

Unsupervised learning of generative and discriminative weights encoding elementary image components in a predictive coding model of cortical function

M. W. Spratling

King's College London, Department of Informatics and Division of Engineering, London. UK.

Abstract

A method is presented for learning the reciprocal feedforward and feedback connections required by the predictive coding model of cortical function. Using this method feedforward and feedback connections are learnt simultaneously and independently in a biologically plausible manner. The performance of the proposed algorithm is evaluated by applying it to learning the elementary components of artificial images and of natural images. For artificial images the bars problem is employed and the proposed algorithm is shown to produce state-of-the-art performance on this task. For natural images, components resembling Gabor functions are learnt in the first processing stage and neurons responsive to corners are learnt in the second processing stage. The properties of these learnt representations are in good agreement with neurophysiological data from V1 and V2. The proposed algorithm demonstrates for the first time that a single computational theory can explain the formation of cortical RFs, and also the response properties of cortical neurons once those RFs have been learnt.

Keywords: cortical circuits; cortical feedback; image components; predictive coding; generative models; primary visual cortex; V2

1 Introduction

The Predictive Coding/Biased Competition (PC/BC) model of cortical function is a reciprocally connected, hierarchical, neural network model. It has been shown to successfully simulate a wide range of neurophysiological data ranging from the response properties of cells in primary visual cortex (Spratling, 2010) to the effects of visual attention (Spratling, 2008a). However, this previous work has used networks in which the synaptic weights were predefined rather than learnt. This article presents a learning method for PC/BC and demonstrates that this algorithm extracts the elementary components from a set of training images.

Importantly, the proposed learning algorithm allows feedforward and feedback connections to be learnt independently and simultaneously. It thus demonstrates one method by which reciprocal connections within (Callaway, 1998) and between (Felleman and Van Essen, 1991) cortical areas might be learnt. The proposed learning algorithm is biologically plausible being unsupervised and employing learning rules that only require information local to each synapse. Furthermore, the proposed algorithm employs only non-negative firing rates and non-negative synaptic weights. The learning rules are also self-normalising avoiding problems with the unbounded growth of synaptic weights without the need for a separate normalisation process. In addition, learning can occur at every iteration of the algorithm, rather than only at set times (such as when the response has settled into a steady-state) and there is no need to reset the network activity prior to the presentation of a new stimulus. This results in an algorithm that requires no knowledge of when the input changes from one stimulus to the next and without any requirement that training stimuli are presented for equal periods of time.

The algorithm is applied to extracting the elementary components from natural images and is shown to learn Gabor-like receptive fields (RFs). This demonstrates that the predefined weights used in previous work (Spratling, 2010, 2011) could be learnt. Hence, not only can the PC/BC model accurately simulate the response properties of cells in primary visual cortex (V1), but can also learn the RFs of those cells. This demonstrates that a single computational theory, predictive coding, can account both for learning V1 RFs and the behaviour of V1 cells. A second processing stage receiving input from the first is also shown to learn RF properties resembling those found in cortical area V2. This demonstrates that the same computational theory can explain cortical response properties beyond V1.

The performance of the algorithm is also tested on an artificial task: a number of variations of the bars problem. In such a task the underlying image components are known, and hence, it is possible to quantitatively evaluate performance. The results are compared to existing algorithms and shown provide a significant improvement in performance on this task.

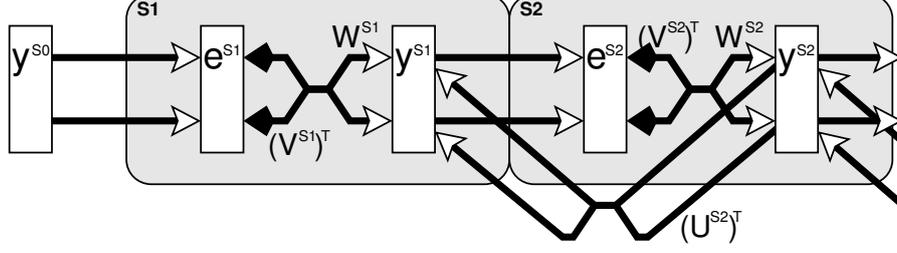


Figure 1: The PC/BC model. Rectangles represent populations of neurons, with y labelling populations of prediction neurons and e labelling populations of error-detecting neurons. Open arrows signify excitatory connections, filled arrows indicate inhibitory connections, crossed connections signify a many-to-many connectivity pattern between the neurons in two populations, parallel connections indicate a one-to-one mapping of connections between the neurons in two populations, and large shaded boxes with rounded corners indicate different cortical areas or processing stages.

2 Methods

2.1 Activation Rules

Spratling (2008a) introduced a nonlinear model of predictive coding (nonlinear PC/BC), illustrated in Figure 1, that is implemented using the following equations:

$$\mathbf{e}^{S_i} = G(\mathbf{y}^{S_{i-1}}) \oslash \left[\epsilon_2 + (\mathbf{V}^{S_i})^T \mathbf{y}^{S_i} \right] \quad (1)$$

$$\mathbf{y}^{S_i} \leftarrow (\epsilon_1 + \mathbf{y}^{S_i}) \otimes \mathbf{W}^{S_i} \mathbf{e}^{S_i} \otimes \left[1 + \eta G \left((\mathbf{U}^{S_{i+1}})^T \mathbf{y}^{S_{i+1}} \right) \right] \quad (2)$$

Where superscripts of the form S_i indicate processing stage i of a hierarchical neural network; \mathbf{e}^{S_i} is a (m by 1) vector of error-detecting neuron activations; \mathbf{y}^{S_i} is a (n by 1) vector of prediction neuron activations; \mathbf{W}^{S_i} , \mathbf{V}^{S_i} and \mathbf{U}^{S_i} are (n by m) matrices of synaptic weight values; ϵ_1 , ϵ_2 , and η are parameters; \oslash and \otimes indicate element-wise division and multiplication respectively (addition and subtraction operations involving scalars and matrices/vectors are also performed element-wise); and G is a function that clips values at 1 (*i.e.*, $G(x) = x \forall x \leq 1$; $G(x) = 1 \forall x > 1$). Note that other saturating functions, such as \tanh or a sigmoid (as was used in De Meyer and Spratling, 2009), can be used for G . A simple clipping function has been used here to avoid introducing extra parameters into the model. G prevents runaway increases in the \mathbf{y} values caused by positive feedback effects in a hierarchical network.

The values of \mathbf{y}^{S_i} represent predictions of the causes underlying the inputs to processing stage S_i . The values of \mathbf{e}^{S_i} represent the residual error between the input reconstructed from these predictions and the actual input to the processing stage.

In equation 1, the first term on the right-hand side, $G(\mathbf{y}^{S_{i-1}})$, represents the feedforward input received by the error-detecting neurons from prediction neurons in the preceding processing stage. Each error-detecting neuron compares this input with a reconstruction of the input, $(\mathbf{V}^{S_i})^T \mathbf{y}^{S_i}$, generated by the prediction neurons in the current processing stage. The comparison takes the form of a division of the actual input by the predicted input, and hence, an element of \mathbf{e}^{S_i} will equal unity when the prediction of that element of the input is correct, and will have a value of less than, or greater than, unity when the prediction is incorrect. ϵ_2 is a constant that prevents division-by-zero errors. It is sufficiently small to have a negligible effect on the calculation of \mathbf{e}^{S_i} except when an element of the vector $(\mathbf{V}^{S_i})^T \mathbf{y}^{S_i} \approx 0$.

In equation 2, the first term on the right-hand side means that the new value of a prediction neuron's response will be proportional to its value at the previous iteration. ϵ_1 is a small constant that allows a node to become active even if its response is initially zero. The $\mathbf{W}^{S_i} \mathbf{e}^{S_i}$ term represents the feedforward input to the prediction neurons from the error-detecting neurons. The strength of this feedforward input will be influenced by the strength of the error neuron outputs (which are in turn influenced by the sensory-driven input to the processing stage and the accuracy of the top-down predictions, as described in the preceding paragraph) and by the similarity between this activity and the synaptic weights, \mathbf{W}^{S_i} . The final term on the right-hand side of equation 2, the term in square brackets, represent the top-down predictions from the subsequent processing stage. These top-down predictions have a modulatory influence on the predictions of the current processing stage. If there are no top-down inputs

(i.e., if $\mathbf{y}^{S^{i+1}} = 0$), then, the last term in equation 2 can be ignored, and equation 2 simplified to:

$$\mathbf{y}^{S^i} \leftarrow (\epsilon_1 + \mathbf{y}^{S^i}) \otimes \mathbf{W}^{S^i} \mathbf{e}^{S^i}. \quad (3)$$

Equations 1 and 2 are evaluated for each processing stage in the hierarchy (starting from the lowest), and the values of \mathbf{y}^{S^i} given by equation 2 are then substituted back into equations 1 and 2 to recursively calculate the changing neural activations at each time-step. To explore the dynamics of these equations, consider a single processing stage in isolation. This allows us to ignore feedback from later stages and substitute equation 3 for equation 2. Equations 1 and 3 describe the activation dynamics within one processing stage (or cortical area) of the PC/BC model. This mechanism for determining the neural activations is called Divisive Input Modulation (DIM; Spratling et al., 2009).

Each element of \mathbf{y}^{S^i} will converge to a steady-state when the corresponding element of the vector $\mathbf{W}^{S^i} \mathbf{e}^{S^i}$ has a value of one. In other words, when an element of the vector $\mathbf{W}^{S^i} \mathbf{e}^{S^i}$ has a value of unity, the values for the corresponding element of \mathbf{y}^{S^i} on the left- and right-hand sides of equation 3 will be the same (assuming ϵ_1 is sufficiently small to be negligible). The neural responses will converge towards this steady-state since if $\mathbf{W}^{S^i} \mathbf{e}^{S^i}$ is greater than unity, \mathbf{y}^{S^i} will increase (due to equation 3). This will subsequently decrease the values of \mathbf{e}^{S^i} (via equation 1) which will in turn reduce the value of $\mathbf{W}^{S^i} \mathbf{e}^{S^i}$. In contrast, if $\mathbf{W}^{S^i} \mathbf{e}^{S^i}$ is less than unity, \mathbf{y}^{S^i} will decrease (due to equation 3). This will subsequently increase the values of \mathbf{e}^{S^i} (via equation 1) which will in turn increase the value of $\mathbf{W}^{S^i} \mathbf{e}^{S^i}$.

If appropriate values for \mathbf{W}^{S^i} and \mathbf{V}^{S^i} are predefined, or are learnt (as described in the next section), then prediction neurons that represent patterns that are present in the input converge to a steady-state value of one, while other prediction neurons converge to a response of zero. Similarly, error-detecting neurons that correspond to active inputs converge to a value of one, while other error-detecting neurons produce a response of zero. \mathbf{W}^{S^i} needs to be a matrix in which the rows represent elementary image components (or basis functions) useful for encoding the input stimuli. \mathbf{V}^{S^i} needs to have the same values as \mathbf{W}^{S^i} but such that the rows are normalised to have a *maximum* value of one. This scaling of the feedback weights is necessary to ensure that the scale of the response \mathbf{y}^{S^i} gives a true reflection of the match between the input pattern and the pattern encoded by a prediction node's synaptic weights, see Fig. 2. In contrast, the rows of \mathbf{W}^{S^i} need to be normalised such that the *sum* of each row is equal to one. When this is true, elements of \mathbf{e}^{S^i} that correspond to active inputs will converge to a value of one, if the input can be accurately predicted by the prediction nodes, see Fig. 3. The deviation of active elements of \mathbf{e}^{S^i} from a value of one is not dependent on the absolute strength of the input. However, when the \mathbf{W}^{S^i} and \mathbf{V}^{S^i} weights are appropriate, the strength of the prediction node responses will be proportional to the strength of the input received, see Fig. 4.

When \mathbf{W}^{S^i} and \mathbf{V}^{S^i} take appropriate values, as described above, the activation dynamics will attempt to reconstruct the input pattern as accurately as possible in terms of the basis functions encoded in the rows of \mathbf{W}^{S^i} and \mathbf{V}^{S^i} . In other words, during the recursive calculation of \mathbf{e}^{S^i} and \mathbf{y}^{S^i} , the values of \mathbf{y}^{S^i} change so as to minimise the error between the input and the top-down reconstruction of the input. If an element of \mathbf{e}^{S^i} is greater than one (respectively less than one) prediction neurons that receive input from that error-detecting neuron will increase (decrease) their response (via equation 3). The output of the prediction neurons will then more strongly (weakly) represent the predicted causes of the input, which will in turn reduce (increase) the residual error encoded by that element of \mathbf{e}^{S^i} (via equation 1). At any step in this iterative process the deviation of the elements of \mathbf{e}^{S^i} from one can be considered as a measure of the degree of mismatch between the top-down reconstruction of the input and the actual input (assuming ϵ_2 is sufficiently small to be negligible). When a value within \mathbf{e}^{S^i} is greater than unity it indicates that a particular element of the input is under-represented in the reconstruction, a value of less than unity indicates that a particular element of the input is over-represented in the reconstruction, and a value of unity indicates that the top-down reconstruction perfectly predicts the bottom-up stimulation. This deviation of \mathbf{e}^{S^i} from unity can be used (as described in the next section) to modify the values of \mathbf{W}^{S^i} and \mathbf{V}^{S^i} to improve the representation of the input stimuli encoded by the synaptic weights.

The PC/BC algorithm is closely related to, and can be interpreted in terms of, several existing computational theories of cortical function:

- As a mechanism for performing predictive coding (Rao and Ballard, 1999; Srinivasan et al., 1982) in which sensory information consistent with a top-down prediction is suppressed, so that only unpredicted information is propagated forward. In contrast to previous implementations of predictive coding, the mechanism proposed for comparing predictions with sensory-driven input is divisive rather than subtractive (Spratling, 2008a). Predictive coding is itself a specific example of more general theories of efficient encoding or redundancy reduction (Attneave, 1954; Barlow, 2001; Olshausen and Field, 1996b, 1997) and of hierarchical perceptual inference or analysis-by-synthesis (Barlow, 1994; Friston, 2005, 2009; Lee and Mumford, 2003; Mumford, 1992; Yuille and Kersten, 2006), hence, PC/BC can be related to these wider frameworks for understanding cortical function.

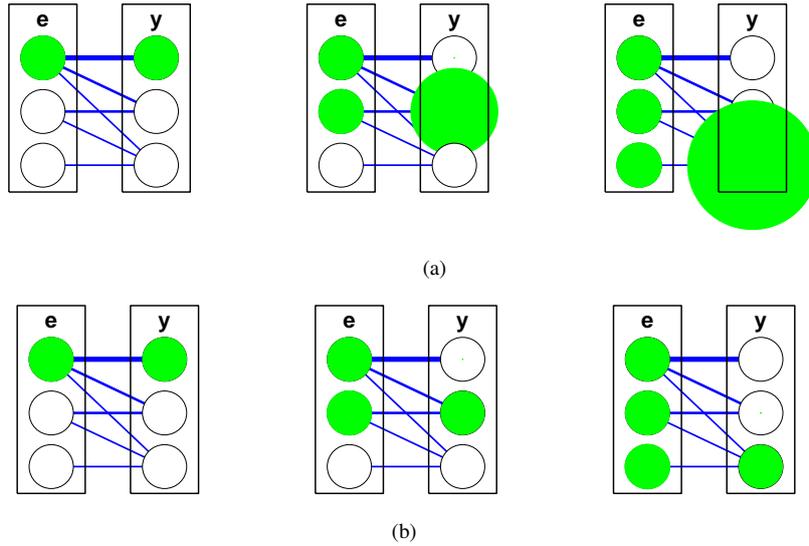


Figure 2: The effects of the strength of \mathbf{V}^{S_i} on the scale of the \mathbf{y}^{S_i} values. Each subplot shows a simple PC/BC processing stage, S_i , with three inputs and three outputs. The responses of the error nodes and the prediction nodes are shown for different combinations of input stimuli (from left to right $\mathbf{y}^{S_{i-1}} = [1, 0, 0]^T$, $\mathbf{y}^{S_{i-1}} = [1, 1, 0]^T$, $\mathbf{y}^{S_{i-1}} = [1, 1, 1]^T$). The strength of each node's response (at the end of 200 iterations) is indicated by the diameter of the shading (the shading is the same diameter as the circle when the response is equal to one). The network is wired-up so that the first prediction node represents the first input, the second prediction node represents the conjunction of the first and second inputs, and the third prediction node represents the conjunction of all three inputs. The width of each connection between the error nodes and prediction nodes is proportional to its strength (*i.e.*, the values of \mathbf{W}^{S_i}). The values of \mathbf{W}^{S_i} are normalised so that the weights received by each prediction node sum to a value of one. In (a) $\mathbf{V}^{S_i} = \mathbf{W}^{S_i}$, in (b) \mathbf{V}^{S_i} is proportional to \mathbf{W}^{S_i} but is normalised so that the feedback weights from each prediction node have a maximum value of one.

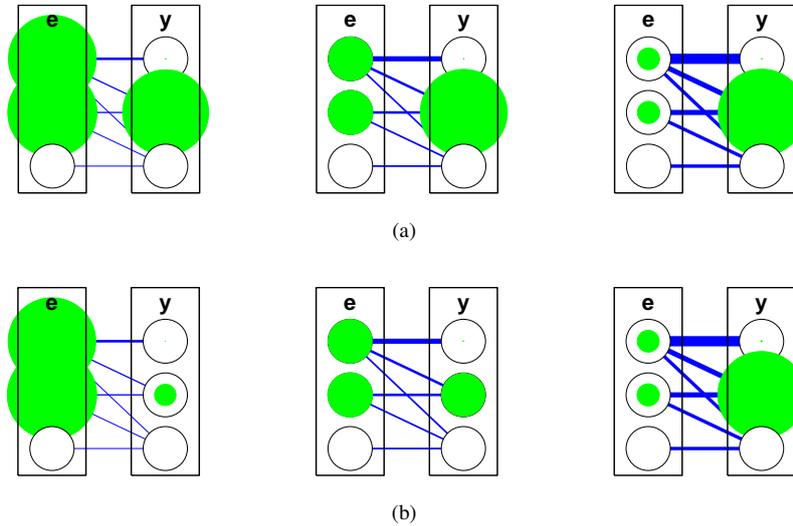


Figure 3: The effects of the strength of \mathbf{W}^{S_i} on the scale of the \mathbf{e}^{S_i} values. Each subplot shows a simple PC/BC processing stage, S_i , identical to that shown in Fig. 2. Each network receives the same input ($\mathbf{y}^{S_{i-1}} = [1, 1, 0]^T$ as for the central networks in Fig. 2). From left to right the strength of \mathbf{W}^{S_i} varies so that the feedforward weights received by each prediction node sum to a value of 0.5 (left), 1 (centre), and 2 (right). In (a) $\mathbf{V}^{S_i} = \mathbf{W}^{S_i}$, in (b) \mathbf{V}^{S_i} is proportional to \mathbf{W}^{S_i} but is normalised so that the feedback weights from each prediction node have a maximum value of one. The format of the diagram is otherwise identical to, and described in the caption of, Fig. 2.

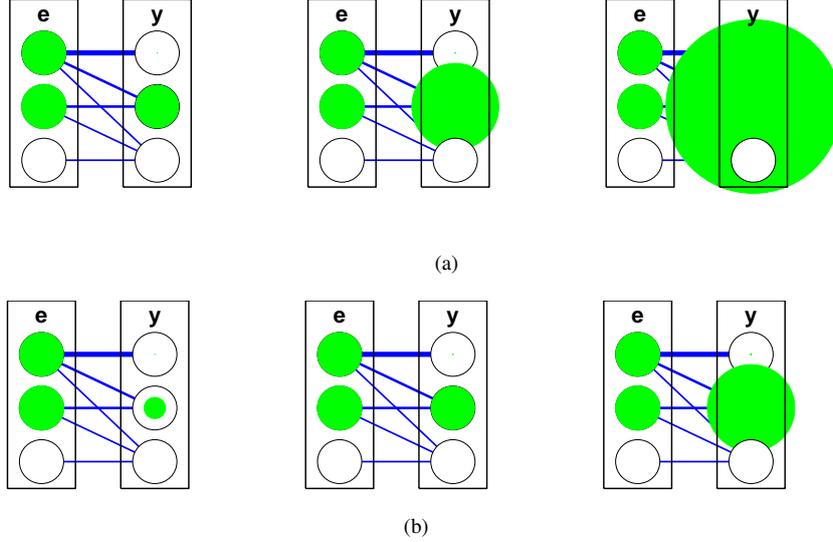


Figure 4: The effects of the strength of $\mathbf{y}^{S_{i-1}}$ on the scale of the \mathbf{y}^{S_i} values. Each subplot shows a simple PC/BC processing stage, S_i , identical to that shown in Fig. 2. From left to right the strength of \mathbf{y}^{S_i} increases such that $\mathbf{y}^{S_{i-1}} = [0.5, 0.5, 0]^T$ (left), $\mathbf{y}^{S_{i-1}} = [1, 1, 0]^T$ (centre), and $\mathbf{y}^{S_{i-1}} = [2, 2, 0]^T$ (right). Note that for the simulations in the right-hand column the clipping of the input to the processing stage has been removed (*i.e.*, $G(\mathbf{y}^{S_{i-1}}) = \mathbf{y}^{S_{i-1}}$ in equation 1), otherwise the results would be identical to those shown in the central column. In (a) $\mathbf{V}^{S_i} = \mathbf{W}^{S_i}$, in (b) \mathbf{V}^{S_i} is proportional to \mathbf{W}^{S_i} but is normalised so that the feedback weights from each prediction node have a maximum value of one. The format of the diagram is otherwise identical to, and described in the caption of, Fig. 2.

- As a mechanism of competition in which neurons compete to receive inputs (Hamker and Wiltchut, 2007; Spratling and Johnson, 2002). In this case \mathbf{e}^{S_i} represents the inhibited inputs to a population of competing prediction neurons. Each prediction neuron modulates its own inputs, which helps stabilise the response of the prediction neurons, since a strongly (or weakly) active prediction neuron will suppress (magnify) its inputs, and hence, reduce (enhance) its own response. When \mathbf{V}^{S_i} is proportional to \mathbf{W}^{S_i} , prediction neurons that share inputs (*i.e.*, that have overlapping RFs) will also modulate each other’s inputs. This generates a form of competition between the prediction neurons, such that each neuron effectively tries to block other prediction neurons from responding to the inputs which it represents. In contrast to previous implementations of this mechanism of competition, inhibition operates divisively rather than subtractively. Excitatory feedback from higher processing stages (see Fig. 1 and equation 2) can increase the responses of specific prediction nodes, increasing the likelihood that they win the ongoing competition, and hence, implements a form of biased competition model (Spratling, 2008a).
- As a generative model that calculates a feature vector, \mathbf{y}^{S_i} , defining a combination of components (or “basis vectors”, or “codebook vectors”) taken from a learnt dictionary, $(\mathbf{V}^{S_i})^T$, which can account for the sensory-driven input $\mathbf{y}^{S_{i-1}}$. As with many previous generative models the input is reconstructed from a linear combination of components (*i.e.*, as $(\mathbf{V}^{S_i})^T \mathbf{y}^{S_i}$), however, in contrast to most generative models the comparison of the top-down reconstruction with the data is nonlinear rather than linear. In contrast to many other generative models, PC/BC can deal with occlusion and overlap between image components (Spratling et al., 2009, and Section 3.1) without the need to introduce other variables into the model to account for depth (*e.g.*, Lücke et al., 2009), or the need to use a nonlinear method of reconstructing the input (*e.g.*, Lücke and Sahani, 2008), or the need for application specific constraints on the sparseness of the weights or the activations (Spratling, 2006). Furthermore, whereas many previous generative models determine the values of \mathbf{y}^{S_i} that most accurately account for the data using random sampling or by performing matching pursuit, in PC/BC the values of \mathbf{y}^{S_i} are determined through the iterative procedure defined by equations 1 and 2, with the initial estimate of these values being provided by the feedforward, discriminative, weights, \mathbf{W}^{S_i} . In this respect the proposed algorithm is similar to the Helmholtz machine (Hinton et al., 1995; Weber and Triesch, 2008), the restricted Boltzmann machine (RBM; Hinton, 2002; Teh et al., 2003), and the encoder-decoder architecture (Ranzato et al., 2007). However, in contrast to the Helmholtz machine the

proposed algorithm uses real-valued rather than binary units, is not stochastic, and does not require a sleep phase to train the feedforward weights independently of the feedback weights (see section 2.2). In contrast to the RBM, both the feedforward and feedback weights are learnt (section 2.2), rather than the feedback weights being made equal to the transpose of the feedforward weights. In contrast to the encoder-decoder architecture there is no sparsifying logic, instead a sparse representation is created through the competition between the prediction neurons. Since RBMs are the building blocks of deep belief networks (Hinton et al., 2006) and since the encoder-decoder architecture has been employed in recent versions of convolutional neural networks (Jarrett et al., 2009; LeCun et al., 2010), the full hierarchical PC/BC model can be related to these deep learning methods (Arel et al., 2010). PC/BC can also be implemented using convolution (Spratling, 2010).

- As a mechanism to perform 'explaining away' in probabilistic inference (Kersten et al., 2004). Hypotheses, \mathbf{y}^{S_i} , compete to represent the evidence, $\mathbf{y}^{S_{i-1}}$. If a prediction neuron wins the competition to represent a particular element of $\mathbf{y}^{S_{i-1}}$, then (if \mathbf{V}^{S_i} is proportional to \mathbf{W}^{S_i}) it inhibits other nodes from representing that input. Hence, if one hypothesis explains a particular piece of data, then support for alternative hypotheses is reduced. However, in ambiguous situations multiple hypotheses to explain a single input can each be concurrently active.
- As a form of divisive normalisation like that proposed by the normalisation model (Albrecht and Geisler, 1991; Carandini and Heeger, 1994; Heeger, 1991, 1992; Wainwright et al., 2001). However, unlike the normalisation model, in PC/BC the normalisation is applied to the inputs to the population of competing neurons rather than the outputs, and (when \mathbf{V}^{S_i} is proportional to \mathbf{W}^{S_i}) the normalisation pool for each neuron is restricted to the population of neurons that have overlapping receptive fields. Such divisive modulation is compatible with mechanisms of shunting inhibition or synaptic depression (Carandini, 2004; Mitchell and Silver, 2003; Rothman et al., 2009). The particular form of divisive normalisation used in PC/BC also gives rise to gain modulation as is observed in various cortical areas (for example when a retinal RF is modulated by eye position) in which different sources of feedforward inputs to the same cortical area are combined such that one modulates the response to the other (De Meyer and Spratling, 2011).

2.2 Learning Rules

In previous work (Spratling, 2008a, 2010, 2011), \mathbf{W}^{S_i} has been given predefined values and been normalised such that the sum of each row is equal to one. \mathbf{V}^{S_i} and \mathbf{U}^{S_i} have been defined to have the same values as \mathbf{W}^{S_i} but such that the rows are normalised to have a maximum value of one (*i.e.*, $\mathbf{V}^{S_i} = \mathbf{U}^{S_i} = \hat{\mathbf{W}}^{S_i}$). This article will consider how appropriate values for \mathbf{W}^{S_i} , \mathbf{V}^{S_i} and \mathbf{U}^{S_i} can be learnt such that at the end of the learning process the \mathbf{W}^{S_i} weights will have become tuned to represent elementary image components useful for representing the training images, and the \mathbf{V}^{S_i} and \mathbf{U}^{S_i} will have become proportional to the \mathbf{W}^{S_i} weights such that $\mathbf{V}^{S_i} \approx \mathbf{U}^{S_i} \approx \hat{\mathbf{W}}^{S_i}$.

A learning rule for the feedforward weights, \mathbf{W}^{S_i} , has previously been introduced and shown to perform well on an artificial task (Spratling et al., 2009). It has also been shown to learn basis functions (De Meyer and Spratling, 2011). In this article, the same learning rule for \mathbf{W}^{S_i} is used, namely:

$$\mathbf{W}^{S_i} \leftarrow \mathbf{W}^{S_i} \otimes \left\{ 1 + \beta \mathbf{y}^{S_i} \left[(\mathbf{e}^{S_i})^T - 1 \right] \right\} \quad (4)$$

Where β is a positive constant which controls the learning rate. Following each update, weights are clipped at zero to ensure that they remain non-negative.

When an element i within \mathbf{e}^{S_i} is greater than unity, indicating that input i is under-represented in the reconstruction, the values of weights connecting the under-represented input with active prediction nodes are increased, since $\left[(\mathbf{e}_i^{S_i})^T - 1 \right] > 0$ in equation 4. This will lead to an increase in the response of those prediction nodes (via a subsequent application of equation 2), and consequently an increase in the strength with which that element is represented in the reconstructed input, and hence, reduce the value of that element of \mathbf{e}^{S_i} towards one (via a subsequent application of equation 1). Similarly, when an element i within \mathbf{e}^{S_i} is less than unity, indicating that input i is over-represented in the reconstruction, the values of weights connecting the over-represented input with active prediction nodes are reduced, since $\left[(\mathbf{e}_i^{S_i})^T - 1 \right] < 0$ in equation 4. This will lead to a decrease in the response of those prediction nodes (via a subsequent application of equation 2), and consequently a decrease in the strength with which that element is represented in the reconstructed input, and hence, increase the value of that element of \mathbf{e}^{S_i} towards one (via a subsequent application of equation 1). When an element i within \mathbf{e}^{S_i} is equal to unity the reconstruction of that element is perfect and the weights stop changing, since $\left[(\mathbf{e}_i^{S_i})^T - 1 \right] = 0$ in equation 4. For elements that are not active in the input vector, the corresponding elements of \mathbf{e}^{S_i} will be zero and

the corresponding weights (for active prediction nodes) will stop changing once they have reached a value of zero. Hence, a particular weight, \mathbf{W}_{ji}^{Si} , stops changing value when the top-down reconstruction of input i is perfect (*i.e.*, when $e_i^{Si} = 1$), when the weight is zero (*i.e.*, when $\mathbf{W}_{ji}^{Si} = 0$), or when the response of the post-synaptic neuron is zero (*i.e.*, when $y_j^{Si} = 0$).

Both the activation dynamics and the learning rules thus operate to minimise the error between the input stimulus (\mathbf{y}^{Si-1}) and the prediction of the input ($(\mathbf{V}^{Si})^T \mathbf{y}^{Si}$). The activation rules do so by adjusting the prediction neuron responses to select those basis functions that most accurately represent the input. The learning rule does so by adjusting the basis functions to allow them to more accurately represent the input. The learning algorithm for \mathbf{W}^{Si} thus attempts to find *non-negative*, elementary, components of the training data, which are used by the activation rules to encode each stimulus with minimal loss of information (*i.e.*, with minimal reconstruction error). As such, the learning procedure can be related to *non-negative* matrix factorisation (NMF; Hoyer, 2004; Lee and Seung, 1999). Indeed, equation 4 can be derived from the method proposed by Lee and Seung (1999) for learning non-negative factors through minimisation of the Kullback-Leibler divergence between the training data and the reconstruction of that data (Spratling et al., 2009). However, in contrast to the original NMF learning algorithm, the method used here is an approximate, on-line learning algorithm rather than a batch one (Spratling et al., 2009). In order to develop an algorithm that is both biologically plausible and can simulate the activation dynamics and response properties cortical neurons, it is necessary to develop an on-line algorithm. This means that it is necessary to forgo the convenience of expressing the algorithm in terms of minimisation of a global objective function.

The response of the error-detecting nodes, e^{Si} , is sensitive to the overall strength of the feedforward weights, see Fig. 3. Hence, nodes with strong (or weak) weights produce strong (respectively weak) feedback that results in small (respectively large) values of e^{Si} . The learning rule acts to drive the error values towards one, which means that nodes with strong weights will have them reduced and nodes with weak weights will have them increased. Hence, the learning rule for \mathbf{W}^{Si} (together with \mathbf{V}^{Si} weights that are proportional to the reciprocal \mathbf{W}^{Si} values) has the effect of normalising the total strength of the weights received by a prediction node. Such normalisation is attractive from the point of view of biological plausibility, as synaptic weights cannot increase without bound.

The PC/BC algorithm requires that the \mathbf{V}^{Si} and \mathbf{U}^{Si} weights are proportional to the \mathbf{W}^{Si} weights. This can be achieved by using learning rules for \mathbf{V}^{Si} and \mathbf{U}^{Si} that are identical to equation 4. Such independent learning rules would, at each iteration, modify all three weight values in a similar fashion and thus result in the weights converging to similar values, as long as the starting conditions were not too dissimilar. However, equation 4 uses terms that are not local to the synapses defining the \mathbf{U}^{Si} connections and the values of \mathbf{V}^{Si} weights should ideally be rescaled, as discussed in the previous section and illustrated in Figures 2 and 4, in order to ensure that the values of \mathbf{y}^{Si} are proportional to the similarity between the input stimulus and the basis function encoded by each node, and that the values of \mathbf{y}^{Si} are also proportional to the absolute strength of the input. For these reasons distinct learning rules are proposed for \mathbf{V}^{Si} and \mathbf{U}^{Si} that enable values proportional to the reciprocal \mathbf{W}^{Si} weights to be learnt independently.

The proposed learning rule for \mathbf{V}^{Si} is as follows:

$$\mathbf{V}^{Si} \leftarrow \mathbf{V}^{Si} \otimes \left\{ 1 + \beta \mathbf{y}^{Si} \left[(\mathbf{e}^{Si})^T - 1 \right] + \beta \mathbf{H}(\mathbf{y}^{Si} - 1) \mathbf{1} \right\} \quad (5)$$

Where \mathbf{H} is the Heaviside function and $\mathbf{1}$ is a 1 by m vector in which each element is equal to 1. Following each update weights are clipped at zero to ensure that they remain non-negative. In this case \mathbf{y}^{Si} is the pre-synaptic term and \mathbf{e}^{Si} is the post-synaptic term. In contrast in equation 4, \mathbf{y}^{Si} is the post-synaptic term and \mathbf{e}^{Si} is the pre-synaptic term. The $\beta \mathbf{y}^{Si} \left[(\mathbf{e}^{Si})^T - 1 \right]$ term in equation 5 is identical to that used in equation 4 and causes similar weights to be learnt. The $\beta \mathbf{H}(\mathbf{y}^{Si} - 1) \mathbf{1}$ term increases, equally, the feedback weights originating from prediction neurons with activations exceeding unity. A strong response from a prediction node can be caused either by the strength of the feedforward weights it receives being too strong (Fig. 3a and b, right), or by the strength of the feedback weights it sends being too weak (Fig. 2a, centre and right). The first problem is addressed by equation 4 decreasing the strength of synapses emerging from error nodes with weak response. The latter problem is addressed by the $\beta \mathbf{H}(\mathbf{y}^{Si} - 1) \mathbf{1}$ term in equation 5 increasing the strength of all synapses emerging from prediction nodes with strong responses. Since the values of \mathbf{e}^{Si} are unaffected by the scale of \mathbf{V}^{Si} (compare Fig. 2a with 2b and Fig. 3a with 3b) the learning of \mathbf{W}^{Si} can proceed while the values of \mathbf{V}^{Si} are still being scaled.

The learning rule for \mathbf{U}^{Si} is as follows:

$$\mathbf{U}^{Si} \leftarrow \mathbf{U}^{Si} \otimes \left\{ 1 + \beta \mathbf{y}^{Si} \left[\left(\mathbf{y}^{Si-1} \odot \left[\epsilon_2 + (\mathbf{U}^{Si})^T \mathbf{y}^{Si} \right] \right)^T - 1 \right] \right\} \quad (6)$$

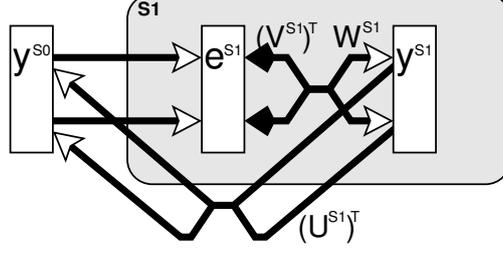


Figure 5: A single processing stage of the PC/BC model as employed in the simulations described in sections 3.1 and 3.2. The format of the figure is identical to, and described in the caption of, Fig. 1. The \mathbf{U}^{S1} connections do not have any effect on the input to the processing stage (\mathbf{y}^{S0}), or on the activations of neurons in the processing stage (\mathbf{e}^{S1} and \mathbf{y}^{S1}).

Following each update weights are clipped at zero to ensure that they remain non-negative. In this case \mathbf{y}^{S^i} is the pre-synaptic term and $\left(\mathbf{y}^{S^{i-1}} \oslash \left[\epsilon_2 + (\mathbf{U}^{S^i})^T \mathbf{y}^{S^i}\right]\right)$ is the post-synaptic term. This post-synaptic term is an approximation to \mathbf{e}^{S^i} calculated using information that is available locally to the prediction neurons in the preceding processing stage. Hence, this rule approximates equation 4 and generates similar weight values. However, as with \mathbf{V}^{S^i} , the \mathbf{U}^{S^i} weights become rescaled so that feedback weights from each prediction node have a maximum value of one. This occurs because if the \mathbf{U}^{S^i} weights targeting an input node are an underestimate (or overestimate) of the \mathbf{V}^{S^i} weights targeting the corresponding error node, then the term $\left(\mathbf{y}^{S^{i-1}} \oslash \left[\epsilon_2 + (\mathbf{U}^{S^i})^T \mathbf{y}^{S^i}\right]\right)$ at that input node will be larger (respectively, smaller) than the corresponding value of \mathbf{e}^{S^i} , so the \mathbf{U}^{S^i} weights will increase (decrease) until $\mathbf{U}^{S^i} = \mathbf{V}^{S^i}$.

In order to calculate the post-synaptic term in equation 6 it is necessary for each post-synaptic neuron to integrate the feedback signals it receives (*i.e.*, to calculate one element of $(\mathbf{U}^{S^i})^T \mathbf{y}^{S^i}$) and so be able to compare this value with the output response (*i.e.*, the corresponding element of $\mathbf{y}^{S^{i-1}}$). This suggests that there should be a separate site for integration for the feedback signals. The apical dendrites of cortical pyramidal cells appear to function as a separate site of integration (Spratling, 2002). This is therefore consistent with the identification of superficial layer pyramidal cells with the prediction neurons in the PC/BC model (Spratling, 2008b, 2011).

2.3 Experimental Procedures

Before the start of training, synaptic weight values were initialised to random values chosen from a Gaussian distribution with mean 0.5 and standard deviation 0.05. The values of \mathbf{W}^{S^i} , \mathbf{V}^{S^i} and \mathbf{U}^{S^i} were independently initialised, and hence, corresponding reciprocal weights started with different initial values. Before the first iteration of the learning algorithm, the prediction neuron responses (\mathbf{y}^{S^i}) were initialised to zero (initialising to random values produced equivalent results). An image was then presented to the network, which determined the values of \mathbf{y}^{S0} , and equations 1, 2, 4, 5, and 6 were applied recursively. At each iteration equations 1–6 were applied to each processing stage in the PC/BC hierarchy starting with the lowest level and working up. After a number of iterations the values of \mathbf{y}^{S0} were changed instantaneously to represent a new training image and the iterative application of equations 1–6 continued with no other external changes to the values of the variables. This process continued, with each new randomly selected training image being presented for a variable number of iterations. The duration for which each image was presented was selected randomly from a uniform distribution of integers in the range [1,400]. All networks were trained for 20000 image presentations or “training cycles”, unless explicitly stated otherwise.

Updating the weights at every iteration, as described above, is computationally expensive. Hence, in some simulations a faster, more practical, algorithm was applied. In this algorithm the weights were updated only once in response to each training image, and this update was performed using the steady-state responses calculated at the end of 200 iterations of equations 1 and 2. For this “steady-state” version of the training procedure a value of $\beta = 0.005$ was used in each simulation. For the “continuous learning” version of the training procedure a value of $\beta = \frac{0.005}{200} = 2.5 \times 10^{-5}$ was used. Hence, when the weights were updated at every iteration the learning rate was reduced in proportion to the average number of iterations for which each image was presented.

The only other free parameters in the model were set equal to values similar to those used in previous work (De Meyer and Spratling, 2009; Spratling, 2010, 2011), namely: $\epsilon_1 = 0.0001$ and $\epsilon_2 = 0.01$.

Sections 3.1 and 3.2 consider learning in a single PC/BC processing stage, as illustrated in Fig. 5. The effects of top-down, modulatory, inputs to this processing stage are ignored (*i.e.*, $\mathbf{y}_j^{S^{i+1}} = 0 \forall j$), and hence, equation 2

can be simplified to equation 3. Note that U^{S_i} does not appear in equations 1 and 3, and hence, these connections have no effect on neural activity. However, the U^{S_i} connections will be learnt (as described by equation 6) to demonstrate that, in principle, feedback weights to a previous processing stage could be learnt at the same time as weights W^{S_i} and V^{S_i} are learnt.

Section 3.3 considers learning in a hierarchical PC/BC model containing two processing stages, as illustrated in Fig. 1. In such a hierarchy, multiple sets of synaptic weights (*i.e.*, W^{S_1} , V^{S_1} , W^{S_2} , V^{S_2} , and U^{S_2} for a two-stage hierarchy) will be learnt simultaneously: it is not found necessary to train the lowest level of the hierarchy first, before training the next level.

As with cortical hierarchies, neurons in later processing stages are given larger RFs than neurons in earlier processing stages. This is achieved by dividing the prediction neurons in the lower processing stage into a number of groups (called “sub-regions”). The maximum extent of the RFs of the neurons in each sub-region is limited to a particular region of the visual input. Prediction neurons in the subsequent processing stage receive input from multiple sub-regions of the preceding processing stage with distinct RFs, and hence, have larger RFs. Restricting the maximum extent of the RF of a prediction neuron to a particular part of the visual input, or to a particular set of neurons in the preceding processing stage, is achieved by simply initialising the synaptic weights outside the chosen RF to be zero. The learning rules (equations 4, 5, and 6) will ensure that weights that are initially zero always remain zero. Synaptic weight values within the RF were initialised using random values. The parameters for setting the initial weights and all other parameters of the network are identical to those used in the experiments using a single processing stage in isolation, as described above. For convenience RFs are restricted to be square regions of the image. For the sake of expediency the “steady-state” method of updating the weights was used to train the hierarchical network.

2.4 Code

Software, written in MATLAB, which implements the PC/BC model described above and performs the experiments described below, is available at http://www.corinet.org/mike/Code/dim_learn_recipe_weights.zip.

3 Results

3.1 Learning Bar-Like Components from Artificial Images using a Single Processing Stage

The bars problem (and its variations) has become a benchmark for testing the ability of algorithms to learn elementary image components. The performance of a learning algorithm on one task does not predict performance on another task. This is true of the bars problem, as it is for any other task. Particularly, given the significant differences between the bars task and most real-world tasks, it is unlikely that performance on the bars task will indicate the performance likely to be achieved on more realistic problems. Despite this, the bars tasks is used here to supplement the results presented in the following sections for real-world images, because it allows a more quantitative assessment of performance to be made. With an artificial task, like the bars problem, the underlying image components of the training images are known. This allows algorithms to be quantitatively tested by comparing the components that have been represented with the known features from which the images were created. To determine which components have been represented by an algorithm, both the responses generated to test images (Lücke and Sahani, 2008; Spratling et al., 2009) and the weights learnt from training images (Spratling, 2006; Spratling et al., 2009) can be analysed. The latter is more stringent, in that (at least for the method used here) it is a more difficult test to pass. The former is more ecologically relevant, since it is the response of the network (rather than the weights of the network) that is used to transmit information.

To allow comparison with the results of previous algorithms which have been evaluated by analysing the learnt synaptic weights, the method described in Spratling (2006) was employed. For a component to be considered represented, it is required that at least one prediction node has weights that correspond to this component. Specifically, all weights corresponding to the individual inputs forming the component must be strong and the sum of these weights must be much stronger than those corresponding to other components. The outcome of this analysis is presented in terms of the number of components correctly represented, averaged over trials. Separate analyses were performed for the three sets of weights learnt by PC/BC algorithm.

To allow comparison with the results of previous algorithms which have been evaluated by analysing the responses to test patterns, the method described in Lücke and Sahani (2008) was employed. This method uses

a set of test patterns each of which contains a single, isolated, component¹. For a component to be considered represented, the node generating the maximal response to that component must not also be the one to respond most strongly to any other component. The outcome of this analysis is presented in terms of the percentage of trials in which all components are correctly represented (often referred to as the “reliability”). Since the measured response is dependent on both the \mathbf{W}^{S1} and \mathbf{V}^{S1} weights learnt by the algorithm, this test provides an analysis of the success in learning these weights (but not the \mathbf{U}^{S1} weights).

The performance of the PC/BC algorithm is compared below to published results for a number of other algorithms which have previously been demonstrated to perform well on the bars task, namely: the Maximal Causes Analysis models (MCA_3 , $R - MCA_2$, and $R - MCA_{NN}$; Lücke and Sahani, 2008), Generalised Softmax Networks (NN_{fast} and NN_{slow} ; Lücke and Sahani, 2007), two models of cortical column self-organisation (Lücke, 2009; Lücke and von der Malsburg, 2004), NMF with Sparseness Constraints (nmfsc; Hoyer, 2004), Non-negative Sparse Coding (npsc; Hoyer, 2002), and dendritic inhibition (Spratling and Johnson, 2002, 2003).

In the experiments described here, 25 trials were performed. In each trial the synaptic weights were initialised to different random values and the algorithm was trained using a different, randomly generated, set of input patterns. Unless otherwise stated, sets of 400 training images were used and each network contained 24 prediction neurons. The networks were training for 20000 image presentations (*i.e.*, each pattern in the training set was presented to the network, on average, 50 times). The results generated by the PC/BC algorithm when tested on six variations of the bars problem are summarised in Table 1. Results are provided for the “continuous” version of the training algorithm, in which the weights were adjusted at every iteration of the algorithm (see Section 2.3). However, updating the weights at every iteration is computationally expensive. Hence, results are also listed in Table 1 for a more practical algorithm in which the weights are updated only once for each training cycles using the steady-state response calculated at the end of 200 iterations (see Section 2.3). The results for these two methods of weight updates are very similar, with the “steady-state” version being slightly poorer. Experiments additional to those reported in Table 1, which are described in the following text, have all been performed for the sake of expediency using the “steady-state” method of updating the weights. These additional results are thus likely to provide an underestimate of the performance of the PC/BC algorithm when using “continuous” learning.

In the standard version of the bars problem, as defined by Földiák (1990), training data consists of 8 by 8 pixel images in which each of the 16 possible one-pixel wide horizontal and vertical bars are independently selected to be present with a probability of $\frac{1}{8}$. A pixel at a location occupied by a bar is said to be “on” and is assigned a value of one, while a pixel at a location not occupied by any bar is said to be “off” and is assigned a value of zero. When tested on this version of the bars problem all three sets of weights learnt to represent all components in every trail, and the network response also correctly represented all components in all 25 trials (Table 1, row 1). A number of other algorithms (*e.g.*, dendritic inhibition and nmfsc) have also been shown (through an analysis of the weights) to be at ceiling performance in this version of the task (Spratling, 2006). For consistency all experiments have been performed using 20000 training cycles. However, the bar components are actually learnt much more quickly. Only 2000 training cycles are required for bars to be represented with 100% reliability when measured using an analysis of the response. For all components to be correctly represented in terms of the analysis of the weights in every trail takes 6000 training cycles. A typical example of the weights learnt by PC/BC when trained on the standard 8 by 8 pixel bars problem are shown in Fig. 6. It can be seen that following training the values of \mathbf{W}^{S1} representing each pixel in a bar have a weight of around $\frac{1}{8}$, and hence, the sum of the afferents into each prediction neuron is very close to unity. In contrast, the feedback weights to represented pixels each have a value very close to unity.

A common variation on the standard bars problem uses 5 by 5 pixel images in which the 10 possible (one-pixel wide) horizontal and vertical bars are independently selected to be present with a probability of $\frac{1}{5}$. Again, the PC/BC algorithm is at ceiling performance on this variation (Table 1, row 2). A number of other algorithms (MCA_3 , $R - MCA_2$, $R - MCA_{NN}$, NN_{slow}) have also been shown (through an analysis of the response) to be 100% reliable at extracting components in this task (Lücke, 2009; Lücke and Sahani, 2007, 2008).

To increase the difficulty of the task, noise can be added to the training images. 10% bit-flip noise was added by changing the value of each pixel in the training set from 0 to 1, or from 1 to 0, with a probability of 0.1. Over 25 trials most, but not all, of the 10 components were correctly represented by the weights. However, an analysis of the response showed 100% accuracy (Table 1, row 3). This discrepancy between the two methods of analysis is not surprising as the noise can result in individual pixels of a bar not being strongly represented by the weights (which will cause failure by the criteria used to analyse the weights). However, the response of the network can still be selective for the individual components (and hence pass the criteria used for analysing the response). In comparison, for an almost identical task Meila and Jordan (2000) report a reliability (measured from an analysis of the network structure) of 95% when using only 2% bit-flip noise. Lücke and von der Malsburg (2004) report

¹A more stringent test would analyse the response generated to test images containing multiple components, as described in Spratling et al. (2009).

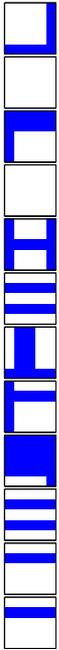
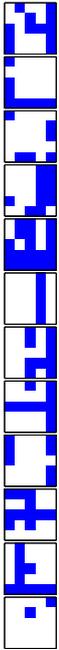
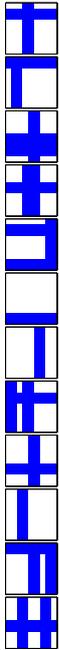
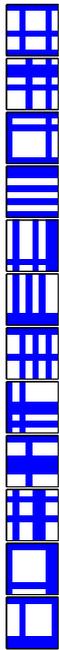
Problem version	Example input patterns	Results for continuous learning				Results for steady-state learning			
		\mathbf{W}^{S1}	\mathbf{V}^{S1}	Average number of components represented by the weights	Reliability of learning measured from response	\mathbf{W}^{S1}	\mathbf{V}^{S1}	Average number of components represented by the weights	Reliability of learning measured from response
Standard 8 by 8		$\frac{16}{16}$	$\frac{16}{16}$	$\frac{16}{16}$	100%	$\frac{16}{16}$	$\frac{16}{16}$	$\frac{16}{16}$	100%
Standard 5 by 5		$\frac{10}{10}$	$\frac{10}{10}$	$\frac{10}{10}$	100%	$\frac{10}{10}$	$\frac{10}{10}$	$\frac{10}{10}$	100%
Noisy 5 by 5		$\frac{9.64}{10}$	$\frac{9.76}{10}$	$\frac{9.8}{10}$	100%	$\frac{9.84}{10}$	$\frac{9.84}{10}$	$\frac{9.84}{10}$	100%
Double width		$\frac{15.96}{16}$	$\frac{15.96}{16}$	$\frac{15.92}{16}$	100%	$\frac{15.84}{16}$	$\frac{15.84}{16}$	$\frac{15.84}{16}$	88%
Fixed number		$\frac{16}{16}$	$\frac{16}{16}$	$\frac{16}{16}$	100%	$\frac{16}{16}$	$\frac{16}{16}$	$\frac{16}{16}$	100%
Unequal		$\frac{16}{16}$	$\frac{16}{16}$	$\frac{16}{16}$	100%	$\frac{16}{16}$	$\frac{16}{16}$	$\frac{16}{16}$	100%

Table 1: Summary of results for the bars problem. The performance of the PC/BC algorithm was measured for six variations of the bars problem. Typical training patterns for each variation are shown. Results were measured using two methods for updating the weights. Results for continuous learning were generated using training images presented for a random number of iterations (uniformly selected from the range 1 to 400) and the weights were updated at every iteration using a learning rate of $\beta = \frac{0.005}{200}$. Results for steady-state learning were generated using training images presented for a fixed number of iterations (200) and the weights updated at the end of this period using a learning rate of $\beta = 0.005$.

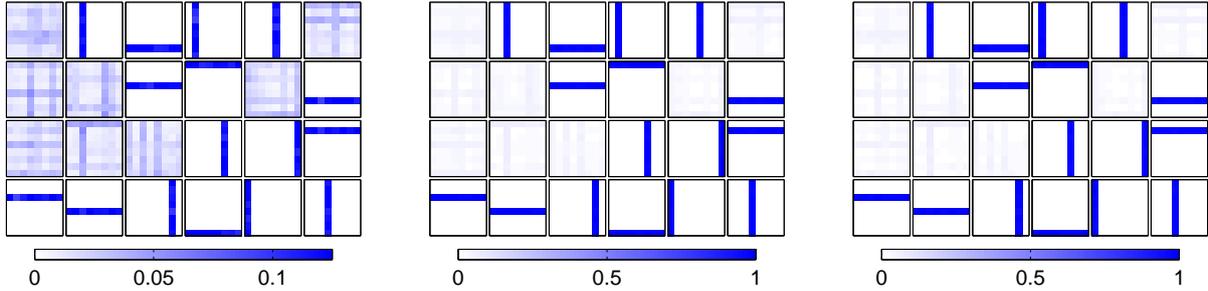


Figure 6: Weights learnt after 10,000 training cycles with the standard 8 by 8 bars problem. The left column shows the W^{S1} weights, centre column the V^{S1} weights, right column the U^{S1} weights.

an algorithm that is 100% reliable with up to 12% bit-flip noise when that algorithm was trained using the same parameters as applied to solving other variations of the bars problem. However, [Lücke and von der Malsburg \(2004\)](#) obtained this result using another variation on the bars problem in which 8 four-pixel wide bars appeared in 16 by 16 pixel images such that parallel bars did not overlap. By constructing each individual bar from 64 pixels the effects of flipping any one bit is much reduced. When PC/BC was applied to learning noisy bars patterns of the type used by [Lücke and von der Malsburg \(2004\)](#) it was found that reliability (measured using an analysis of the response as used by [Lücke and von der Malsburg \(2004\)](#)) was 100% with up to 30% bit-flip noise. [Lücke \(2009\)](#) reports an algorithm that is 100% reliable with up to 38% bit-flip noise using yet another variation on the bars problem in which 16 two-pixel wide bars appeared in 16 by 16 pixel images such that parallel bars did not overlap. One million training cycles were used ([Lücke, 2009](#)) with a new, random, training image being generated at each cycle ([Lücke, personal communication, 2010](#)). Using the same variation of the bars problem, and the same experimental procedure, the PC/BC algorithm was 100% reliable with up to 29% bit-flip noise. However, reliability with higher noise values can be achieved using more prediction neurons in the PC/BC network, see below.

Another more challenging version of the bars problem uses 9 by 9 pixel images in which each of the 16 possible, two-pixel wide, horizontal and vertical bars can be present with a probability $\frac{1}{8}$. In this task, as in the standard bars problem, perpendicular bars overlap; however, the proportion of overlap is increased. Furthermore, neighbouring parallel bars also overlap considerably (by 50%). The analysis of the weights learnt by the PC/BC algorithm (Table 1, row 4) compares well with the same analysis performed on the weights learnt by other algorithms applied to an identical task ([Spratling, 2006](#)). Specifically, the average number of components learnt by the dendritic inhibition algorithm was 15.6, by nmfsc was 15.96, and by nnscc was 16. The analysis of the response (Table 1, row 4) shows that the PC/BC algorithm, in common with two other algorithms (MCA_3 , $R-MCA_2$; [Lücke and Sahani, 2008](#)), was performing at ceiling for this task.

The fixed number of bars variation of the bars problem used 8 by 8 pixel images containing different combinations of 16 possible, one-pixel wide, horizontal and vertical bars. However, in contrast to the standard bars task, in this version each training image was generated from a randomly selected combination of five bars. Therefore, each training image contained exactly five bars. In this case it is not possible for an algorithm to learn by relying on components occasionally appearing in isolation in certain training images. The PC/BC algorithm learnt all components in every trial for this task (Table 1, row 5). Performance of the PC/BC algorithm was also at ceiling when trained using patterns containing exactly 1, 2, 3, and 4 bars. Using training images each containing exactly 6 bars resulted in a deterioration of performance with an average of 15.48 bars represented by the weights, and the reliability of representing (in terms of response) all 16 components being 92%. Experiments with a fixed number of bars have also been performed by [Savin et al. \(2010\)](#), but they provide no report of the reliability of their algorithm on this task. The task used here is also similar to one used in [Lücke and Sahani \(2008\)](#) except that they generated patterns in the standard way (with each bar being selected to be present with a probability of $\frac{1}{8}$) and then removed images that contained fewer than b bars. They report that using $b = 2$ (*i.e.*, removing only patterns containing zero or one bar) resulted in the $R - MCA_2$ having a reliability of only 69%, and that for $b = 3$ each of their algorithms (MCA_3 , $R - MCA_2$, and $R - MCA_{NN}$) failed.

A particularly difficult version of the bars problem is one in which components are unequal in size and components appear with an unequal probability. In the specific task used here training data consisted of 16 by 16 pixel images. Image components consisted of seven one-pixel wide bars and one nine-pixel wide bar in both the horizontal and vertical directions. Hence, as in the standard bars problem, there were eight image components at each orientation and 16 in total. Parallel bars did not overlap, however, the proportion of overlap between the nine-pixel wide bars and all other perpendicular bars was large, while the proportion of overlap between perpendicular

one-pixel wide bars was less than in the standard bars problem. Each vertical bar was selected to be present in an image with a probability of $\frac{1}{8}$ while horizontal bars occurred with a probability of $\frac{1}{32}$. Hence, in this test case half the underlying image components occurred at a different frequency to the other half and two of the components were a different size to the other 14. A PC/BC network with 24 prediction neurons (as has been used in all the preceding experiments) fails to learn all the components in any individual trial. However, quadrupling the number of prediction neurons, to 96, allows all components to be learnt in every trial (Table 1, row 6). For comparison, the average number of components learnt by other algorithms applied to an identical task was 15.72 for the dendritic inhibition algorithm, 14 for nmfsc, and 13.62 for nnscc (Spratling, 2006).

Increasing the number of prediction neurons in the PC/BC network also improves the performance of the learning algorithm on other versions of the bars task. For example, using a network with 30 neurons to learn (using the steady-state activation values) the Double Width bars problem, results in reliability increasing to 100% and the average number of components represented by the weights also increasing to 16 (for each of the \mathbf{W}^{S1} , \mathbf{V}^{S1} , and \mathbf{U}^{S1} weights). Similarly, increasing the number of prediction neurons to 96 when learning the bit-flip noise version of the bars problem employed by Lücke (2009) results in PC/BC learning this task with 100% reliability with up to 35% noise. It is not surprising that the success of the algorithm depends on there being sufficient prediction neurons in the population, since the random initial weights to and from these neurons are the only source of variability. Other algorithms (e.g., Lücke, 2009; Lücke and von der Malsburg, 2004) apply noise to the neural dynamics which can help with searching the solution space. In contrast, here there is no random element in the dynamics in order to allow a fairer comparison with other deterministic algorithms such as nmfsc, nnscc, and dendritic inhibition. With sufficient neurons in the network, the PC/BC algorithm succeeds in learning every component in every trial across all five variations of the noise-free bars problem it has been applied to. Previous algorithms have equalled this performance on some variations of the bars problem, but no individual algorithm has previously come close to this level of performance across all these variations of the bars problem. Furthermore, with sufficient neurons in the network, the PC/BC algorithm can learn with noise levels competitive with the current state-of-the-art (Lücke, 2009).

3.2 Learning Gabor-Like Components from Natural Images using a Single Processing Stage

Several learning algorithms, when applied to natural images, learn elementary components that resemble Gabor functions, and hence, also resemble the RFs of orientation selective simple cells in V1 which are accurately modelled by Gabor functions (Daugman, 1980, 1988; Jones and Palmer, 1987a; Marcelja, 1980). This suggests that the RFs of V1 simple cells are adapted to the statistical regularities of the visual world (Olshausen and Field, 1996b, 1997). Examples of such algorithms include: a linear generative model with the objective of preserving information while producing a sparse code (sparsenet; Harpur, 1997; Olshausen and Field, 1996a, 1997); a mathematically similar algorithm interpreted as an implementation of predictive coding in which the representation is constrained to be a sparse (Rao and Ballard, 1999); a non-negative version of sparsenet (Hoyer, 2003); a closely related algorithm for performing non-negative matrix factorisation (NMF) with an additional constraint applied to make the representation sparse (nmfsc; Hoyer, 2004); a nonlinear generative model with a strict sparseness constraint to limit the number of active neurons (SSC; Rehn and Sommer, 2007); a nonlinear generative model that combines causes using a maximum operation, rather than using summation ($R - MCA_2$; Lücke and Sahani, 2008); a generative model combining two sets of causes representing the identity and appearance of visual elements (Berkes et al., 2009); a generative model in which causes are selected via matching pursuit and which is interpreted as an implementation of predictive coding (Jehee and Ballard, 2009); an algorithm that combines matching pursuit with homeostasis (Perrinet, 2010); another sparse generative model that employs a Helmholtz machine together with mechanisms to enforce sparseness and homeostasis (Weber and Triesch, 2008); models that perform independent component analysis (ICA) on image patches, and hence, attempt to find mutually independent image components that can be used represent images using a factorial code (Bell and Sejnowski, 1997; Hoyer and Hyvärinen, 2000; Van Hateren and van der Schaaf, 1998); ICA performed using a RBM (Teh et al., 2003); an encoder-decoder energy-based model (Ranzato et al., 2007); competitive neural network algorithms in which lateral connections are trained via anti-Hebbian learning rules to decorrelate the node outputs, and hence, produce a sparse code (Falconbridge et al., 2006; Wiltchut and Hamker, 2009); a competitive neural network algorithm employing pre-integration lateral inhibition as the mechanism of competition between the nodes (Hamker and Wiltchut, 2007); a competitive neural network algorithm in which nodes are inhibited by the maximum excitation of other nodes in the network (Lücke, 2009). A common feature of all these algorithms is that the learnt representation is sparse, meaning that any patch of an image is represented by a small number of components, basis vectors, codebook vectors, or active neurons. This sparseness constraint is either imposed explicitly or is the implicit outcome of competition between nodes in a neural network with synaptic weights representing the basis vectors.

The competition performed by the PC/BC algorithm is also of the appropriate form to learn Gabor-like RFs when natural images are used as the input. To demonstrate this, training data was generated using the same set of ten gray-scale images of natural scenes as was used in Olshausen and Field (1996a)². The intensity values of each image were independently scaled to lie in the range [0,1] and each image was then preprocessed by convolving it with a Laplacian of Gaussian (LoG) filter with a standard deviation of 1.5 pixels (generated by the MATLAB command “fspecial”³) to simulate the receptive field structure of centre-surround cells along the pathway leading to primary visual cortex (*i.e.*, in retina and LGN). The positive and negative values produced by the filter were separated into ON and OFF channels, simulating the output of on-centre-off-surround and off-centre-on-surround cells respectively.

The training data consisted of corresponding pairs of 11 by 11 pixel patches from the ON and OFF channels of an image (similar results were obtained for all other patch sizes for which tests were performed: all integer values from 8x8 pixels up to 16x16 pixels). Each patch was taken from one of the 10 images chosen at random and was centred at a randomly selected pixel. A new image patch was chosen at each training cycle. The network was trained for 20000 training cycles. Figure 7 shows the receptive fields learnt by each of the 180 prediction nodes in the network (similar results were obtained for all other network sizes for which tests were performed: 120 nodes, 360 nodes, and 3000 nodes). The weights learnt to each of the 11 by 11 pixels forming the ON channel are shown in the top row of Figure 7 and the weights representing the OFF channel are shown in the centre row of Figure 7. The bottom row of Figure 7 visualises the RF of each neuron, including the effects of the image pre-processing. These RF reconstructions are obtained by convolving the learnt ON and OFF weights with the LoG filter, and then subtracting the resulting OFF channel RF from the ON channel RF.

Figure 8a-c shows in more detail the afferent weights learnt by one prediction neuron (that shown in row one, column two of each sub-figure in Fig. 7). By comparing corresponding sets of weights in the top and middle rows of Fig. 7, and from comparing Fig. 8a with Fig. 8b, it can be seen that inputs from the ON-centre and OFF-centre sub-populations of LGN neurons are spatially segregated, such that weights from ON centre cells (respectively OFF centre cells) are approximately zero in regions of the prediction neuron’s RF that receive strong weights from OFF (respectively ON) centre cells. This segregation of ON and OFF inputs is consistent with neurophysiological data (Jones and Palmer, 1987b; Reid and Alonso, 1995). It has recently been claimed that spatially segregated ON and OFF regions also emerge when a different implementation of Predictive Coding is applied to learning V1 RFs (Jehee and Ballard, 2009). However, in that algorithm the correspondence between ON and OFF inputs at the same spatial location is built into the algorithm. Similarly, Savin et al. (2010) show an example RF with segregated ON and OFF inputs. However, their algorithm separately normalises the total strength of the ON and OFF weights. Hence, in both these previous algorithms, connections from the ON and OFF channels are treated differently which may contribute to the resulting segregation of these sets of connections. In contrast, in the PC/BC algorithm inputs from the ON and OFF channels are treated identically, *i.e.*, both sets of inputs are simply elements of the vector \mathbf{y}^{S0} . Hence, this segregation emerges as a result of the learning algorithm. This also appears to be the case with algorithms based on non-negative matrix factorisation using sparseness constraints (Hoyer, 2003, 2004).

As well as being spatially segregated, the ON and OFF regions learnt by PC/BC are often elongated and aligned in parallel. Such an organisation of afferent inputs was proposed by Hubel and Wiesel (1962, 1968) to underlie the orientation selectivity of V1 simple cells. This idea that a V1 cell receives input from LGN cells with spatially aligned RFs has subsequently be formalised into the proposal that V1 RFs can be modelled as Gabor functions (Daugman, 1980, 1988; Jones and Palmer, 1987a; Lee, 1996; Marcelja, 1980).

The RFs reconstructed from the synaptic weights learnt by PC/BC are well fitted by Gabor functions. The Gabor fit to a single example RF is shown in Fig. 8d, and the fits to the RFs of all the neurons in the model are shown in Fig. 9b. Fitting was performed using a genetic algorithm (Houck et al., 1995) to minimise the normalised mean squared error (NMSE) between the learnt RF and the Gabor fit to the RF:

$$\text{NMSE} = \frac{\sum (W_r - W_g)^2}{\sum W_r^2}.$$

Where W_r are the weights of the learnt RF, and W_g are the weights of the Gabor function fitted to this RF. Fig. 8e shows the difference between the exemplar learnt RF and the Gabor fit to that RF. Fig. 9c shows the distribution of fitting errors across the population of prediction neurons. Across the population of prediction nodes the mean NMSE was 0.16 and the median value was 0.13. Fitting Gabors to the RF reconstructed from the \mathbf{V}^{S1} weights gave almost identical results. The weights learnt by the PC/BC model are thus accurately represented as Gabor functions.

²available from <https://redwood.berkeley.edu/bruno/sparsenet/>

³<http://www.mathworks.com/help/toolbox/images/ref/fspecial.html>

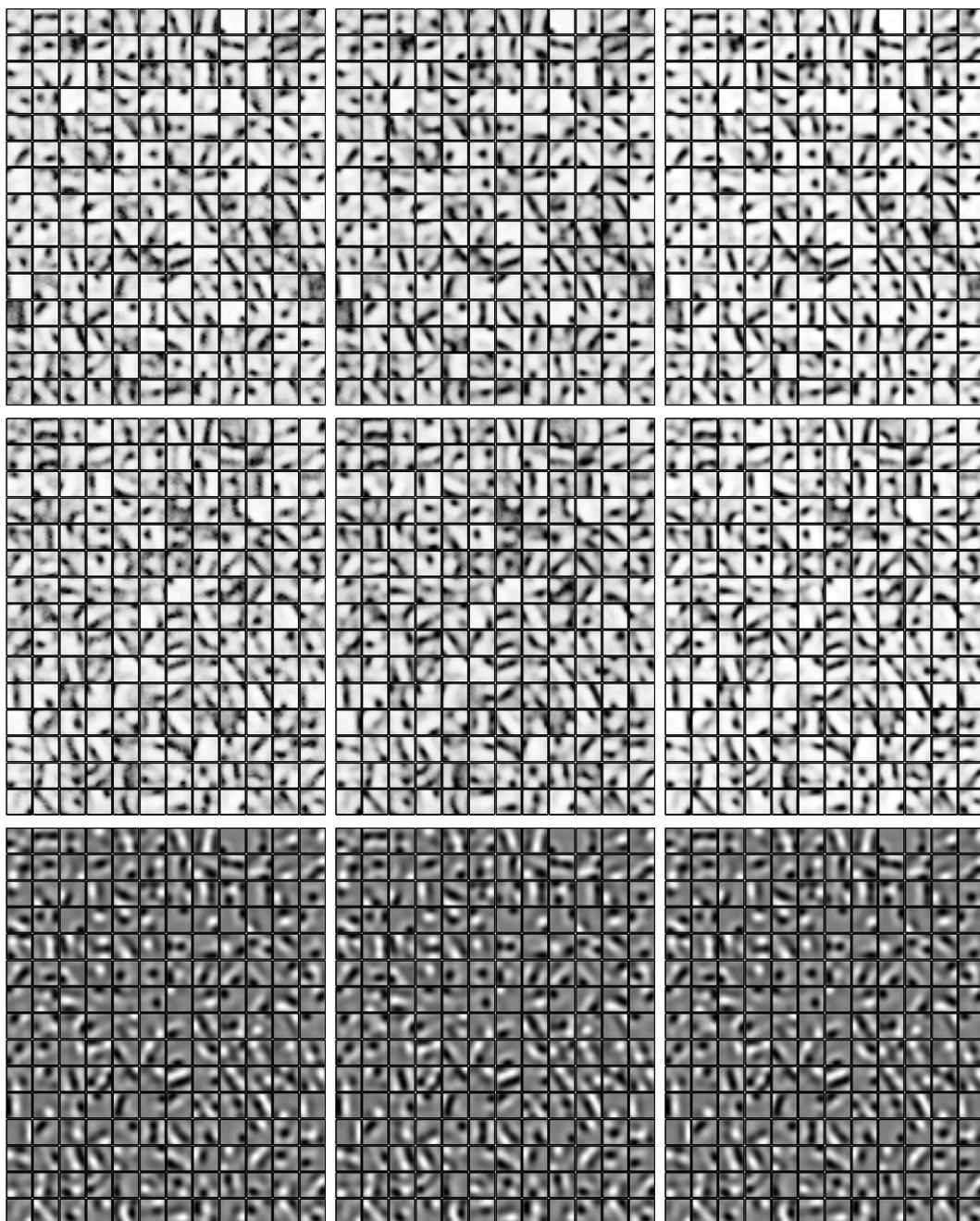


Figure 7: Weights learnt by the PC/BC algorithm when trained on natural image patches. Weights for 180 prediction neurons are shown: left column shows the \mathbf{W}^{S1} weights, centre column the \mathbf{V}^{S1} weights, right column the \mathbf{U}^{S1} weights. Each neuron receives or sends 11×11 connections from/to the ON channel of the input, and 11×11 weights from/to the OFF channel of the input. The two parts of each node's weight vector corresponding to the ON and OFF channels are shown separately in the top row and the middle row respectively (the strength of each individual weight corresponds to the darkness of each pixel, with white pixels representing weights with zero strength). The bottom row shows a composite image of the ON weights minus the OFF weights after each set of weights have been convolved with the filter used to pre-process the images. This shows the effective strength of the connection received by a node from each input pixel, *i.e.*, the reconstructed receptive field of each node. Mid-gray represents zero, lighter pixels represent negative values and darker pixels represent positive values. Each RF has been scaled separately to occupy the full range of gray values.

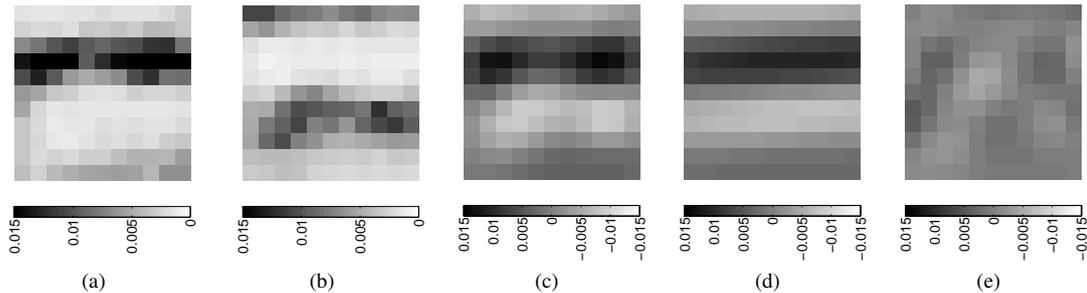


Figure 8: Example afferent weights (*i.e.*, \mathbf{W}^{S1} weights) learnt by a single prediction neuron (a) ON channel weights, (b) OFF channel weights, (c) the RF reconstructed from the weights, (d) Gabor fit to the reconstructed RF, (e) difference between the learnt RF and the Gabor fit.

In physiological experiments the synaptic weights are unknown and so the RFs of V1 cells are estimated from their response properties. One method is through reverse correlation (Ringach and Shapley, 2004). During a reverse correlation experiment, a stream of images is presented to the animal while the response of a neuron is recorded. The correlation between strong neural responses and the image that was present at some (variable) time preceding the response enables the spatio-temporal receptive field of the recorded neuron to be reconstructed. Ringach (2002) employed sinusoidal grating images with various spatial frequencies, spatial phases and orientations taken from a two dimensional Hartley basis function set (Ringach et al., 1997) to estimate the RFs of cells in macaque V1. The same procedure was used to reconstruct the RFs learnt by the PC/BC model. Specifically, at each iteration of the PC/BC algorithm, for a total of $T = 262440$ iterations, a different randomly selected image from the Hartley set was presented to the network. The reconstructed RFs were determined by calculating the cross correlation between the stimulus and the response at a particular time-lag, τ , between the iteration in which the stimulus was present and the iteration at which the response was recorded:

$$RF(\tau) = \frac{\sum_{t=1+\tau}^T \mathbf{r}_t \bar{I}_{t-\tau}}{T - \tau}$$

Where \bar{I} is the input image normalised such that the mean luminance is zero, and \mathbf{r} is the time-varying response of the particular prediction neuron whose RF is being reconstructed. Reconstructed RFs are shown for the time-lag at which the variance of the reconstructed RF was greatest, which was at $\tau = 0$. $RF(\tau)$ is effectively the mean of all stimuli weighted by the evoked response of the recorded neuron at time-lag τ . This is equivalent to the spike-triggered average (STA) method used in the neurophysiological literature, but adapted to work with rate-coded, rather than spiking, neurons.

The RFs reconstructed using reverse correlation are shown in Fig. 9d. Comparing Figs. 9a and 9d, it can be seen that the RFs reconstructed from the learnt weights are very similar to the RFs reconstructed via reverse correlation with sinusoidal gratings. The RFs reconstructed via reverse correlation are also well fitted by Gabor functions. The fitted Gabors are shown in Fig. 9e and the distribution of fitting errors is shown in Fig. 9f. Across the population of prediction nodes the mean NMSE was 0.17 and the median value was 0.13.

The quality of the fit between the learnt RFs and the Gabor functions is difficult to assess in relation to existing results since previous work, in both the neurophysiological and modelling literature, has not reported overall fit errors. The only exception is (Berkes et al., 2009) which reports a very similar distribution of fit error to that reported here (compare Fig. 9f with Fig. 8 in Berkes et al., 2009). Berkes et al. (2009) compare the Gabor fits achieved for their model RFs with those reported for neurophysiological data by DeAngelis et al. (1999). However, this comparison is not entirely justified since DeAngelis et al. (1999) report Gabor fits to 2D spatio-temporal RFs, whereas Berkes et al. (2009) report fits to 2D spatio-spatio RFs.

The parameters of the fitted Gabor functions can be used to analyse the properties of the learnt RFs. For this purpose the fits to the RFs reconstructed using reverse correlation were used and the 24 nodes with the worse fit (a NMSE greater than twice the mean NMSE) were excluded from this analysis. Figure 9g shows the joint distribution of spatial frequency and orientation preference for the fitted Gabors. As with several previous models (*e.g.*, Lücke, 2009; Olshausen and Field, 1997), the spatial frequency is fairly narrowly distributed while orientation is uniformly distributed. Fig. 9h shows the distribution of RF “shapes” parametrised in terms of the product of the frequency (f) of the fitted Gabor and the standard deviation of the fitted Gabor along its width (σ_x) and length (σ_y). For comparison, the distribution of the same values measured from Gabor functions fitted to the RFs of V1 simple cells measured using reverse correlation (Ringach, 2002) is also shown in Fig. 9h. The

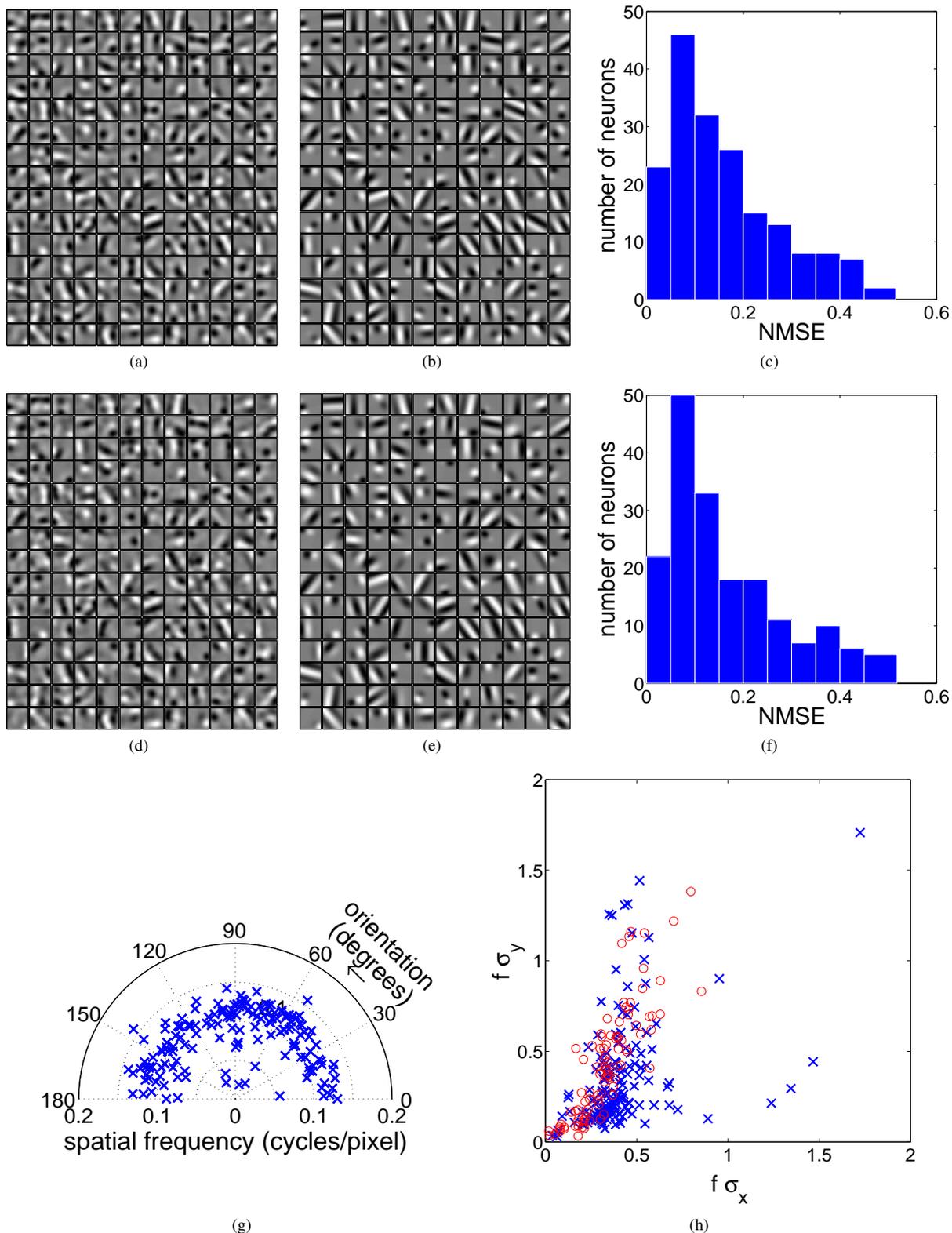


Figure 9: (a) Learnt RFs reconstructed from the \mathbf{W}^{S1} weights, as shown in Fig. 7 bottom left. (b) Gabor fits to the learnt RFs shown in (a). (c) Distribution of Gabor fit errors for the fits shown in (b). (d) Learnt RFs reconstructed using reverse correlation. (e) Gabor fits to the learnt RFs shown in (d). (f) Distribution of Gabor fit errors for the fits shown in (e). (g) Distribution of spatial frequency and orientation preference for the Gabor fits to the RFs learnt by PC/BC. (h) Distribution of receptive field shapes for the Gabor fits to the RFs learnt by PC/BC (crosses) and for the Gabor fits to RFs measured in macaque V1 (circles) (adapted from Ringach, 2002, Fig. 4, using data provided at <http://web.mac.com/darioringach/lab/Data.html>).

distribution of RF shapes measured in V1 appears to form a subset of the distribution measured from the RF learnt by the model. Specifically, the model shows a similar diversity of RF shapes as V1, but with the addition of some RFs that are wider than they are long. In contrast to the good correspondence between the distribution of RF shapes produced by PC/BC, and those measured in V1, several previous algorithms which learn Gabor-like RFs when trained using natural images (*e.g.*, [Bell and Sejnowski, 1997](#); [Olshausen and Field, 1996a, 1997](#); [Van Hateren and van der Schaaf, 1998](#)) have been shown ([Lücke, 2009](#); [Rehn and Sommer, 2007](#); [Ringach, 2002](#); [Wiltschut and Hamker, 2009](#)) to fail to successfully learn the “blob-like” RFs represented by points near the origin of Fig. 9h. Other algorithms do successfully simulate this aspect of the neurophysiology (*e.g.*, [Berkes et al., 2009](#); [Hamker and Wiltschut, 2007](#); [Rehn and Sommer, 2007](#); [Wiltschut and Hamker, 2009](#)).

Although the PC/BC algorithm does not explicitly enforce sparseness, the activity of the prediction neurons in response to natural image patches is highly sparse. To quantify sparsity, three popular measures were used.

- Kurtosis ([Field, 1994](#); [Lehky et al., 2005](#); [Olshausen and Field, 2004](#); [Willmore and Tolhurst, 2001](#)):

$$S_K = \frac{1}{p} \sum_{k=1}^p \frac{(y_k - \bar{y})^4}{\sigma^4} - 3$$

Where \bar{y} and σ are the mean and standard deviation of the response. $S_K > 0$ for sparse distributions.

- The specific implementation used by [Vinje and Gallant \(2000\)](#) of the Rolls-Tovee measure ([Lehky et al., 2005](#); [Olshausen and Field, 2004](#); [Rolls and Tovee, 1995](#); [Willmore and Tolhurst, 2001](#)):

$$S_{RT} = \frac{1 - \left(\sum_{k=1}^p \frac{y_k}{p} \right)^2 / \left(\sum_{k=1}^p \frac{y_k^2}{p} \right)}{1 - (1/p)}$$

S_{RT} ranges from 0 to 1, with higher values corresponding to sparser distributions.

- The measure proposed by [Hoyer \(2004\)](#):

$$S_H = \frac{\sqrt{p} - (\sum_{k=1}^p y_k) / \sqrt{\sum_{k=1}^p y_k^2}}{\sqrt{p} - 1}$$

S_H ranges from 0 to 1, with higher values corresponding to sparser distributions.

To calculate the (population) sparseness, y_k in each of the above equations was the time-averaged (over 20 iterations) response of the k th neuron to a test stimulus, and p was the number of neurons in the population. Sparseness was measured using 1000 randomly generated image patches. A separate sparsity measure was calculated for each image, providing a distribution of population sparsity, as shown in Fig. 10(top-row). The mean value of S_K measured for the PC/BC model (56.4) compares to values of 10.04 for ICA ([Bell and Sejnowski, 1997](#)), 20 for sparsenet ([Olshausen and Field, 1996a](#)), 21 to 196 (depending on the parameters of the model) measured by [Wiltschut and Hamker \(2009\)](#), and 287.3 for SSC ([Rehn and Sommer, 2007](#)). The mean value of S_H measured for the PC/BC model (0.78) compares to values of 0.72 to 0.92 (depending on the parameters of the model) measured by [Wiltschut and Hamker \(2009\)](#).

Rather than the population sparseness, most neurophysiological experiments measure the selectivity of each neuron’s response to different stimuli ([Lehky et al., 2005](#)), sometimes referred to as the lifetime sparseness ([Willmore and Tolhurst, 2001](#)). To calculate the (lifetime) selectivity, y_k in each of the above equations was the time-averaged (over 20 iterations) response of a neuron to the k th stimulus, and p was the number of stimuli in the test set. Selectivity was measured using 1000 randomly generated image patches. A separate selectivity measure was calculated for each neuron, providing a distribution of selectivities, as shown in Fig. 10(bottom-row). The mean value of S_K measured for the PC/BC model (150.1) compares to a value of 2.8 measured in V1 ([Lehky et al., 2005](#)). The mean value of S_{RT} measured for the PC/BC model (0.95) compares to values measured in V1 of 0.78 ([Tolhurst et al., 2009](#)) and, depending on stimulus size, between 0.41 and 0.62 ([Vinje and Gallant, 2000](#)). Note that the model neurons do not have a spontaneous firing rate, and hence, the sparsity and selectivity measured for the model would be expected to be greater than that measured in cortex.

An efficient code requires that neural responses are statistically independent, so that the response of one neuron does not provide any information about the response of another. One method to visualise the degree of independence between two neurons is to plot a conditional probability histogram ([Schwartz and Simoncelli, 2001](#); [Simoncelli and Olshausen, 2001](#); [Wiltschut and Hamker, 2009](#)). Such a histogram plots the joint response distribution of the two neurons, as shown in Fig. 11a. Each column in the histogram indicates the probability

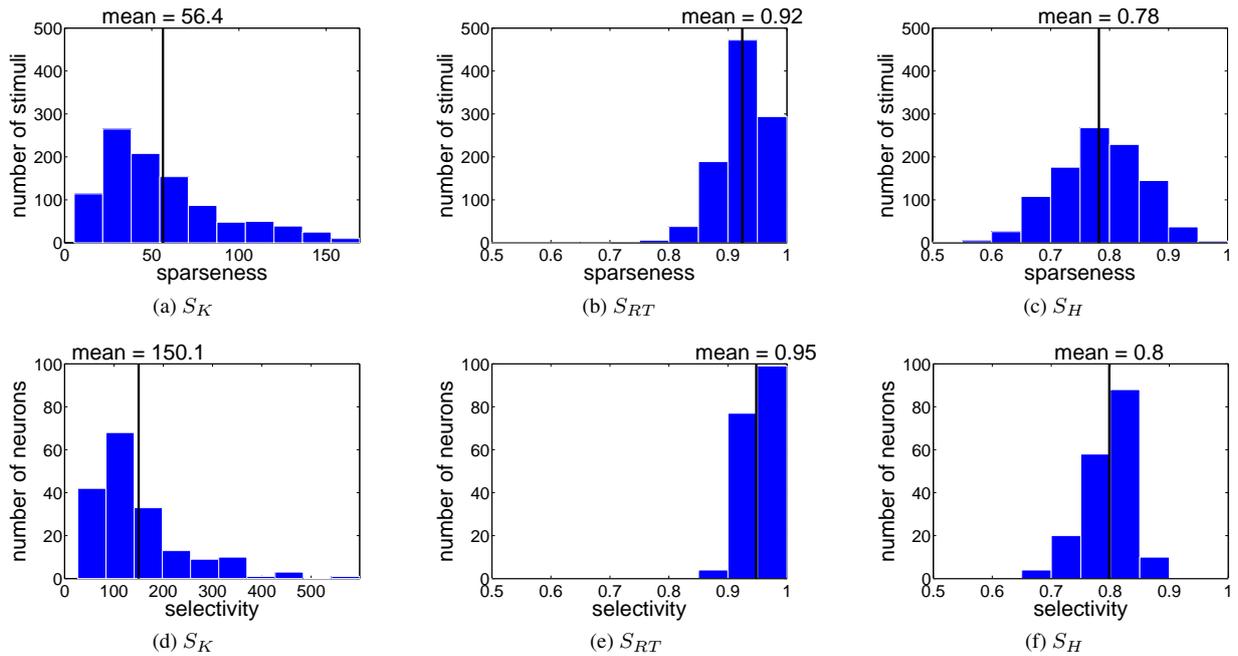


Figure 10: Top-row: distribution of (population) sparsity measured for 1000 image patches. Sparsity measured using (a) Kurtosis, (b) Rolls-Tovee method, (c) Hoyer’s method. Bottom-row: distribution of (lifetime) selectivity measured for 180 prediction neurons. Selectivity measured using (d) Kurtosis, (e) Rolls-Tovee method, (f) Hoyer’s method.

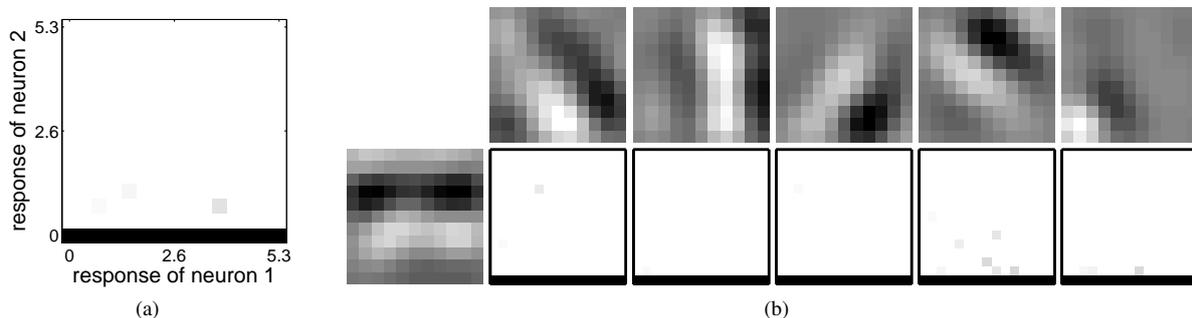


Figure 11: Conditional probability histograms of responses to natural image patches. (a) An example histogram. The response of each neuron is binned into 15 equal-sized bins. Each pixel shows the frequency with which each pair of binned responses co-occur: a dark pixel indicates a high conditional probability. Each column is independently rescaled to fill the full range of intensity values. (b) Typical conditional probability histograms for five pairs of neurons. The RF of neuron 1 is shown on the left and those of neuron 2 are shown above each histogram.

that neuron 2 generates an output of the given magnitude given that neuron 1 has generated an output of the magnitude shown on the abscissa. For independence, all columns of the histogram should be identical (Schwartz and Simoncelli, 2001), which can be quantified by saying that the variance across all columns of the variance along each column should be small (Wiltschut and Hamker, 2009). To test the independence of the encoding learnt by the PC/BC model, the steady-state responses of all 180 prediction neurons were recorded to the presentation of 20000 randomly selected image patches. Figure 11b shows typical examples of the conditional probability histograms generated for the example neuron shown in Fig. 8 and five other randomly selected neurons. The variance across all columns of the variance along each column was calculated for every pair of neurons, and the mean value was 2.6×10^{-04} , indicating that (as appears to be the case for the examples shown in Fig. 11) across the population there was a high degree of independence in the coding of image patches.

To be considered an efficient code, the response of the network must also accurately encode the input it re-

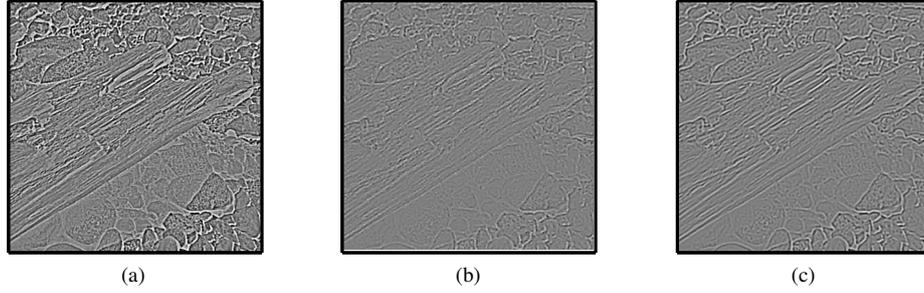


Figure 12: Image reconstruction. (a) The original image pre-processed, as described above, using a LoG filter. (b) The reconstruction generated by PC/BC using non-overlapping 11x11 pixel patches. (c) The reconstruction generated by PC/BC using overlapping 11x11 pixel patches. In each case the OFF channel is subtracted from the ON channel, hence dark intensity values represent strong ON channel response, while light intensity values represent strong OFF channel response.

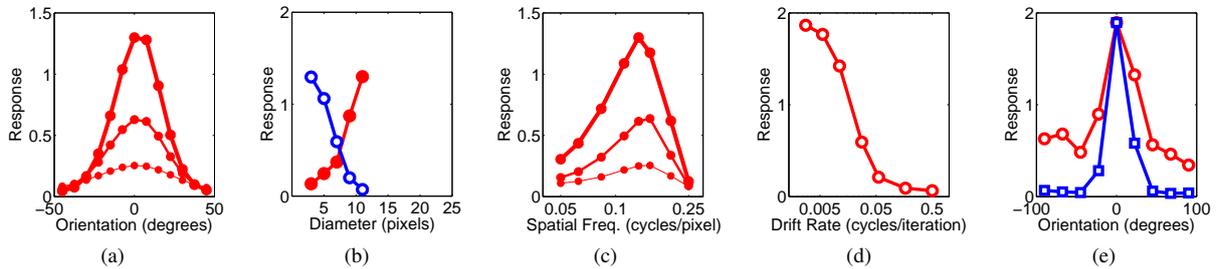


Figure 13: Response properties for a typical model prediction neuron after training on natural image patches. The recorded neuron is the one whose RF is shown in Fig. 8. (a) Response as a function of grating orientation relative to the neuron’s preferred orientation. The thickness of each line corresponds to the contrast of the stimulus used: 25% (thin), 50% (medium) and 100% (thick). Compare with [Skottun et al. \(1987, Fig. 3a\)](#) and [Spratling \(2010, Fig. 5a\)](#). (b) Response as a function of the diameter of a circular grating (filled circles) and as a function of the inner diameter of an annular grating (open circles). Compare with [Jones et al. \(2001, Fig. 1\)](#) and [Spratling \(2010, Fig. 5b\)](#). (c) Response as a function of grating spatial frequency with variable grating contrast. The thickness of each line corresponds to the contrast of the stimulus used: 25% (thin), 50% (medium) and 100% (thick). Compare with [Skottun et al. \(1987, Fig. 4a\)](#) and [Spratling \(2010, Fig. 5e\)](#). (d) Response as a function of grating temporal frequency. Compare with [Freeman et al. \(2002, Fig. 3c\)](#) and [Spratling \(2010, Fig. 5f\)](#). (e) Response as a function of the orientation of a single grating (squares) and as a function of the orientation of a mask grating additionally superimposed on an optimally orientated grating (circles). Compare with [Schwartz and Simoncelli \(2001, Fig. 5\)](#) and [Spratling \(2010, Fig. 6a\)](#).

ceives. To test this the trained network was presented with 20000 randomly selected image patches and the NMSE was calculated between the input to the PC/BC processing stage and the reconstruction of the input estimated using the steady-state responses of the prediction neurons. The mean NMSE was 0.36 and the median NMSE was 0.35. To visualise the coding quality, the reconstructions of individual patches from one of the training images were combined together to reconstruct the whole image (this method has been used previously by [Rehn and Sommer, 2007](#); [Wiltscut and Hamker, 2009](#)). Figure 12a shows the original pre-processed image. Figure 12b shows the reconstruction using non-overlapping patches. Figure 12c shows the reconstruction when reconstructions from every possible 11x11 pixel patch are combined together, taking the average pixel value for overlapping reconstructions. In both cases the reconstructions were calculated using the steady-state responses (generated after 200 iterations) of all the prediction neurons. Not surprisingly, the reconstruction with non-overlapping patches shows some significant discontinuities at the borders of the patches, whereas the reconstruction using (many more) overlapping image patches is far superior.

In [Spratling \(2010\)](#) it was shown that a PC/BC model of V1 in which the receptive fields of the prediction neurons were predefined using perfectly uniform and regularly spaced Gabor functions could successfully simulate the response properties of V1 simple cells. Several experiments described in [Spratling \(2010\)](#) were repeated using

the RFs learnt by the network described here, to confirm that V1 response properties can be successfully simulated using the weights learnt by PC/BC when trained using natural images. Results of these experiments are shown in Figure 13. These results show that the trained network can reproduce all (but one) of the major response properties of V1 cells, namely: contrast invariant orientation tuning (Fig. 13a) with an tuning width typical of that found in V1 (Van Hooser, 2007), contrast invariant spatial frequency tuning (Fig. 13c) with a band-pass profile as is typical for V1 (Van Hooser, 2007), temporal frequency tuning (Fig. 13d) that shows strong suppression at high frequencies as is found in V1 (Freeman et al., 2002), and cross-orientation suppression (Fig. 13e). The current model fails to show surround suppression. However, this is due to neurons in the model all having RFs restricted to the same small 11x11 patch of image, and hence, any parts of an image beyond this 11x11 pixel patch are entirely ignored by the model, and therefore, can not result in any changes in response. Specifically, there are no neurons with RFs in the surround of the recorded neuron that can generate surround suppression. Figure 15b demonstrates that learnt weights do give rise to surround suppression when there are prediction neurons with RFs in the surround of the recorded neuron.

3.3 Learning Gabor-Like and Corner-Like Components from Natural Images using a Two-Stage Hierarchy

A two-stage PC/BC hierarchy was trained using patches taken from natural images. Stage 1 consisted of nine sub-regions each containing 48 prediction neurons. Each sub-region had an RF restricted to an 11x11 pixel patch of image. The RFs of different sub-regions were arranged on a 3 by 3 grid to evenly cover a 23 by 23 patch of image. Hence, the maximum allowable extent of the RFs of neurons in neighbouring sub-regions overlapped. Stage 2 consisted of 180 prediction neurons. All prediction neurons in stage 2 could receive input from all the prediction neurons in all sub-regions of stage 1. Hence, prediction neurons in stage 2 had RFs that could cover the whole of the 23 by 23 pixel image patch.

The training procedure was identical to that described in section 3.2, except that 23 by 23 pixel image patches were used. Figure 14a shows the receptive fields learnt by each of the 48 prediction in the sub-region of stage 1 with RFs restricted to the centre of the image patch. These RF reconstructions were obtained by convolving the learnt ON and OFF weights with the LoG filter used to simulated LGN processing, and then subtracting the resulting OFF channel RF from the ON channel RF. The learnt RFs appear to be Gabor-like.

It is possible to assess the RF properties of a prediction neuron in the first processing stage by reconstructing its RF from the learnt weights (as in Figure 14a). For stage 1 neurons, the RF reconstructed from the weights provides an accurate prediction of the RF reconstructed from the response properties of the neuron (compare Fig. 9a with Fig. 9d). This is the case because the LGN has been modelled as a linear filter bank. However, beyond the first processing stage non-linear processes occurring in the preceding stage(s) of the PC/BC hierarchy mean that (linear) RFs reconstructed from the weights are a poor indication of the stimulus selectivities of prediction neurons in higher levels. In order to assess the response properties of the neurons in the second stage of the hierarchy, the stimulus set used by Ito and Komatsu (2004) to assess the response properties of cells in cortical area V2 was used. This stimulus set consists of 66 angle stimuli constructed from all combinations of two bars presented at angles 0 to 330° in steps of 30°, plus the twelve single bar stimuli at each of these orientations.

Figures 14b and 14c show the strength of the response of two example prediction neurons to the full range of stimuli in the test set, after training with random patches taken from natural images. These two examples have been chosen because they have similar response profiles to two example V2 cells shown in Ito and Komatsu (2004). As with the neurons in V2, the prediction neurons in the second stage of the PC/BC hierarchy, showed a wide range of selectivities. Some neurons were selective to stimuli formed by the conjunction of two short bars to form an angle stimulus or an elongated bar (e.g., Fig. 14b), while other neurons were selective to one short bar, but with the magnitude of response modulated by the angle of the second bar (e.g., Fig. 14c).

To assess the degree of selectivity (i.e., the tuning width) for angle stimuli, Ito and Komatsu (2004) measured the peak response area for each tested neuron in V2. The peak response area was measured from the response profile (like the examples shown in Fig. 14b and c) as the size of the continuous region surrounding the peak response for which the response was $\geq 50\%$ of the peak response. The distribution of peak response areas measured for the 180 neurons in the second stage of the PC/BC model is similar to that recorded for neurons in V2 (Fig. 14d). The median peak response area in the model was 7 which compares to a median response area of 8 in V2 (Ito and Komatsu, 2004). In comparison, the median peak response area for the 432 prediction neurons in the first stage of the PC/BC model was 10. Tuning thus becomes slightly narrower from stage 1 to stage 2, and hence, the neurons in the second stage are more selective for stimuli in the angles stimulus set.

To assess the magnitude of the response to angles, composed of two bars, in comparison to that in response to

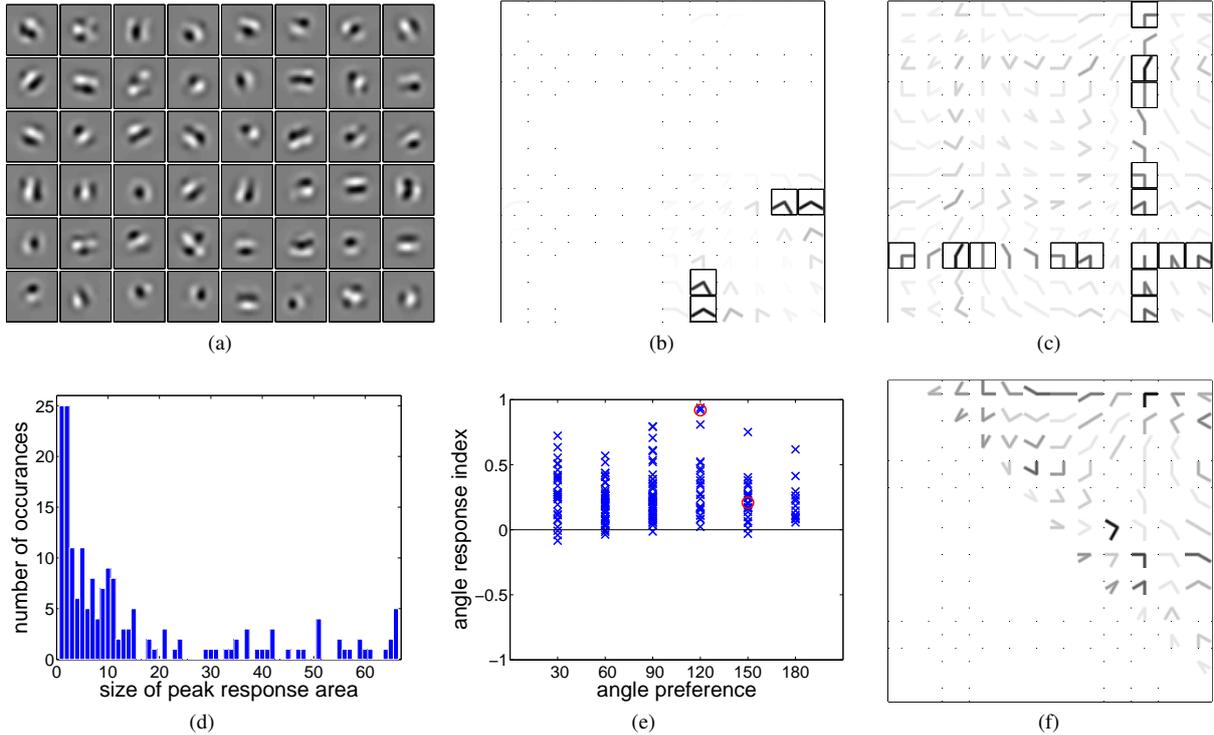


Figure 14: Weights learnt by the hierarchical PC/BC algorithm when trained on natural image patches. (a) Learnt RFs reconstructed from the \mathbf{W}^{S1} weights for the 48 prediction neurons in the central sub-region of the first processing stage. (b) and (c) Example “response profiles” for two prediction neurons in the second processing stage (compare with Ito and Komatsu, 2004, Fig. 2c-f and Fig. 3c-f). Each square in the 12 by 12 grid shows an angle stimulus, the darkness of the stimulus shows the strength of the neuron’s response to that input. Borders around grid locations indicate stimuli for which the response was $\geq 50\%$ of the maximum response. For consistency with Ito and Komatsu (2004), responses below the leading diagonal are a mirror image of those above the leading diagonal. In contrast to Ito and Komatsu (2004), responses to individual bars are shown on the leading diagonal rather than in a separate plot. (d) The distribution of peak response area across the population of 180 prediction neurons in the second processing stage (compare with Ito and Komatsu, 2004, Fig. 5). (e) The angle response index, plotted against preferred angle, for each prediction neuron in the second processing stage (compare with Ito and Komatsu, 2004, Fig. 4c). The circled values, at angle preferences of 120 and 150, are those for the two nodes whose response profiles are shown in (b) and (c) respectively. (f) Distribution of preferred angle across the population of 180 prediction neurons in the second processing stage (compare with Ito and Komatsu, 2004, Fig. 4a). The darkness of the stimulus corresponds to the count of the number of neurons for which that stimulus generated the maximal response.

a single bars, Ito and Komatsu (2004) measured the angle response index (ARI):

$$ARI = \frac{R_{angle} - R_{bar}}{R_{angle} + R_{bar}}$$

Where R_{angle} and R_{bar} are the maximum responses of the neuron across all angle stimuli and all bar stimuli, respectively. The ARIs, plotted as a function of preferred angle, for the prediction neurons in the second stage of the PC/BC model are shown in Fig. 14e. For V2 cells, the ARI was mostly confined to the range -0.33 to +0.33 indicating that neurons had similar maximal responses to angles and bars. For the model, there are far fewer neurons with negative ARI and many neurons with highly positive ARI. Hence, the neurons in the model have a greater preference for angled stimuli over bar stimuli than seems to be the case in V2.

To measure the ability of each stage in the hierarchy to discriminate between different stimuli, the method proposed by Wiltschut and Hamker (2009) was used. For each processing stage the population response was recorded to each stimulus in the angles stimulus set. For each pair of stimuli the normalised cross-correlation (NCC) of the corresponding population response vectors was calculated. The NCC is equal to the cosine of the angle between the vectors, it thus ranges from zero for population response vectors that are orthogonal, up to a

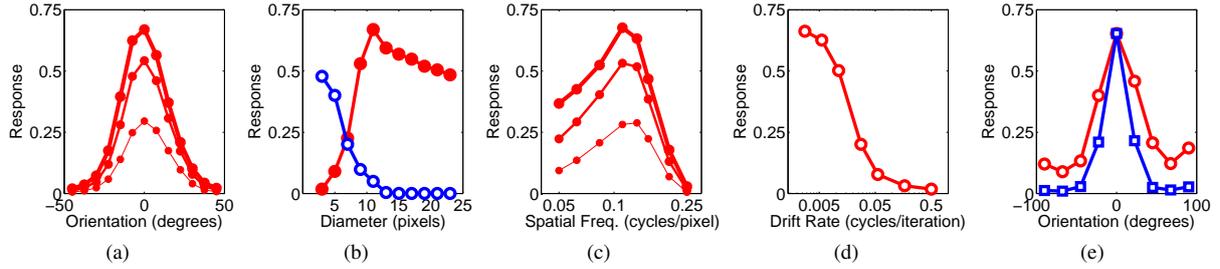


Figure 15: Response properties for a typical model prediction neuron, in the first stage of a 2-stage hierarchy, after training on natural image patches. The recorded neuron is the one whose RF is shown in the top-left of Fig. 14a. The format of the figure is identical to, and described in the caption of, Fig. 13.

value of one for population response vectors that are identical up to a scaling factor. The mean NCC for all pairs of stimuli was calculated as 0.63 in stage 1, and 0.55 in stage 2. The second stage of the hierarchy is thus slightly more able to discriminate between the different stimuli in the test set.

The response preferences of the prediction neurons in the second processing stage provided a fairly even coverage of all the stimuli in the angles stimulus set (including the straight bars). Virtually all the stimuli in the data set were the preferred stimulus (*i.e.*, the one generating the maximum response) for at least one neuron. To illustrate this, Figure 14f shows the number of times each stimulus was the preferred stimulus across the population of prediction neurons in stage 2. Neurons in V2 were also found to provide a full coverage of the stimulus set (Ito and Komatsu, 2004).

The response properties of the neurons in the first stage of the hierarchical network were tested. As shown in Figure 15, results were very similar to those found in section 3.2, except that the prediction neurons in sub-regions surrounding the sub-region of the recorded neuron gave rise to iso-oriented surround suppression for large optimally oriented gratings (Fig. 15b).

4 Discussion

Like many previous algorithms, PC/BC employs reciprocal feedforward and feedback connections. Unlike these previous algorithms PC/BC learns *both* these sets of connections *simultaneously* using only information that is locally available at each synapse. Other aspects of PC/BC that make it more biologically plausible than many preceding algorithms are that it uses on-line (rather than batch) learning, it uses only non-negative firing rates and weights, and the magnitude of the weights are self-limiting avoiding the need for normalisation. Furthermore, unlike most previous learning algorithms PC/BC can learn when the weights are modified at each step in the activation dynamics, hence, it does not need to restrict learning to times when activity has reached a steady-state. It is also not necessary to reset the activity of the network between stimulus presentations. As a consequence, the algorithm does not require any knowledge of when the input stimulus changes, and can learn when stimuli are presented for random durations.

When the training data consists of artificially generated images containing bars patterns, the PC/BC algorithm accurately and reliably learns the elementary components from which these images were generated (section 3.1). With sufficient neurons in the network, the PC/BC algorithm equals or exceeds the performance of all previous algorithms on several variations of the noise-free bars problem and it is competitive with the state-of-the-art on the noisy bars problem.

When the training data consists of patches taken from natural images, the PC/BC algorithm learns elementary components that resemble Gabor functions, and hence, also resemble the RFs of orientation selective simple cells in V1 (section 3.2). The RFs learnt by PC/BC have shapes covering a range that is in good agreement with the properties of cortical V1 cells (Ringach, 2002). Furthermore, consistent with the neurophysiological data (Jones and Palmer, 1987b; Reid and Alonso, 1995), the weights learnt by PC/BC form spatially segregated ON and OFF sub-regions. It has previously been shown that, when the prediction neurons in PC/BC are given Gabor RFs, this model successfully simulates a very wide range of V1 response properties including orientation tuning, size tuning, spatial frequency tuning, temporal frequency tuning, cross-orientation suppression, and surround suppression (Spratling, 2010, 2011). Very similar response properties are produced using the weights learnt by PC/BC when trained using natural images (sections 3.2 and 3.3). Hence, unlike most previous algorithm which account for the formation of V1 RFs, the PC/BC model also accounts for the response properties of V1 once those RFs have been learnt.

Previous models of V1 response properties (e.g., Adorján et al., 1999; Busse et al., 2009; Carandini et al., 2002; Cavanaugh et al., 2002; Dagoi and Sur, 2000; Priebe and Ferster, 2006; Sceniak et al., 1999; Schwabe et al., 2006; Somers et al., 1995; Stetter et al., 2000) can be categorised as descriptive models or structural models (Spratling, 2011). Such models characterise the behaviour of the system being modelled (“what” it does) and consider how the underlying mechanisms give rise to the observed behaviour (“how” the system operates), but they do not provide a computational theory to explain “why” the system operates in the way it does. In contrast, PC/BC provides a computational theory of V1 response properties. It proposes that V1 performs a form of predictive coding. Specifically, that V1 *employs* a set of elementary image components to represent input stimuli efficiently (using a small number of active components) and accurately (with a small reconstruction error).

These computational principles are shared by a large number of learning algorithms: algorithms that propose that the cortex *learns* elementary image components that can represent input stimuli efficiently and accurately. Many of these existing learning algorithms when applied to natural images learn RFs that resemble Gabor functions (reviewed in section 3.2). However, these previous algorithms have failed to simulate the response properties of V1 cells once the RFs have formed⁴. In other words, previous algorithms have proposed learning rules that are successful at simulating the formation of V1 RFs, but have not proposed activation rules that are capable of simulating the behaviour of V1 cells. PC/BC is the first algorithm to extend efficient coding theory, which has been successful in explaining learning in V1, to also explain the subsequent behaviour of V1. By successfully learning V2 RFs in a hierarchical model (section 3.3), PC/BC in common with other recent models (e.g., Lee et al., 2008; Malmir and Ghidary, 2010; Nuding and Zetsche, 2007) also extends this computational theory to explain the formation of RFs beyond V1.

Acknowledgements

Thanks to three anonymous referees for insightful comments on earlier drafts of this article that have led to significant improvements. This work was funded by the Engineering and Physical Sciences Research Council grant number EP/D062225/1.

References

- Adorján, P., Levitt, J. B., Lund, J. S., and Obermayer, K. (1999). A model for the intracortical origin of orientation preference and tuning in macaque striate cortex. *Vis. Neurosci.*, 16:303–18.
- Albrecht, D. G. and Geisler, W. S. (1991). Motion selectivity and the contrast-response function of simple cells in the visual cortex. *Vis. Neurosci.*, 7:531–46.
- Arel, I., Rose, D. C., and Karnowski, T. P. (2010). Deep machine learning: A new frontier in artificial intelligence research. *IEEE Computational Intelligence Magazine*, 5(4):13–8.
- Attneave, F. (1954). Informational aspects of visual perception. *Psychol. Rev.*, 61:183–93.
- Barlow, H. (2001). Redundancy reduction revisited. *Network: Computation in Neural Systems*, 12:241–53.
- Barlow, H. B. (1994). What is the computational goal of the neocortex? In Koch, C. and Davis, J. L., editors, *Large-Scale Neuronal Theories of the Brain*, chapter 1, pages 1–22. MIT Press, Cambridge, MA.
- Bell, A. J. and Sejnowski, T. J. (1997). The ‘independent components’ of natural scenes are edge filters. *Vision Res.*, 37:3327–38.
- Berkes, P., Turner, R. E., and Sahani, M. (2009). A structured model of video reproduces primary visual cortical organisation. *PLoS Comput. Biol.*, 5(9):e1000495.
- Busse, L., Wade, A. R., and Carandini, M. (2009). Representation of concurrent stimuli by population activity in visual cortex. *Neuron*, 64(6):931–42.
- Callaway, E. M. (1998). Local circuits in primary visual cortex of the macaque monkey. *Annu. Rev. Neurosci.*, 21:47–74.
- Carandini, M. (2004). Receptive fields and suppressive fields in the early visual system. In Gazzaniga, M. S., editor, *The Cognitive Neurosciences, 3rd edition*, pages 313–26. MIT Press, Cambridge, MA.
- Carandini, M. and Heeger, D. J. (1994). Summation and division by neurons in primate visual cortex. *Science*, 264(5163):1333–6.
- Carandini, M., Heeger, D. J., and Senn, W. (2002). A synaptic explanation of suppression in visual cortex. *J. Neurosci.*, 22(22):10053–65.

⁴Rao and Ballard (1999) have shown that a different implementation of predictive coding is capable of learning Gabor-like RFs from natural images as well as simulating a very limited range of V1 response properties (namely end-stopping). However, two different versions of their algorithm were used to obtain these results: a linear one for simulating end-stopping and a non-linear version for learning Gabor RFs.

- Cavanaugh, J. R., Bair, W., and Movshon, J. A. (2002). Nature and interaction of signals from the receptive field center and surround in macaque V1 neurons. *J. Neurophysiol.*, 88(5):2530–46.
- Dagoi, V. and Sur, M. (2000). Dynamic properties of recurrent inhibition in primary visual cortex: contrast and orientation dependence of contextual effects. *J. Neurophysiol.*, 83:1019–30.
- Daugman, J. G. (1980). Two-dimensional spectral analysis of cortical receptive field profiles. *Vision Res.*, 20:847–56.
- Daugman, J. G. (1988). Complete discrete 2-D Gabor transformations by neural networks for image analysis and compression. *IEEE Trans. Acoust.*, 36(7):1169–79.
- De Meyer, K. and Spratling, M. W. (2009). A model of non-linear interactions between cortical top-down and horizontal connections explains the attentional gating of collinear facilitation. *Vision Res.*, 49(5):553–68.
- De Meyer, K. and Spratling, M. W. (2011). Multiplicative gain modulation arises through unsupervised learning in a predictive coding model of cortical function. *Neural Comput.*, 23(6):1536–67.
- DeAngelis, G. C., Ghose, G. M., Ohzawa, I., and Freeman, R. D. (1999). Functional micro-organization of primary visual cortex: Receptive field analysis of nearby neurons. *J. Neurosci.*, 19(10):4046–64.
- Falconbridge, M. S., Stamps, R. L., and Badcock, D. R. (2006). A simple Hebbian/anti-Hebbian network learns the sparse, independent components of natural images. *Neural Comput.*, 18(2):415–29.
- Felleman, D. J. and Van Essen, D. C. (1991). Distributed hierarchical processing in primate cerebral cortex. *Cerebral Cortex*, 1:1–47.
- Field, D. J. (1994). What is the goal of sensory coding? *Neural Comput.*, 6(4):559–601.
- Földiák, P. (1990). Forming sparse representations by local anti-Hebbian learning. *Biol. Cybern.*, 64:165–70.
- Freeman, T. C. B., Durand, S., Kiper, D. C., and Carandini, M. (2002). Suppression without inhibition in visual cortex. *Neuron*, 35(4):759–71.
- Friston, K. J. (2005). A theory of cortical responses. *Philos. Trans. R. Soc. Lond., B, Biol. Sci.*, 360(1456):815–36.
- Friston, K. J. (2009). The free-energy principle: a rough guide to the brain? *Trends Cogn. Sci.*, 13(7):293–301.
- Hamker, F. H. and Wiltchut, J. (2007). Hebbian learning in a model with dynamic rate-coded neurons: an alternative to the generative model approach for learning receptive fields from natural scenes. *Network*, 18:249–66.
- Harpur, G. F. (1997). *Low Entropy Coding with Unsupervised Neural Networks*. PhD thesis, Department of Engineering, University of Cambridge.
- Heeger, D. J. (1991). Nonlinear model of neural responses in cat visual cortex. In Landy, M. S. and Movshon, J. A., editors, *Computational Models of Visual Processing*, pages 119–33. MIT Press, Cambridge, MA.
- Heeger, D. J. (1992). Normalization of cell responses in cat striate cortex. *Vis. Neurosci.*, 9:181–97.
- Hinton, G. E. (2002). Training products of experts by minimizing contrastive divergence. *Neural Comput.*, 14(8):1711–1800.
- Hinton, G. E., Dayan, P., Frey, B. J., and Neal, R. M. (1995). The wake-sleep algorithm for unsupervised neural networks. *Science*, 268(5214):1158–61.
- Hinton, G. E., Osindero, S., and Teh, Y.-W. (2006). A fast learning algorithm for deep belief nets. *Neural Comput.*, 18:1527–54.
- Houck, C. R., Joines, J. A., and Kay, M. G. (1995). A genetic algorithm for function optimization: A matlab implementation. Technical Report NCSU-IE 9509, North Carolina State University, Department of Industrial Engineering, Raleigh, NC.
- Hoyer, P. O. (2002). Non-negative sparse coding. In *Neural Networks for Signal Processing XII: Proceedings of the IEEE Workshop on Neural Networks for Signal Processing*, pages 557–65.
- Hoyer, P. O. (2003). Modeling receptive fields with non-negative sparse coding. *Neurocomputing*, 52–54:547–52.
- Hoyer, P. O. (2004). Non-negative matrix factorization with sparseness constraints. *J. Mach. Learn. Res.*, 5:1457–69.
- Hoyer, P. O. and Hyvärinen, A. (2000). Independent component analysis applied to feature extraction from colour and stereo images. *Network: Computation in Neural Systems*, 11(3):191–210.
- Hubel, D. H. and Wiesel, T. N. (1962). Receptive fields, binocular interaction and functional architecture in the cat’s visual cortex. *J. Physiol. (Lond.)*, 160:106–54.
- Hubel, D. H. and Wiesel, T. N. (1968). Receptive fields and functional architecture of monkey striate cortex. *J. Physiol. (Lond.)*, 195:215–43.
- Ito, M. and Komatsu, H. (2004). Representation of angles embedded within contour stimuli in area V2 of macaque monkeys. *J. Neurosci.*, 24(13):3313–24.
- Jarrett, K., Kavukcuoglu, K., Ranzato, M., and LeCun, Y. (2009). What is the best multi-stage architecture for object recognition? In *Proc. International Conference on Computer Vision (ICCV’09)*. IEEE.
- Jehee, J. F. M. and Ballard, D. H. (2009). Predictive feedback can account for biphasic responses in the lateral geniculate nucleus. *PLoS Comput. Biol.*, 5(5):e1000373.

- Jones, H. E., Grieve, K. L., Wang, W., and Sillito, A. M. (2001). Surround suppression in primate V1. *J. Neurophysiol.*, 86:2011–28.
- Jones, J. P. and Palmer, L. A. (1987a). An evaluation of the two-dimensional Gabor filter model of simple receptive fields in cat striate cortex. *J. Neurophysiol.*, 58(6):1233–58.
- Jones, J. P. and Palmer, L. A. (1987b). The two-dimensional spatial structure of simple receptive fields in cat striate cortex. *J. Neurophysiol.*, 58(6):1187–211.
- Kersten, D., Mamassian, P., and Yuille, A. (2004). Object perception as Bayesian inference. *Annu. Rev. Psychol.*, 55(1):271–304.
- LeCun, Y., Kavukcuoglu, K., and Farabet, C. (2010). Convolutional networks and applications in vision. In *Proceedings of the International Symposium on Circuits and Systems (ISCAS10)*. IEEE.
- Lee, D. D. and Seung, H. S. (1999). Learning the parts of objects by non-negative matrix factorization. *Nature*, 401:788–91.
- Lee, H., Ekanadham, C., and Ng, A. Y. (2008). Sparse deep belief net model for visual area V2. In Platt, J. C., Koller, D., Singer, Y., and Roweis, S., editors, *Adv. Neural Info. Proc. Sys.*, volume 20, pages 873–80, Cambridge, MA. MIT Press.
- Lee, T. S. (1996). Image representation using 2D Gabor wavelets. *IEEE Trans. Pattern Anal. Mach. Intell.*, 18(10):959–71.
- Lee, T. S. and Mumford, D. (2003). Hierarchical Bayesian inference in the visual cortex. *J. Opt. Soc. Am. A Opt. Image Sci. Vis.*, 20:1434–48.
- Lehky, S. R., Sejnowski, T. J., and Desimone, R. (2005). Selectivity and sparseness in the responses of striate complex cells. *Vision Res.*, 45(1):57–73.
- Lücke, J. (2009). Receptive field self-organization in a model of the fine structure in v1 cortical columns. *Neural Comput.*, 21(10):2805–45.
- Lücke, J. and Sahani, M. (2007). Generalized softmax networks for non-linear component extraction. In *Proc. Int. Conf. on Artif. Neural Netw.*, volume 4668 of *Lecture Notes in Computer Science*, pages 657–67. Springer.
- Lücke, J. and Sahani, M. (2008). Maximal causes for non-linear component extraction. *J. Mach. Learn. Res.*, 9:1227–67.
- Lücke, J., Turner, R., Sahani, M., and Henniges, M. (2009). Occlusive components analysis. In *Adv. Neural Info. Proc. Sys.*, pages 1069–77.
- Lücke, J. and von der Malsburg, C. (2004). Rapid processing and unsupervised learning in a model of the cortical macrocolumn. *Neural Comput.*, 16(3):501–33.
- Malmir, M. and Ghidary, S. S. (2010). A model of primate visual cortex based on category-specific redundancies in natural images. *Connect. Sci.*, 22(4):313–29.
- Marcelja, S. (1980). Mathematical description of the responses of simple cortical cells. *J. Opt. Soc. Am. A Opt. Image Sci. Vis.*, 70:1297–1300.
- Meila, M. and Jordan, M. I. (2000). Learning with mixtures of trees. *J. Mach. Learn. Res.*, 1:1–48.
- Mitchell, S. J. and Silver, R. A. (2003). Shunting inhibition modulates neuronal gain during synaptic excitation. *Neuron*, 38(3):433–45.
- Mumford, D. (1992). On the computational architecture of the neocortex II: the role of cortico-cortical loops. *Biol. Cybern.*, 66:241–51.
- Nuding, U. and Zetsche, C. (2007). Learning the selectivity of V2 and V4 neurons using non-linear multi-layer wavelet networks. *Biosystems*, 89(1-3):273–9.
- Olshausen, B. A. and Field, D. J. (1996a). Emergence of simple-cell receptive properties by learning sparse code for natural images. *Nature*, 381:607–9.
- Olshausen, B. A. and Field, D. J. (1996b). Natural image statistics and efficient coding. *Network*, 7(2):333–9.
- Olshausen, B. A. and Field, D. J. (1997). Sparse coding with an overcomplete basis set: A strategy employed by V1? *Vision Res.*, 37(23):3311–25.
- Olshausen, B. A. and Field, D. J. (2004). Sparse coding of sensory inputs. *Curr. Opin. Neurobiol.*, 14:481–7.
- Perrinet, L. U. (2010). Role of homeostasis in learning sparse representations. *Neural Computation*, 22(7):1812–36.
- Priebe, N. J. and Ferster, D. (2006). The mechanism underlying cross-orientation suppression in cat visual cortex. *Nat. Neurosci.*, 9(4):552–61.
- Ranzato, M., Poultney, C., Chopra, S., and LeCun, Y. (2007). Efficient learning of sparse representations with an energy-based model. In Schölkopf, B., Platt, J., and Hoffman, T., editors, *Adv. Neural Info. Proc. Sys.*, volume 19, pages 1137–44, Cambridge, MA. MIT Press.
- Rao, R. P. N. and Ballard, D. H. (1999). Predictive coding in the visual cortex: a functional interpretation of some extra-classical receptive-field effects. *Nat. Neurosci.*, 2(1):79–87.
- Rehn, M. and Sommer, F. T. (2007). A network that uses few active neurons to code visual input predicts the

- diverse shapes of cortical receptive fields. *J. Comput. Neurosci.*, 22:135–46.
- Reid, R. C. and Alonso, J. M. (1995). Specificity of monosynaptic connections from thalamus to visual cortex. *Nature*, 378:281–4.
- Ringach, D. and Shapley, R. (2004). Reverse correlation in neurophysiology. *Cogn. Sci.*, 28(2):147–66.
- Ringach, D. L. (2002). Spatial structure and symmetry of simple-cell receptive fields in macaque primary visual cortex. *J. Neurophysiol.*, 88(1):455–63.
- Ringach, D. L., Sapiro, G., and Shapley, R. (1997). A subspace reverse correlation method for the study of visual neurons. *Vision Res.*, 37:2455–64.
- Rolls, E. T. and Tovee, M. J. (1995). Sparseness of the neuronal representation of stimuli in the primate temporal cortex. *J. Neurophysiol.*, 73:713–26.
- Rothman, J., Cathala, L., Steuber, V., and Silver, R. A. (2009). Synaptic depression enables neuronal gain control. *Nature*, 457:1015–8.
- Savin, C., Joshi, P., and Triesch, J. (2010). Independent component analysis in spiking neurons. *PLoS Comput. Biol.*, 6(4):e1000757.
- Sceniak, M. P., Ringach, D. L., Hawken, M. J., and Shapley, R. (1999). Contrast’s effect on spatial summation by macaque V1 neurons. *Nat. Neurosci.*, 2(8).
- Schwabe, L., Obermayer, K., Angelucci, A., and Bressloff, P. C. (2006). The role of feedback in shaping the extra-classical receptive field of cortical neurons: A recurrent network model. *J. Neurosci.*, 26(36):9117–29.
- Schwartz, O. and Simoncelli, E. P. (2001). Natural signal statistics and sensory gain control. *Nat. Neurosci.*, 4:819–25.
- Simoncelli, E. P. and Olshausen, B. A. (2001). Natural image statistics and neural representation. *Annu. Rev. Neurosci.*, 24:1193–216.
- Skottun, B. C., Bradley, A., Sclar, G., Ohzawa, I., and Freeman, R. D. (1987). The effects of contrast on visual orientation and spatial frequency discrimination: a comparison of single cells and behavior. *J. Neurophysiol.*, 57:773–86.
- Somers, D. C., Nelson, S. B., and Sur, M. (1995). An emergent model of orientation selectivity in cat visual cortical simple cells. *J. Neurosci.*, 15:5448–65.
- Spratling, M. W. (2002). Cortical region interactions and the functional role of apical dendrites. *Behav. Cogn. Neurosci. Rev.*, 1(3):219–28.
- Spratling, M. W. (2006). Learning image components for object recognition. *J. Mach. Learn. Res.*, 7:793–815.
- Spratling, M. W. (2008a). Predictive coding as a model of biased competition in visual selective attention. *Vision Res.*, 48(12):1391–408.
- Spratling, M. W. (2008b). Reconciling predictive coding and biased competition models of cortical function. *Front. Comput. Neurosci.*, 2(4):1–8.
- Spratling, M. W. (2010). Predictive coding as a model of response properties in cortical area V1. *J. Neurosci.*, 30(9):3531–43.
- Spratling, M. W. (2011). A single functional model accounts for the distinct properties of suppression in cortical area V1. *Vision Res.*, 51(6):563–76.
- Spratling, M. W., De Meyer, K., and Kompass, R. (2009). Unsupervised learning of overlapping image components using divisive input modulation. *Comput. Intell. Neurosci.*, 2009(381457):1–19.
- Spratling, M. W. and Johnson, M. H. (2002). Pre-integration lateral inhibition enhances unsupervised learning. *Neural Comput.*, 14(9):2157–79.
- Spratling, M. W. and Johnson, M. H. (2003). Exploring the functional significance of dendritic inhibition in cortical pyramidal cells. *Neurocomputing*, 52-54:389–95.
- Srinivasan, M. V., Laughlin, S. B., and Dubs, A. (1982). Predictive coding: A fresh view of inhibition in the retina. *Proc. R. Soc. Lond., B, Biol. Sci.*, 216(1205):427–59.
- Stetter, M., Bartsch, H., and Obermayer, K. (2000). A mean-field model for orientation tuning, contrast saturation, and contextual effects in the primary visual cortex. *Biol. Cybern.*, 82(4):291–304.
- Teh, Y. W., Welling, M., Osindero, S., and Hinton, G. E. (2003). Energy-based models for sparse overcomplete representations. *J. Mach. Learn. Res.*, 4:1235–60.
- Tolhurst, D. J., Smyth, D., and Thompson, I. D. (2009). The sparseness of neuronal responses in ferret primary visual cortex. *J. Neurosci.*, 29(8):2355–70.
- Van Hateren, J. H. and van der Schaaf, A. (1998). Independent component filters of natural images compared with simple cells in primary visual cortex. *Proc. Biol. Sci.*, 265:359–66.
- Van Hooser, S. D. (2007). Similarity and diversity in visual cortex: Is there a unifying theory of cortical computation? *Neuroscientist*, 13:639–56.
- Vinje, W. E. and Gallant, J. L. (2000). Sparse coding and decorrelation in primary visual cortex during natural vision. *Science*, 287(5456):1273–6.

- Wainwright, M. J., Schwartz, O., and Simoncelli, E. P. (2001). Natural image statistics and divisive normalization: modeling nonlinearities and adaptation in cortical neurons. In Rao, R., Olshausen, B., and Lewicki, M., editors, *Statistical Theories of the Brain*, pages 203–22. MIT Press, Cambridge, MA.
- Weber, C. and Triesch, J. (2008). A sparse generative model of V1 simple cells with intrinsic plasticity. *Neural Comput.*, 20:1261–84.
- Willmore, B. and Tolhurst, D. J. (2001). Characterizing the sparseness of neural codes. *Network*, 12:255–70.
- Wiltchut, J. and Hamker, F. H. (2009). Efficient coding correlates with spatial frequency tuning in a model of V1 receptive field organization. *Vis. Neurosci.*, 26:21–34.
- Yuille, A. and Kersten, D. (2006). Vision as Bayesian inference: analysis by synthesis? *Trends Cogn. Sci.*, 10(7):301–8.