for \underline{x} , co-ordinatised by the parameters $\underline{\alpha}$ of the stochastic neuron. It is possible to introduce an exponential family of co-ordinates $\theta_{\underline{u}}$, with

$$\alpha_{\underline{u}} = (1 + e^{-\theta_{\underline{u}}})^{-1} \quad (79)$$

in terms of which a dually-flat Riemannian structure can be defined.

A dually flat manifold is defined in terms of two special sets of dually coupled co-ordinates $\underline{\theta}$ and \underline{z} . Linear curves in $\underline{\theta}$ or in \underline{z} are geodesics which are dual to each other. This duality is defined by the tangent vectors $\underline{e}_{\underline{u}}$ along the co-ordinates $\theta^{\underline{u}}$ and $\underline{e}^{\underline{u}}$ along the $\alpha_{\underline{u}}$, with the condition

$$\underline{e}^{\underline{u}} \cdot \underline{e}_{\underline{v}} = \delta_{\underline{v}}^{\underline{u}} \quad (80)$$

The structure of the manifold is determined by potential functions $\psi(\underline{\theta})$ and $\phi(\underline{\alpha})$ with

$$\theta^{\underline{u}} = (\delta / \delta \alpha_{\underline{u}}) \phi(\underline{\alpha}) \quad (81a)$$

$$\alpha_{\underline{u}} = (\delta / \delta \theta^{\underline{u}}) \psi(\underline{\theta}) \quad (81b)$$

$$\phi(\underline{\alpha}) + \psi(\underline{\theta}) = \sum_{\underline{u}} \theta^{\underline{u}} \alpha_{\underline{u}} \quad (81c)$$

and the metrics

$$g_{\underline{u}\underline{v}} = (\delta^2 / \delta \theta^{\underline{u}} \delta \theta^{\underline{v}}) \psi(\underline{\theta}) \quad (82a)$$

or inverse

$$g^{\underline{u}\underline{v}} = (\delta^2 / \delta \alpha_{\underline{u}} \delta \alpha_{\underline{v}}) \phi(\underline{\alpha}) \quad (82b)$$

An important result [1] is that a dually flat manifold admits a unique invariant divergent measure $D(P, Q)$ between any two points P, Q with value

$$D(P, Q) = \psi(\underline{\theta}_P) + \phi(\underline{\alpha}_Q) - \underline{\theta}_P \cdot \underline{\alpha}_Q \quad (83)$$

If the manifold is that of probability distributions then this divergence is identical with the Kullback-Liebler divergence which was mentioned briefly in the previous section. The divergence is itself an extension of the Euclidean distance to the case of the metric (82). It has the useful generalised Pythagorean property

$$D(P, Q) + D(Q, R) = D(P, R) \quad (84)$$

if the $\underline{\theta}$ geodesic connecting P and Q is orthogonal at Q to the dual \underline{z} -geodesic connecting Q and R . Moreover it has the further valuable projection property that the point Q_P in a sub-manifold M of S which minimises the divergence to any point P in S is given by the \underline{z} -geodesic projection of P onto M .

6.4 Geometry of the Neuron Manifold

In the case of the pRAM of §5.2, the $\theta_{\underline{u}}$ and $\alpha_{\underline{u}}$ co-ordinates were already defined by equation (75) and (79). The corresponding metric and potential functions of §5.2 are then [2]

$$\psi(\underline{\theta}) = \sum_{\underline{u}} \left[\theta_{\underline{u}} + \ln(1 + e^{-\theta_{\underline{u}}}) \right] \quad (84a)$$

$$\phi(\underline{\alpha}) = \sum_{\underline{u}} \left[\alpha_{\underline{u}} \ln \alpha_{\underline{u}} + (1 - \alpha_{\underline{u}}) \ln(1 - \alpha_{\underline{u}}) \right] \quad (84b)$$

$$g_{\underline{u}\underline{v}} = \text{diag}[\alpha_{\underline{u}}(1 - \alpha_{\underline{u}})] \quad (84c)$$

$$\alpha_{\underline{u}} = (1 + e^{-\theta_{\underline{u}}}) \quad (84d)$$

$$g^{\underline{u}\underline{v}} = \text{diag}[\alpha_{\underline{u}}^{-1}(1 - \alpha_{\underline{u}})^{-1}] \quad (84e)$$

where $g_{\underline{u}\underline{v}}$ is the Fisher information metric and its inverse. For N observations, the estimation error of an output $r_{\underline{u}}$ is given by

$$E((\hat{r}_{\underline{u}} - r_{\underline{u}})^2) = \frac{1}{N} \alpha_{\underline{u}}(1 - \alpha_{\underline{u}}) \quad (85)$$

The invariant divergence between two points $P(\underline{\alpha})$ and $Q(\underline{\beta})$ is given simply by the Kullback-Liebler information distance

$$D(P, Q) = \sum_{\underline{u}} \left[\alpha_{\underline{u}}^P \ln \frac{\alpha_{\underline{u}}^P}{\alpha_{\underline{u}}^Q} + (1 - \alpha_{\underline{u}}^P) \ln \frac{1 - \alpha_{\underline{u}}^P}{1 - \alpha_{\underline{u}}^Q} \right] \quad (86)$$

It is possible to use the geometric theorems of the previous subsection to show [2] how the approximation error $D(P^*, P_k)$, for a neuron P_k of order k , so with $\alpha_{\underline{u}} = 0$ for $|\underline{u}| > k$, can be decomposed into a sum of distances between approximations to neurons of successively higher orders

$$D(P^*, P_k) = \sum_{m=k}^{n-1} D(P_{m+1}^*, P_m^*) + D(P_k^*, P_k) \quad (87)$$

with $P_n^* = P^*$.

The above approach has been applied to analyse learning in the Boltzman machine [3] and to obtain [2] the error between P and the maximum likelihood estimate P^* , obtained when T is the number of training examples and d the number of free parameters (2^n in this case), to be

$$D(P, P^*) = (d/2T) \quad (88)$$

This is an important result on generalisation, and is at the basis of the ‘‘Rule of Thumb’’ that there must be an order of magnitude more training patterns than free parameters in order to guarantee a generalisation error of less than 5%. More detailed analysis have been given of learning in particular situations [3], to which attention is directed.

It is also possible to extend the structure to graded inputs and/or outputs, when the spaces of networks becomes an infinite-dimensional function space. The associated expression (78) becomes the weighted log likelihood

$$\ell(\underline{r}, \underline{z}) = \int P(\underline{x}) \log[\alpha(\underline{x})r(\underline{x}) + (1 - \alpha(\underline{x}))(1 - r(\underline{x}))] d\underline{x} \quad (89)$$

with $p(\underline{x})$ the input probability distribution, $\alpha(\underline{x})$ is the output for \underline{x} , and $r(\underline{x})$ is the random variable with

$$\text{prob}(r(\underline{x}) = 1) = \alpha(\underline{x}) \quad (90)$$

The above framework can evidently be extended to more than one output in an evident manner.

7 Discussion

It should be clear from the foregoing that information-theoretic approaches are making important progress in analysing both artificial neural networks and possible processing strategies in early vision (at both retinal and visual cortical levels). This is particularly true for understanding preprocessing, in terms of predictive coding in the retina and PCA and decorrelation processing in early visual cortex. However there is a difficult question which must be considered before we can be satisfied with the explanations given in this chapter of neurobiological information processing, which arises because living neurons and their nets are far more complex than those considered in the present analyses. Even if we are including higher orders of non-linearity in the response function in §5 there is no inclusion of complex temporal features as might arise from channel openings and closings (alpha functions) or from neuronal geometry, or of the details of stochastic synaptic transmission, or of the many other features of living neurons. The question is therefore if the explanations of retinal and early visual cortical processing, given in terms of information-theoretic principles, and on the basis of non-physiological realistic learning rules, will still be valid when more realistic neurons and nets are used? Moreover learning rules themselves must have a biological reality. Will there be realistic rules which will lead to the desired connection weights?

It is clear that some of the discussion in the earlier sections contravenes known biological features. Thus the anti-Hebbian learning rules of (2.12) of ref [34], and of equation (25) here, use adaptivity of inhibitory synapses, which is a feature with no experimental validation. It is possible to avoid this problem by using fixed inhibitory feedback but variable lateral excitatory connections on the inhibitory neurons. This corresponds to having the adaptive connection weight matrix V for the lateral excitatory connections in fig 3(b), but the fixed inhibitory connection matrix- C for the feedback.

Thus the equation (23) becomes

$$y = x - Cz, \quad z = V^T y \quad (91)$$

and (24)

$$y = (1 + CV^T)^{-1} x \quad (92)$$

Then the learning algorithm of equations (25) or (26) will again give uncorrelated output when learning has been completed, but now only the excitatory synapses have been trained. The inhibitory interneurons will still perform decorrelation on the outputs. However care would have to be taken to ensure that the weights v_{ij} do not go negative, so that equation (25) would have to be modified so as to be clipped when the v_{ij} become zero. The resultant level of decorrelation has yet to be analysed.

Another aspect where biological realism could be introduced is associated with PCA in both sections 4 and 5. This in particular involves the presence of the decay term proportional to the square of the output in (41) or (54). Both of these expressions involve a global assessment of output, and its use in a local manner at each synapse. To avoid this one could take a decay term depending only on the weight value at the synapse [36], as

$$\Delta w_{ij} = \eta(x_i y_j - w_{ij}^n) \quad (93)$$

This has been shown to result in asymptotic weight values proportional to a high root of the principal eigenvector components of the correlation function; its value for PCA is not yet clarified.

These modifications only go a very short way towards responding to the equation raised earlier. There is no reason, however, why approximations to living neurons cannot be used to advance understanding in this area; the use of more realistic neurons will ultimately have to be faced up to. Use of non-trivial temporal features [12] and other properties are increasingly being modelled, so that this aspect of the program can be started.

Finally we note there are many directions for future work. We have only scratched the surface of this very important approach to neural networks, leaving out numerous avenues being actively followed by others. However we hope that we have given in this chapter enough of a survey to indicate the nature of the approach.

Acknowledgements

One of the authors (MDP) is supported by a Temporary Lectureship from the Academic Initiative of the University of London. The other author (JGT) would like to thank DRA for financial support to allow part of this work to be done.

References

- [1] S.-I. Amari. Differential geometry of a parametric family of invertible linear systems—Riemannian metric, dual affine connections and divergence. *Mathematical Systems Theory*, 20:53–82, 1987.
- [2] S.-I. Amari. Dualistic geometry of the manifold of higher-order neurons. *Neural Networks*, 4:443–451, 1991.
- [3] S.-I. Amari, K. Kwata, and H. Nagaoka. Geometry of Boltzmann machine manifolds. Mathematical Engineering Technical Report 90-19, Univeristy of Tokyo, Faculty of Engineering, 1990.
- [4] J. J. Atick and A. N. Redlich. Quantitative tests of a theory of retinal processing: Contrast sensitivity curves. Technical Report IASSNS-HEP-90/51, NYU-NN-90/2, School of Natural Sciences, Institute for Advanced Study, Princeton; Center for Neural Science, New York University, 1990.

- [5] J. J. Atick and A. N. Redlich. Towards a theory of early visual processing. *Neural Computation*, 2:308–320, 1990.
- [6] J. J. Atick and A. N. Redlich. What does the retina know about natural scenes? *Neural Computation*, 4:196–210, 1992.
- [7] J. J. Atick and A. N. Redlich. Convergent algorithm for sensory receptive field development. *Neural Computation*, 5:45–60, 1993.
- [8] F. Attneave. Some informational aspects of visual perception. *Psychological review*, 61:183–193, 1954.
- [9] H. B. Barlow. Three points about lateral inhibition. In W. Rosenblith, editor, *Sensory Communication*, pages 782–786. MIT Press, 1961.
- [10] H. B. Barlow and P. Földiák. Adaptation and decorrelation in the cortex. In *The Computing Neuron*, pages 54–72. Addison-Wesley, Wokingham, England, 1989.
- [11] S. Becker and G. E. Hinton. Spatial coherence as an internal teacher for a neural network. Technical Report CRG-TR-89-7, Department of Computer Science, University of Toronto, Dec. 1989.
- [12] P. C. Bressloff and J. G. Taylor. Spatio-temporal pattern processing in a compartmental model neuron. *Physics Review E*, 1993. (in press).
- [13] J. S. Bridle. Probabilistic interpretation of feedforward classification network outputs, with relationships to statistical pattern recognition. In F. F. Soulié and J. Hérault, editors, *Neurocomputing - Algorithms, Architectures and Applications*, pages 227–236, Berlin, 1990. Springer-Verlag.
- [14] T. G. Clarkson, D. Gorse, and J. G. Taylor. Hardware realisable models of neural processing. In *Proceedings of the IEE First International Conference on Artificial Neural Networks*, pages 242–246, 1989.
- [15] S. Coombes, R. Petersen, J. G. Taylor, and S. Wright. Non-linear principal component analysis and pattern reconstruction. In preparation.
- [16] P. Földiák. Adaptive network for optimal linear feature extraction. In *Proceedings of the International Joint Conference on Neural Networks, IJCNN-89*, pages 401–405, Washington DC, 18-22 June 1989.
- [17] D. Gorse and J. G. Taylor. On the equivalence and properties of noisy neural and probabilistic RAM nets. *Physics Letters A*, 131:326–332, 1988.
- [18] D. Gorse and J. G. Taylor. A review of the theory of pRAMs. In *Proceedings of the Weightless Neural Networks Conference*, York, UK, Apr. 1993. (To appear).
- [19] D. O. Hebb. *The Organization of Behavior*. Wiley, New York, 1949.

- [20] D. H. Hubel and T. N. Wiesel. Receptive fields of single neurones in the cat's striate cortex. *Journal of Physiology*, 148:574–591, 1959.
- [21] R. Linsker. Self-organization in a perceptual network. *IEEE Computer*, 21(3):105–117, Mar. 1988.
- [22] R. Linsker. Local synaptic learning rules suffice to maximize mutual information in a linear network. *Neural Computation*, 4:691–702, 1992.
- [23] E. Oja. A simplified neuron model as a principal component analyser. *Journal of Mathematical Biology*, 15:267–273, 1982.
- [24] E. Oja and J. Karhunen. On stochastic approximation of the eigenvectors and eigenvalues of the expectation of a random matrix. *Journal of Mathematical Analysis and Applications*, 106:69–84, 1985.
- [25] E. Oja, H. Ogawa, and J. Wanguiwattana. Learning in nonlinear constrained Hebbian networks. In T. Kohonen et al., editors, *Artificial Neural Networks*, pages 385–390. Elsevier, 1991.
- [26] A. Papoulis. *Probability, Random Variables and Stochastic Processes*. McGraw-Hill, second edition, 1984.
- [27] B. A. Pearlmutter and G. E. Hinton. G-maximization: An unsupervised learning procedure for discovering regularities. In J. S. Denker, editor, *Proceedings of Neural Networks for Computing*, pages 333–338. American Institute of Physics, 1986.
- [28] M. D. Plumbley. On information theory and unsupervised neural networks. Technical Report CUED/F-INFENG/TR.78, Cambridge University Engineering Department, Cambridge, UK, 1991.
- [29] M. D. Plumbley. Approximating optimal information transmission using local Hebbian algorithms in a double feedback loop. In *Proceedings of the International Conference on Artificial Neural Networks, ICANN'93*, Amsterdam, Sept. 1993. (To appear).
- [30] M. D. Plumbley. Efficient information transfer and anti-Hebbian neural networks. *Neural Networks*, 1993. (in press).
- [31] M. D. Plumbley. A Hebbian/anti-Hebbian network which optimizes information capacity by orthonormalizing the principal subspace. In *Proceedings of the IEE Artificial Neural Networks Conference, ANN-93*, Brighton, UK, May 1993. (To appear).
- [32] M. D. Plumbley and F. Fallside. An information-theoretic approach to unsupervised connectionist models. In D. Touretzky, G. Hinton, and T. Sejnowski, editors, *Proceedings of the 1988 Connectionist Models Summer School*, pages 239–245, San Mateo, CA., 1988. Morgan-Kaufmann.

- [33] M. D. Plumbley and F. Fallside. The effect of receptor signal-to-noise levels on optimal filtering in a sensory system. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing, ICASSP-91*, volume 4, pages 2321–2324, Toronto, Canada, May 1991.
- [34] T. Poggio. On optimal nonlinear associative recall. *Biol. Cybernetics*, 19:201–209, 1975.
- [35] D. A. Pollen, J. P. Gaska, and L. D. Jacobson. Physiological constraints on models of vision. In R. M. J. Cotterill, editor, *Models of Brain Function*, pages 115–136. Cambridge University Press, 1989.
- [36] H. Riedel and D. Schild. The dynamics of Hebbian synapses can be stabilized by a nonlinear decay term. *Neural Networks*, 5:459–463, 1992.
- [37] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning internal representations by error propagation. In *Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Vol. 1: Foundations*. Bradford Books/MIT Press, Cambridge, MA, 1986.
- [38] T. D. Sanger. Optimal unsupervised learning in a single-layer linear feedforward neural network. *Neural Networks*, 2:459–473, 1989.
- [39] T. D. Sanger. An optimality principle for unsupervised learning. In D. S. Touretzky, editor, *Advances in Neural Information Processing Systems 1*, pages 11–19. Morgan Kaufmann, San Mateo, CA, 1989.
- [40] C. E. Shannon. A mathematical theory of communication. *Bell System Technical Journal*, 27:379–423, 623–656, 1948.
- [41] C. E. Shannon. Communication in the presence of noise. *Proceedings of the IRE*, 37:10–21, 1949.
- [42] C. E. Shannon. Prediction and entropy of printed english. *Bell System Technical Journal*, 30:50–64, 1951.
- [43] S. A. Solla, E. Levin, and M. Fleisher. Accelerated learning in layered neural networks. *Complex Systems*, 2, 1988.
- [44] M. V. Srinivasan, S. B. Laughlin, and A. Dubs. Predictive coding; a fresh view of inhibition in the retina. *Proceedings of the Royal Society of London, Series B*, 216:427–459, 1982.
- [45] J. G. Taylor and S. Coombes. Learning higher order correlations. *Neural Networks*, 6(3):423–428, 1993.
- [46] C. von der Malsburg. Self-organization of orientation sensitive cells in the striate cortex. *Kybernetik*, 14:85–100, 1973.

- [47] S. Watanabe. *Pattern Recognition: Human and Mechanical*. John Wiley & Sons, New York, 1985.
- [48] R. J. Williams. Feature discovery through error-correction learning. ICS Report 8501, University of California, San Diego, 1985.
- [49] R. S. Zemel and G. E. Hinton. Discovering viewpoint-invariant relationships that characterize objects. In *Advances in Neural Information Processing Systems 3*, San Mateo, CA, 1991. Morgan Kaufmann.