Constrained Predictive Coding as a Biologically Plausible Model of the Cortical Hierarchy

Siavash Golkar[*1] **Tiberiu Tesileanu**[*1] **Yanis Bahroun**[1]

Anirvan M. Sengupta[2,3,4] **Dmitri Chklovskii**[1,5]

[1] Center for Computational Neuroscience, Flatiron Institute
[2] Center for Computational Mathematics, Flatiron Institute
[3] Center for Computational Quantum Physics, Flatiron Institute
[4] Department of Physics and Astronomy, Rutgers University
[5] Neuroscience Institute, NYU Medical Center

{sgolkar,ttesileanu,ybahroun,dchklovskii}@flatironinstitute.org
anirvans.physics@gmail.com

Abstract

Predictive coding (PC) has emerged as an influential normative model of neural computation with numerous extensions and applications. As such, much effort has been put into mapping PC faithfully onto the cortex, but there are issues that remain unresolved or controversial. In particular, current implementations often involve separate value and error neurons and require symmetric forward and backward weights across different brain regions. These features have not been experimentally confirmed. In this work, we show that the PC framework in the linear regime can be modified to map faithfully onto the cortical hierarchy in a manner compatible with empirical observations. By employing a disentangling-inspired constraint on hidden-layer neural activities, we derive an upper bound for the PC objective. Optimization of this upper bound leads to an algorithm that shows the same performance as the original objective and maps onto a biologically plausible network. The units of this network can be interpreted as multi-compartmental neurons with non-Hebbian learning rules, with a remarkable resemblance to recent experimental findings. There exist prior models which also capture these features, but they are phenomenological, while our work is a normative derivation. Notably, the network we derive does not involve one-to-one connectivity or signal multiplexing, which the phenomenological models required, indicating that these features are not necessary for learning in the cortex. The normative nature of our algorithm in the simplified linear case allows us to prove interesting properties of the framework and analytically understand the computational role of our network's components. The parameters of our network have natural interpretations as physiological quantities in a multi-compartmental model of pyramidal neurons, providing a concrete link between PC and experimental measurements carried out in the cortex.

1 Introduction

Over the past decades, predictive coding (PC), a normative framework for learning representations that maximize predictive power, has played an important role in computational neuroscience [1, 2].

[*] Equal contribution

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

Initially proposed as an unsupervised learning paradigm in the retina [3], it has since been expanded to the supervised regime [4] with arbitrary graph topologies [5, 6]. The PC framework has been analyzed in many contexts [7, 8] and has found many applications, from clinical neuroscience [9, 10] to memory storage and retrieval [11]. We refer the reader to [12, 13] for recent reviews.

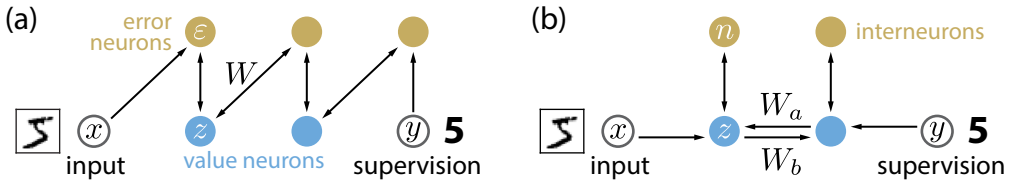

Figure 1: Schematic architecture of the predictive coding network (PCN) and our covariance-constrained network (BioCCPC). (a) PCN from [4], figure adapted from [13]. The intra-layer connectivity is to be one-to-one, while the inter-layer connectivity is symmetric. (b) Our BioCCPC network. There is no requirement for symmetric weights across layers, and the connectivity within layers can be diffuse.

Predictive coding is viewed as a possible theory of cortical computation, and many parallels have been drawn with the known neurophysiology of cortex [12]. While the initial works proposed a biologically plausible network [1, 2], the connection with cortex was more closely examined in [14], where the PC module was mapped onto a cortical-column microcircuit. However, there are aspects of this mapping that have proved controversial. Among these are the requirement of multiple redundant cortical operations, the symmetric connectivity pattern, the one-to-one connectivity of value and error neurons, and also the requirement that feedback connections be inhibitory [12, 15, 16], as sketched in Fig. 1a . The presence of separate error and value neurons has itself been called into question [15].

The PC-based neural circuits also do not account for more recent experimental findings highlighting the details of computation in the cortex [17–26]. For example, the learning dynamics of multi-compartmental pyramidal neurons has been closely investigated. In particular, it was observed that the plasticity of the synapses of the basal compartment is driven by the activity in the apical tuft of the neuron by so-called calcium plateau potentials [19, 20, 24], leading to non-Hebbian learning rules [25]. These experiments have motivated the development of several models of microcircuits with multi-compartmental neurons [27–33]. In a number of cases, it has been shown that these models can replicate learning similar to the backpropagation algorithm under specific assumptions [30, 34, 35]. However, because of their rather phenomenological nature, detailed analysis of these models is in many cases challenging and one must resort to numerical simulations, rendering the task of understanding the role of various neurophysiological quantities difficult. To this date, a normative framework (PC or otherwise) that can explain these experimental findings is still lacking.

In this work, we show how to modify the PC framework to make it compatible with the aforementioned experimental observations while maintaining much of the analytical control afforded to use by a normative approach. Inspired by prior work which explored the effects of finding decorrelated representations [36–41], we add a decorrelating inequality constraint to the covariance of the latent representations of PC. Using this constraint, we derive an upper bound for the PC objective. By working in the linear regime, we can prove several useful properties of our algorithm. We show that the learning algorithm derived from this upper bound does not suffer from the issues of prior implementations and naturally maps onto the hierarchical structure of the cortical pyramidal neurons.

Contributions

- We introduce BioCCPC, a novel biologically plausible algorithm derived as a modification of predictive coding (PC), and show that it closely matches the known physiology of the cortical hierarchy.
- We find an analytic upper bound to the PC objective by imposing a decorrelation-inspired inequality constraint on the latent space.
- We interpret the different parameters of the algorithm in terms of the conductances and leaks of the separate compartments of the pyramidal neuron. We find that the neural compartmental conductances encode the variances of the PC framework, and the somatic leak maps onto a thresholding mechanism of the associated eigenvalue problem.

2 Related work and review of predictive coding

The backpropagation algorithm [34, 42] is the predominant tool in the training of deep neural networks but is generally not considered biologically plausible [43]. Over the years, many authors have explored biologically plausible approximations or alternatives to backpropagation [30,31,44–58] (for a more complete review see [59]). These approaches generally fall into two categories. First are normative approaches, such as Predictive Coding [1–3], Target Propagation and variations [44,45,47], Equilibrium Propagation [49, 53] and others [51, 54–56], where one starts from a mathematically motivated optimization objective. These methods, by virtue of their normative derivation, have a firm grounding in theory; however, they do not fully conform to the experimental observations in the brain (see below and Sec. 4). The second approach is driven by biology, with network structures and learning rules inspired by experimental findings [28, 30, 31, 48, 60]. While these works mostly conform to the experimentally observed findings, they are more challenging to analyze because of their conjectured phenomenological nature.

The goal of the present paper is *not* to propose yet another biologically plausible alternative to backpropagation. It is rather to demonstrate that the normative framework of predictive coding, when combined with a constraint, can indeed closely match experimental observations. For this reason, in this work, we focus on comparing our method with previous implementations of predictive coding and do not concern ourselves with other biologically plausible alternatives to backpropagation. The relationship between the PC framework and backpropagation was explored in [4, 61–64]. The advantages of PC over backpropagation were highlighted in [13].

We now turn to a brief review of the predictive coding framework from [4] before delving into the details of our method.

Notation. Bold upper case $\mathbf{M}$ and lower case variables $\mathbf{v}$ denote matrices and vectors, respectively. By upper case letter $\mathbf{X}, \mathbf{Y}, \mathbf{Z}$, we denote data matrices in $\mathbb{R}^{d\times T}$, where d and T are the dimensions of the relevant variable and the number of samples. Lower case $\mathbf{x}, \mathbf{y}, \mathbf{z}$, denote the relevant quantities of a single sample, and $\mathbf{x}_t, \mathbf{y}_t, \mathbf{z}_t$, denote the tth sample. $\|\mathbf{M}\|_F^2$ denotes the Frobenius norm of $\mathbf{M}$.

2.1 Review of predictive coding

Probabilistic model. In this section, we review the supervised predictive coding algorithm from [4]. The derivation starts from a probabilistic model for supervised learning, which parallels the architecture of an artificial neural network (ANN) with $n+1$ layers. In this model, the neurons of each layer are random variables $\mathbf{z}^{(l)}$ (denoting the vector of activations in the lth layer) with layers 0 and n, respectively, denoting the input and output layers of the network. We assume that the joint probability of the latent variables factorizes in a Markovian structure

$$p(\mathbf{z}^{(0)}, \mathbf{z}^{(1)}, \cdots, \mathbf{z}^{(n)}) = p(\mathbf{z}^{(n)}|\mathbf{z}^{(n-1)}) \times p(\mathbf{z}^{(n-1)}|\mathbf{z}^{(n-2)}) \times \cdots \times p(\mathbf{z}^{(0)})$$

with the relationship between the random variables of adjacent layers given by:

$$p\big(\mathbf{z}^{(l)}|\mathbf{z}^{(l-1)}\big) = \mathcal{N}\big(\mathbf{z}^{(l)}; \boldsymbol{\mu}^{(l)}, \boldsymbol{\Sigma}^{(l)}\big)\,, \qquad \text{with} \quad \boldsymbol{\mu}^{(l)} = \mathbf{W}^{(l-1)} f(\mathbf{z}^{(l-1)}), \quad \boldsymbol{\Sigma}^{(l)} = \sigma^{(l)2}\mathbf{I}\,. \tag{1}$$

The mean of the probability density on layer l mirrors the activity of the analogous ANN given by $\boldsymbol{\mu}^{(l)} = \mathbf{W}^{(l-1)} f(\mathbf{z}^{(l-1)})$, where $\mathbf{W}^{(l-1)}$ are the weights connecting layers $l-1$ and l. The objective function is then given by the negative log-likelihood of the joint distribution function:

$$L = -\sum_t \log p(\mathbf{z}_t^{(0)}, \ldots, \mathbf{z}_t^{(n)}) = \frac{1}{2}\sum_l \frac{\left\|\mathbf{Z}^{(l)} - \mathbf{W}^{(l-1)} f(\mathbf{Z}^{(l-1)})\right\|_F^2}{\sigma^{(l)2}} + \text{const}\,, \tag{2}$$

where we have switched to the data matrix notation for brevity and assumed the variances $\sigma^{(l)2}$ are fixed hyperparameters. In the following, we refer to L, Eq. (2), without the constant term.

Learning. Learning in the Whittington and Bogacz framework takes place in two steps [4]. First, the values of the random variables are determined by finding the most probable configuration of the joint distribution function when both the input and output layers are conditioned on the given input and output ($\mathbf{z}^{(0)} = \mathbf{x}$ and $\mathbf{z}^{(n)} = \mathbf{y}$):

$$\mathbf{z}^{*(1)}, \ldots, \mathbf{z}^{*(n-1)} = \underset{\mathbf{z}^{(1)},\ldots,\mathbf{z}^{(n-1)}}{\arg\min}\; L(\mathbf{z}^{(0)} = \mathbf{x}, \mathbf{z}^{(n)} = \mathbf{y}). \tag{3}$$

The solution to this minimization problem can be found via gradient descent, which we evaluate component-wise for clarity as:

$$\dot{z}_j^{(l)} = -\eta \partial_{z_j} L = \eta(-\varepsilon_j^{(l)} + \sum_i \varepsilon_i^{(l+1)} W_{ij}^{(l)} f'(z_j^{(l)})), \quad \text{with} \quad \varepsilon_i^{(l)} = \frac{z_i^{(l)} - \mu_i^{(l)}}{\sigma^{(l)2}}, \tag{4}$$

where η is the gradient descent step size. The second step is then to minimize the objective with respect to the weights while keeping the previously obtained neuron values fixed. This corresponds to optimizing the value of the loss at the MAP estimate and can also be carried out by gradient descent, leading to:

$$\delta W_{ij}^{(l)} \propto -\frac{\partial L}{\partial W_{ij}^{(l)}} = \varepsilon_i^{*(l+1)} f\big(z_j^{*(l)}\big), \tag{5}$$

where the star superscript denotes equilibrium values of the latent variable dynamics obtained in the first step (Eq. (3)). This algorithm can be implemented by a biologically plausible network as described in [4]; see Figure 1a. However, as discussed in Section 1, its mapping onto the cortex has proved controversial. For further details regarding these steps see [4].

3 A constrained predictive coding framework

In this section, we introduce and discuss our novel covariance-constrained predictive coding (CCPC) model that builds upon the supervised PC paradigm of [4]. Our model also straightforwardly extends to the unsupervised PC framework of [1, 2]. For simplicity, we work in the linear regime ($f(x) = x$), which allows us to prove different properties of our framework.

3.1 Derivation of an upper bound objective

Reduction to a sum of objectives. We start by reducing the optimization problem, Eq. (2), into a set of overlapping sub-problems, which will allow us to break the symmetry between feedforward and feedback weights. To do so, we first introduce a copy of the terms containing the weights $\mathbf{W}^{(1)}$ to $\mathbf{W}^{(n-2)}$, denoted by $\mathbf{W}_a^{(l)}$ and $\mathbf{W}_b^{(l)}$ respectively, as

$$\begin{aligned}
\min_{\mathbf{Z},\mathbf{W}} L &= \min_{\mathbf{Z},\mathbf{W}} \frac{1}{2} \sum_{l=1}^{n} \frac{\left\|\mathbf{Z}^{(l)} - \mathbf{W}^{(l-1)}\mathbf{Z}^{(l-1)}\right\|_F^2}{\sigma^{(l)2}} = \min_{\mathbf{Z},\mathbf{W}_a,\mathbf{W}_b} \hat{L} \\
\hat{L} &= \min_{\mathbf{Z},\mathbf{W}_a,\mathbf{W}_b} \frac{1}{2} \frac{\left\|\mathbf{Z}^{(1)} - \mathbf{W}_b^{(0)}\mathbf{Z}^{(0)}\right\|_F^2}{\sigma^{(1)2}} + \frac{1}{2} \frac{\left\|\mathbf{Z}^{(n)} - \mathbf{W}_a^{(n-1)}\mathbf{Z}^{(n-1)}\right\|_F^2}{\sigma^{(n)2}} \\
&\quad + \frac{1}{4} \sum_{l=2}^{n-1} \left[\frac{\left\|\mathbf{Z}^{(l)} - \mathbf{W}_b^{(l-1)}\mathbf{Z}^{(l-1)})\right\|_F^2}{\sigma^{(l)2}} + \frac{\left\|\mathbf{Z}^{(l)} - \mathbf{W}_a^{(l-1)}\mathbf{Z}^{(l-1)})\right\|_F^2}{\sigma^{(l)2}} \right].
\end{aligned} \tag{6}$$

For consistency, we rename $\mathbf{W}^{(0)}$, $\mathbf{W}^{(n-1)}$ to $\mathbf{W}_b^{(0)}$, $\mathbf{W}_a^{(n-1)}$, respectively. Introducing these copies does not change the optimization[2] but will help us avoid weight sharing in the steps below. We now pair the terms two by two as

$$\min_{\mathbf{Z},\mathbf{W}_a,\mathbf{W}_b} \hat{L} = \frac{1}{2} \sum_{l=1}^{n-1} \left[g_b^{(l)} \left\|\mathbf{Z}^{(l)} - \mathbf{W}_b^{(l-1)}\mathbf{Z}^{(l-1)}\right\|_F^2 + g_a^{(l)} \left\|\mathbf{Z}^{(l+1)} - \mathbf{W}_a^{(l)}\mathbf{Z}^{(l)}\right\|_F^2 \right], \tag{7}$$

with $g_a^{(n-1)} = 1/\sigma^{(n)2}$, $g_b^{(1)} = 1/\sigma^{(1)2}$, and $g_a^{(l-1)} = g_b^{(l)} = 1/(2\sigma^{(l)2})$ for $l = 2, \ldots, n-1$.

Weight sharing occurs here from terms like $\mathbf{z}^{(l+1)\top}\mathbf{W}\mathbf{z}^{(l)}$, obtained from expanding the squared norms in Eq. (7). Indeed, the gradient descent dynamics with respect to $\mathbf{z}^{(l+1)}$ (resp. $\mathbf{z}^{(l)}$) leads to terms of the form $\mathbf{W}\mathbf{z}^{(l)}$ in $\dot{\mathbf{z}}^{(l+1)}$ (resp. $\mathbf{W}^\top\mathbf{z}^{(l+1)}$ in $\dot{\mathbf{z}}^{(l)}$), which use the same weights $\mathbf{W}$. Thanks

[2] This can be directly verified by finding the optima for $\mathbf{W}$'s before and after the change. We have $\mathbf{W}^{(l)} = \mathbf{W}_a^{(l)} = \mathbf{W}_b^{(l)} = \mathbf{Z}^{(l+1)}\mathbf{Z}^{(l)\top}(\mathbf{Z}^{(l+1)}\mathbf{Z}^{(l)\top})^{-1}$. Plugging these back into Eq. (6) we see that the equality holds. However, in the next step, since we treat $\mathbf{W}_a$'s and $\mathbf{W}_b$'s differently, $\mathbf{W}_a = \mathbf{W}_B$ will no longer hold.

to the doubling of the weights, we can avoid this problem by optimizing each term in the sum in Eq. (7) separately. In other words,

$$\min_{\mathbf{Z},\mathbf{W}_a,\mathbf{W}_b} \hat{L} \leq \sum_{l=1}^{n-1} \min_{\mathbf{Z}^{(l)},\mathbf{W}_a^{(l)},\mathbf{W}_b^{(l-1)}} \hat{L}^{(l)} \,, \tag{8}$$

$$\text{where } \hat{L}^{(l)} \equiv \frac{1}{2}\sum_{l=0}^{n}\left[g_b^{(l)}\left\|\mathbf{Z}^{(l)} - \mathbf{W}_b^{(l-1)}\mathbf{Z}^{(l-1)}\right\|_F^2 + g_a^{(l)}\left\|\mathbf{Z}^{(l+1)} - \mathbf{W}_a^{(l)}\mathbf{Z}^{(l)}\right\|_F^2\right].$$

This inequality holds simply because we are no longer finding the minimum of the full objective $\hat{L}$. We are instead finding the minimum of each component separately and then evaluating $\sum_l \hat{L}^{(l)} = \hat{L}$. This splits the $(n+1)$-layer optimization problem into a set of 3-layer optimizations, in each of which only the middle layer is being optimized. Note, however, that these are overlapping, so the different optimization problems need to be solved self-consistently. We make this precise in the supplementary materials section and show that it provides an upper bound for our objective L (SM Sec. A). Separating the objective function in this manner eliminates the weight sharing problem for $\mathbf{W}_b$, but the problem remains for $\mathbf{W}_a$. This can be seen by noting that the gradient descent equations only depend on $\mathbf{W}_b$ but depend on both $\mathbf{W}_a$ and $\mathbf{W}_a^\top$. We address this problem next.

Whitening constraint. The idea of decorrelating internal representation has been widely used for unsupervised tasks, often motivated by neuroscience [36–38]. In the case of deep learning, the main motivations were improved convergence speed and generalization [39–41]. Decorrelation has also been used to circumvent the weight transport problem [55]. Inspired by these observations, we introduce the inequality constraint $\frac{1}{T}\mathbf{Z}^{(l)}\mathbf{Z}^{(l)\top} \preceq \mathbf{I}$, imposing an upper bound on the eigenvalues of the covariance matrix. In Sec. 5 and SM Sec. B we show that under some assumptions this inequality constraint leads to whitening the latent variables. Adding this inequality constraint to the objective we have:

$$\begin{aligned}
\min_{\mathbf{Z}^{(l)},\mathbf{W}_a^{(l)},\mathbf{W}_b^{(l)}} \hat{L}^{(l)} &\leq \min_{\substack{\mathbf{Z}^{(l)},\mathbf{W}_a^{(l)},\mathbf{W}_b^{(l)}\\ \mathbf{Z}^{(l)}\mathbf{Z}^{(l)\top}\preceq T\times\mathbf{I}}} \hat{L}^{(l)} \\
&\leq \min_{\substack{\mathbf{Z}^{(l)},\mathbf{W}_a^{(l)},\mathbf{W}_b^{(l)}\\ \mathbf{Z}^{(l)}\mathbf{Z}^{(l)\top}\preceq T\times\mathbf{I}}} \frac{1}{2}\Bigg[g_b^{(l)}\left\|\mathbf{Z}^{(l)} - \mathbf{W}_b^{(l-1)}\mathbf{Z}^{(l-1)}\right\|_F^2 + c^{(l)}\left\|\mathbf{Z}^{(l)}\right\|_F^2 \\
&\qquad + g_a^{(l)}\operatorname{Tr}\Big(-2\mathbf{Z}^{(l+1)\top}\mathbf{W}_a^{(l)}\mathbf{Z}^{(l)}\Big) + g_a^{(l)}T\,\mathbf{W}_a^{(l)\top}\mathbf{W}_a^{(l)}
\end{aligned}$$

The first inequality results from the fact that the parameters of the new optimization problem are a subset of the parameters of the original optimization problem. In the second inequality we have expanded the second term of $\hat{L}^{(l)}$ and used the constraint $\frac{1}{T}\mathbf{Z}^{(l)}\mathbf{Z}^{(l)\top} \preceq \mathbf{I}$ to replace $\mathbf{Z}^{(l)}\mathbf{Z}^{(l)\top}$ with T, making the term larger, resulting again in an upper bound. This replacement eliminates the weight sharing problem in $\mathbf{W}_a$, as the gradient descent equations now only depend on $\mathbf{W}_a^\top$ and not on $\mathbf{W}_a$ (cf. Eqs. (10) and (11)).

The inequality can be implemented by using a positive-definite Lagrange multiplier $\mathbf{Q}^{(l)\top}\mathbf{Q}^{(l)}$:

$$\begin{aligned}
\min_{\mathbf{Z}^{(l)},\mathbf{W}_a^{(l)},\mathbf{W}_b^{(l-1)}} \max_{\mathbf{Q}^{(l)}} \frac{1}{2}\Bigg[&g_b^{(l)}\left\|\mathbf{Z}^{(l)} - \mathbf{W}_b^{(l-1)}\mathbf{Z}^{(l-1)}\right\|_F^2 + c^{(l)}\left\|\mathbf{Z}^{(l)}\right\|_F^2 \\
&+ g_a^{(l)}\operatorname{Tr}\Big(-2\mathbf{Z}^{(l+1)\top}\mathbf{W}_a^{(l)}\mathbf{Z}^{(l)}\Big) + g_a^{(l)}T\,\mathbf{W}_a^{(l)\top}\mathbf{W}_a^{(l)} \\
&+ \operatorname{Tr}\mathbf{Q}^{(l)\top}\mathbf{Q}^{(l)}\big(\mathbf{Z}^{(l)}\mathbf{Z}^{(l)\top} - T\mathbf{I}\big) + \text{const}\Bigg],
\end{aligned} \tag{9}$$

where we have added an additional quadratic term in $\mathbf{Z}$ as a regularizer. This can also be interpreted as an extra term in the prior of the latent space. Equation (9) is the final form of the objective, which we then optimize using gradient descent/ascent.

3.2 Neural dynamics and learning rules.

Similar to the PCN of [4], the dynamics of our network during learning proceeds in two steps. First the neural dynamics is derived by taking gradient steps of $\hat{L}^{(l)}$ from Eq. (9) with respect to $\mathbf{z}^{(l)}$:

$$\dot{\mathbf{z}}^{(l)} = g_b^{(l)}\mathbf{W}_b^{(l-1)}\mathbf{z}^{(l-1)} + g_a^{(l)}\mathbf{W}_a^{(l)\top}\mathbf{z}^{(l+1)} - \big(g_b^{(l)} + c^{(l)}\big)\mathbf{z}^{(l)} - g_a^{(l)}\mathbf{Q}^{(l)\top}\mathbf{n}^{(l)}, \tag{10}$$

where we have defined the variables $\mathbf{n}^{(l)} = (1/g_a^{(l)})\mathbf{Q}^{(l)}\mathbf{z}^{(l)}$ for each layer of the network.

The weight updates are derived via stochastic gradient descent of the loss given in Eq. (9) after the neural dynamics have reached equilibrium. These are given by

$$\delta\mathbf{W}_b^{(l)} \propto \Big[g_a^{(l+1)}\big(\mathbf{W}_a^{(l+1)\top}\mathbf{z}^{(l+2)} - \mathbf{Q}^{(l+1)\top}\mathbf{n}^{(l+1)}\big) - c^{(l+1)}\mathbf{z}^{(l+1)}\Big]\mathbf{z}^{(l)\top}, \tag{11a}$$

$$\delta\mathbf{W}_a^{(l)\top} \propto \mathbf{z}^{(l)}\mathbf{z}^{(l+1)\top} - \mathbf{W}_a^{(l)\top}, \tag{11b}$$

$$\delta\mathbf{Q}^{(l)} \propto \mathbf{n}^{(l)}\mathbf{z}^{(l)\top} - \mathbf{Q}^{(l)}. \tag{11c}$$

We used the neural dynamics equilibrium equation for $\mathbf{z}^{(l)}$ to simplify the weight update for $\mathbf{W}_b^{(l)}$. This yields our online algorithm (Alg. 1 in SM Sec. E), with the architecture shown in Fig. 1b. The algorithm can be implemented in a biologically plausible neural network as in Fig. 2; see Sec. 4.

4 Biological implementation and comparison with experimental observations

In this section, we introduce a biologically plausible neural circuit that implements the CCPC algorithm, denoted by BioCCPC. We also demonstrate that the details of this circuit resemble the neurophysiological properties of pyramidal cells in the neocortex and the hippocampus.

Neural architecture. The algorithm for BioCCPC (Alg. 1) summarized by the neural dynamics from Eq. (10) and weight update rules from Eqs. (11) can be implemented by a neural circuit with schematic shown in Fig. 2.

The activity of the $n-1$ hidden layers, $\{\mathbf{z}^{(l)}\}_{l=1}^{n-1}$, is encoded as the outputs of $n-1$ sets of neurons, representing pyramidal neurons of different cortical regions. The matrices $\mathbf{W}_a^{(l)}$ and $\mathbf{W}_b^{(l)}$ are encoded as the weights of synapses between the pyramidal neurons of adjacent layers. Explicitly, the matrix $\mathbf{W}_a^{(l)}$ (resp. $\mathbf{W}_b^{(l-1)}$) is the efficacy of the synapse connecting $\mathbf{z}^{(l+1)}$ (resp. $\mathbf{z}^{(l-1)}$) to the pyramidal neurons $\mathbf{z}^{(l)}$. Because of the disjoint nature of these two inputs, we model these as synapsing respectively onto the distal (apical tuft) and proximal (mostly basal) dendrites of the pyramidal neurons, respectively; see Fig. 2. This is reminiscent of cortical pyramidal neurons, which also have two integration sites, the proximal compartment comprised of the basal and proximal apical dendrites providing inputs to the soma, and the distal compartment comprised of the apical dendritic tuft [22, 65]. These two compartments receive excitatory inputs from two separate sources [18, 66].

Similarly, the auxiliary variables $\mathbf{n}^{(l)}$ are represented by the activity of interneurons in each cortical region.[3] The $\mathbf{Q}^{(l)}$ synaptic weights are encoded in the weights of synapses connecting $\mathbf{n}^{(l)}$ to $\mathbf{z}^{(l)}$, while $\mathbf{Q}^{(l)\top}$ models the weights of synapses from $\mathbf{z}^{(l)}$ to $\mathbf{n}^{(l)}$. In a biological setting, the implied equality of weights of synapses from $\mathbf{z}^{(l)}$ to $\mathbf{n}^{(l)}$ and the transpose of those from $\mathbf{n}^{(l)}$ to $\mathbf{z}^{(l)}$ can be guaranteed by applying the same Hebbian learning rule [55]. However, unlike previous work, we do not require interneurons of one layer to be connected to the pyramidal neurons of another layer.

Neural dynamics and canonical components. The neural dynamics of the pyramidal neurons of our circuit given by Eq. (10) can be recast as the dynamics of a three compartment neuron:

$$\tau\dot{\mathbf{z}}^{(l)} = -g_{\text{lk}}^{(l)}\mathbf{z}^{(l)} + g_a^{(l)}(\mathbf{v}_a^{(l)} - \mathbf{z}^{(l)}) + g_b^{(l)}(\mathbf{v}_b^{(l)} - \mathbf{z}^{(l)}), \tag{12}$$

where we have defined the apical and basal compartmental membrane potentials

$$\mathbf{v}_a^{(l)} = \mathbf{W}_a^{(l)\top}\mathbf{z}^{(l+1)} - \mathbf{Q}^{(l)\top}\mathbf{n}^{(l)}, \qquad \mathbf{v}_b^{(l)} = \mathbf{W}_b^{(l-1)}\mathbf{z}^{(l-1)}. \tag{13}$$

[3] There are multiple types of interneurons targeting pyramidal cells [67, 68]. The interneurons of BioCCPC most closely resemble the somatostatin-expressing interneurons, which preferentially inhibit the apical dendrites.

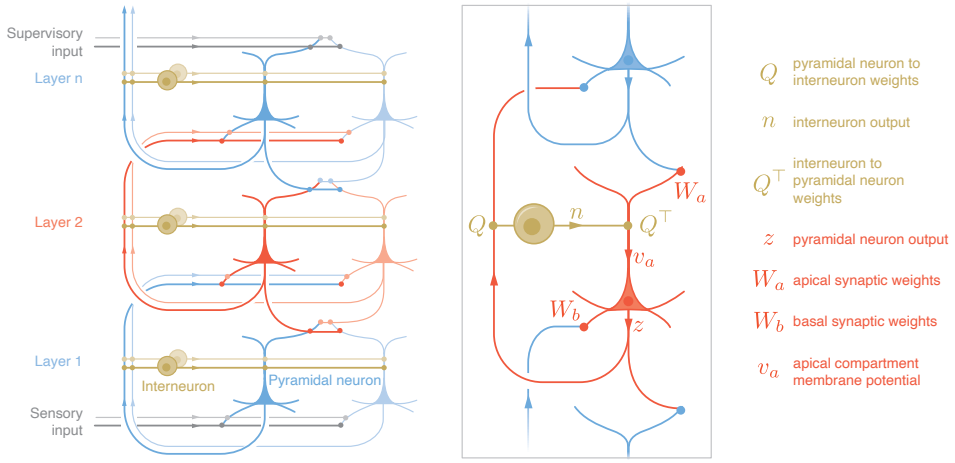

Figure 2: Schematic of the biological implementation of the BioCCPC algorithm. Left: overall connectivity pattern, showing all-to-all connections between pyramidal neurons in consecutive layers, and between pyramidal and interneurons in the same layer. Right: zoom in on one neuron and its immediate neighbors, showing the membrane potentials and synaptic weights that are relevant for our algorithm.

Here we defined the leak conductance $g_{\text{lk}}^{(l)} = c^{(l)} - g_a^{(l)}$. Interestingly, this is the same neural dynamics of a three-compartment neuron posited previously in [28, 30], which we derived in a normative way. Here, g_a (resp. g_b) are conductances between the apical (resp. basal) compartments and the soma, and g_{lk} is the somatic leak. Our derivation clarifies the relationship between the physiological quantities and the objective function. Indeed, the basal and apical conductances play the role of the inverse variances in the predictive coding objective. These inverse variances, generally referred to as 'precisions' in the PC literature, have previously been argued to be encoded in lateral inhibitions [2] or top-down attention [69, 70]. In our model, they are simply encoded in the compartmental conductances. In Sec. 5 we show that the leak term also has a functional role and determines a threshold for dynamically learning more diverse variables [71, 72].

Synaptic weight updates. Matching observations in the cortex, in our circuit [18, 22, 65, 73, 74], the basal and apical weights ($\mathbf{W}_b$ and $\mathbf{W}_a$) of our algorithm are updated differently. The basal synaptic weights of the pyramidal neurons, given by the elements of $\mathbf{W}_b^{(l)}$, are updated by the product of two factors represented in the corresponding post- and pre-synaptic neurons (Eq. 11a):

$$\delta \mathbf{W}_b^{(l-1)} \propto \left[g_a^{(l)} \mathbf{v}_a^{(l)} - c^{(l)} \mathbf{z}^{(l)} \right] \mathbf{z}^{(l-1)\top} . \tag{14}$$

The first term in parentheses, $g_a^{(l)} \mathbf{v}_a^{(l)}$, is proportional to the total apical current, given by the difference between the excitatory synaptic current in the apical tuft, $\mathbf{W}_a^{(l)\top} \mathbf{z}^{(l+1)}$, and the inhibitory current induced by interneurons synapsing onto the distal compartment, $\mathbf{Q}^{(l)\top} \mathbf{n}^{(l)}$. Biologically, this factor can be approximated by the calcium plateau potential traveling down the apical shaft. The calcium plateau potential has been experimentally seen to drive plasticity of the basal synapses matching the derived update rule of our circuit [18, 20, 23–25]. The second factor is simply the neural output. Because the total update is not purely dependent on the action potentials of the pre- and post-synaptic neurons, such plasticity is called non-Hebbian [25]. The synaptic learning rule for the apical weights $\mathbf{W}_a^{(l)}$ (Eq. 11b) and the synapses connecting the interneurons and the pyramidal neurons $\mathbf{Q}^{(l)}$ (Eq. 11c) are simply Hebbian, also matching experimental observations in the cortex [19, 75].

Features and limitations. Here we summarize the main differences between the biological implementations of CCPC and PC. In PC, the forward and backward weights (in our notation $\mathbf{W}_b$ and $\mathbf{W}_a^\top$) are symmetric, whereas, in CCPC, they are not constrained to be symmetric and are generically not so (see Sec. 6). Another difference is that in PC, the connection between value neurons and error neurons is one-to-one and fixed; this constraint does not exist between the pyramidal and interneurons of our model, which are no longer one-to-one, and have plastic connectivity.

While we have shown the improvement in biological realism of our model over the traditional PC network, this was done in the linear case. However, recent work on deep linear networks [76] has provided many insights into the learning dynamics of deep networks. Some of the properties discovered in the deep linear network, like "balancedness" of weights [77, 78], generalize for certain nonlinear networks [78]. We believe several of our observations will generalize in a similar fashion.

There are other aspects that can also be improved. For example, the connectivity between pyramidal neurons and interneurons of CCPC is required to be symmetric. This symmetry can be achieved via Hebbian learning [55]; however, it would be interesting to explore if such symmetry is indeed required. In CCPC, symmetric connectivity is only required between neurons of the *same* cortical region, which is a much less stringent biological requirement than that of symmetric weights between *different* cortical regions, as in PC. Another shortcoming of our model is that while the teaching signal for the basal synapses in CCPC is signed and graded, in the cortex these signals are generally believed to be stereotypical [18].

5 Theoretical arguments and interpretation

In this section, we summarize some of the salient theoretical features of our framework and describe their biological significance. Further details and proofs are given in SM Sec. B.

5.1 Implicit error computation

In backpropagation, as well as in PC, the learning algorithm computes a loss (or a local error) which is then used to compute updates to the weights. In CCPC, no error or loss is explicitly computed. So how does CCPC learn? In the following proposition, we show that in the $g_a \to 0$ limit, the algorithm implicitly computes an error in its weight updates. This quasi feed-forward or weak nudging limit has been used to explore the learning dynamics of biologically plausible networks [30] and is related to the 'fixed prediction' assumption [4, 61, 62].

Proposition 1. *Assume that the learning rules (Eqs. 11) are at equilibrium and we receive a new datapoint given by $\mathbf{x}_{T+1}, \mathbf{y}_{T+1}$. For $c^{(l)} = 0$ and in the limit of $\epsilon \equiv g_a/g_b \to 0$, the leading term in ϵ for the forward weight updates $\delta \mathbf{W}_b^{(l-1)}$ for this new sample is given by*

$$\delta \mathbf{W}_b^{(l-1)} \propto \mathbf{v}_{a,T+1}^{(l)} \mathbf{z}_{T+1}^{(l-1)\top} = \epsilon^{n-l} \left[\mathbf{W}_a^{(l)\top} \cdots \mathbf{W}_a^{(n-1)\top} (\mathbf{y}_{T+1} - \tilde{\mathbf{y}}_{T+1}) \right] \mathbf{z}_{T+1}^{(l-1)\top} + \mathcal{O}(\epsilon^{n-l+1}),$$

where $\tilde{\mathbf{y}}_{T+1} = \mathbf{Y}\mathbf{Z}^{(l-1)\top} \left(\mathbf{Z}^{(l-1)} \mathbf{Z}^{(l-1)\top} \right)^{-1} \mathbf{z}_{T+1}^{(l-1)}$ is the optimal linear inferred value for $\mathbf{y}_{T+1}$.

Biological interpretation Recall that the learning rule for the basal synapses is proportional to $\mathbf{v}_a^{(l)}$ (Eq. (14)). We argued in the previous section that this quantity can be implemented as the calcium plateau potential. In this light, Prop. 1 shows that the calcium plateau potential is implicitly encoding the prediction error. This is remarkable as in our algorithm, neither the prediction nor the prediction error is explicitly computed. However, via the combined learning dynamics of the synapses and the neural dynamics of the interneurons, the final apical signal implicitly encodes the prediction error. This is in spirit similar to difference target propagation (DTP) [47], however, in DTP, the difference between the forward pass and the target is *explicitly* computed and backpropagated.

5.2 Adaptive latent dimension

For optimization within a single hidden layer, we show that the system performs adaptive latent dimension discovery by thresholding eigenvalues of a Gramian including contributions from both adjacent layers. If we take the loss function in (9), setting $g_a = 1, g_b = \epsilon$, and optimizing over $\mathbf{W}_a, \mathbf{W}_b$, we get the objective $\mathrm{Tr}[(c+\epsilon)\mathbf{Z}^\top\mathbf{Z} - \mathbf{Z}^\top\mathbf{Z}\big[\epsilon\mathbf{X}^\top(\mathbf{X}\mathbf{X}^\top)^{-1}\mathbf{X} + \mathbf{Y}^\top\mathbf{Y}\big]$ for $\mathbf{Z}$. The next theorem indicates the solution:

Theorem 1. *Let the concatenated matrix $\mathbf{\Xi} = [\epsilon^{1/2}\mathbf{C}_x^{-1/2}\mathbf{X}, \mathbf{Y}] \in \mathbb{R}^{(d_X+d_Y)\times T}$ have SVD $\mathbf{\Xi} = \sum_{\alpha=1}^{(d_X+d_Y)} \mathbf{u}_\alpha \lambda_\alpha \mathbf{v}_\alpha^\top$ with $\{\mathbf{u}_\alpha \in \mathbb{R}^{(d_X+d_Y)}\}$ and $\{\mathbf{v}_\alpha \in \mathbb{R}^T\}$ both being sets of orthonormal vectors, with the convention that (λ_α) are sorted in decreasing order. We consider the minimization*

$$\min_{\mathbf{Z}\in\mathbb{R}^{d\times T},\ \mathbf{Z}\mathbf{Z}^\top \preccurlyeq I_d} \mathrm{Tr}\left[(c+\epsilon)\mathbf{Z}^\top\mathbf{Z} - \mathbf{Z}^\top\mathbf{Z}\big[\epsilon\mathbf{X}^\top(\mathbf{X}\mathbf{X}^\top)^{-1}\mathbf{X} + \mathbf{Y}^\top\mathbf{Y}\big]\right]. \tag{15}$$

Then one of the optimal solutions is given by $\hat{\mathbf{Z}} = \sum_{\alpha=1}^{D} \mathbf{w}_\alpha \mathbb{1}(\lambda_\alpha > c + \epsilon)\mathbf{v}_\alpha^\top$ *where* $\{\mathbf{w}_\alpha \in \mathbb{R}^d\}$ *is an arbitrary set of orthonormal vectors, with* $D = \min(d, d_X + d_Y)$.

Thus, the latent modes are found by thresholding the eignevalues [72] of the sum of Gramians [71] in Eq. (15). From this theorem, we derive the computational limits of $\epsilon \to \infty$ and $\epsilon \to 0$ in the supplementary materials. We also show that the neurons of the narrowest layer get whitened and the other layers become low-rank. For further details of these statements and all proofs see SM Sec. B.

Biological interpretation In our model, the number of neurons in the intermediate layers is fixed. However, the dimensionality of the representation encoded in these intermediate layers is determined dynamically according to the spectrum of the eigenvalue problem specified in Thm. 1. In particular, it is the quantity $c^{(l)}$, which encodes the somatic compartment leak conductance, that biologically sets an adaptive threshold for the dimensionality of the latent representation. In other words, lower (resp. higher) somatic leak corresponds to higher (resp. lower) sensitivity to the small eigenvalues of the above eigenvalue problem. For details of the interpretation of the eigenvalue problem in different limits ($\epsilon \to \infty$ and $\epsilon \to 0$), see SM Sec. B.

6 Numerical experiments

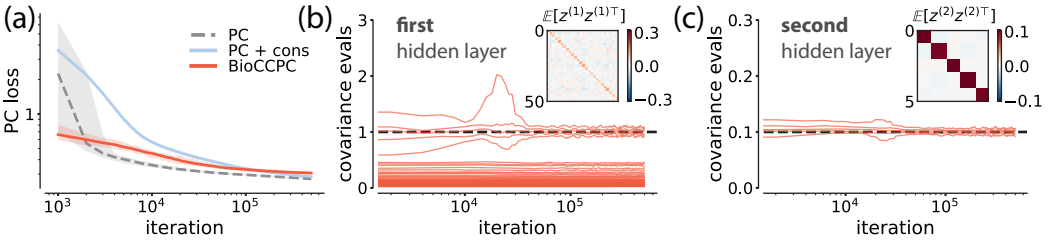

Figure 3: Our algorithm (BioCCPC) compares well to less biologically plausible predictive-coding (PC) implementations while enforcing an inequality constraint on the covariance of hidden-layer activations. Simulations run on the MNIST dataset using a network with two hidden layers, sizes 50 and 5. The shaded area represents the 95% range out of 40 runs. (a) Evolution of the predictive-coding loss, Eq. (6), on a validation set during learning. (b), (c) Evolution of the eigenvalues of the covariance matrix of hidden-layer activations in the two hidden layers. The dashed lines show the scale of the constraint; see SM Sec. A. Inset: covariance matrix at the end of training. Note how the activity is whitened in the second hidden layer (c), and low-rank in the first hidden layer (b).

We tested our BioCCPC algorithm on the MNIST [79] as well as other datasets.[4] We compared our results with a standard predictive coding network [4] with linear activation (PC) and with a version that includes a covariance constraint (PC+cons; Sec. C). We used unit variances, $\sigma^{(l)2} = 1$, in the PC networks, and matching conductances in our algorithm, $g_b^{(1)} = g_a^{(n)} = 1$, $g_a^{(l)} = g^{(l)} = \frac{1}{2}$ for other l.

Our method finds solutions that are almost optimal in terms of predictive-coding loss on a held-out validation set (see Fig. 3a) and are generally better than the solutions obtained from simply adding the covariance constraint to a predictive-coding network. Learning is slower in our network than in unconstrained PCN, as it takes time for the covariance constraint to be enforced. It is possible that different initialization schemes could improve the learning speed, but we leave this for future work. Note that since we focus on linear networks, metrics based on prediction accuracy are not very informative for the datasets we consider, which are not linearly separable.

The covariance constraint is saturated in the narrowest layer—in our example, the second hidden layer, of size 5—see Fig. 3c. The activation in the other hidden layer is rank-restricted by the narrowest layer, as described above, and so it is only whitened within a 5D subspace, Fig. 3b. In SM Sec. C, we provide further numerical experiments, exploring the effects of thresholding, the size of the network, and performance on other datasets. In all tested scenarios, BioCCPC achieves the same validation loss as PC (Figs. 5 and 6 of SM Sec. C). We also show that the forward and backward weights, $\mathbf{W}_b$ and $\mathbf{W}_a^\top$, are not symmetric, in accordance with biological observations, and explore their relationship.

[4]The code is available at `https://github.com/ttesileanu/bio-pcn`.

In SM Sec. D we compare PC and BioCCPC with regards to generalization. We find that there is very little generalization gap for either case (Fig. 9). This is due to the linearity of our model, which acts as a strong regularizer. Because of this, the potential generalization benefits of whitening the latent variables are not observed. Therefore, the primary benefit of the covariance constraint in this work is the elimination of weight sharing in $\mathbf{W}_a$ (cf. Sec. 3.1).

7 Conclusion

We have derived a biologically plausible algorithm for covariance-constrained predictive coding and have shown that it avoids the criticisms of PC and has many features in common with recent experimental observations. Our algorithm does not have symmetric forward and backward connectivity matrices, and does not need one-to-one connectivity between any neurons. Furthermore, we showed that the learning rules of our algorithm closely match experimental observations, and we connected our parameters to neurophysiological quantities. Using the simplicity afforded to us because of the linearity, we showed a number of interesting properties of our algorithm, including the effect of the somatic leak term, which acts as an adaptive thresholding mechanism. We also showed how our algorithm implicitly computes a difference term that it uses for learning. We argued that this term could be encoded in the calcium plateau potential, which is known to affect the learning of basal synapses. We hope that this concrete connection between our normative model and observation can further our understanding of cortical computation by guiding future experiments.

Acknowledgments and Disclosure of Funding

We would like to thank Colin Bredenberg for guidance and helpful comments throughout the project. SG, TT, YB, and DC were internally funded by the Simons Foundation. AS was partly funded by Simons Foundation Neuroscience grant SF 626323 during this work.

Promised changes

1. [IGNORE, LOW PRIORITY EXPERIMENT] Reviewer: Experiments are restricted to a toy model and dataset (MNIST).
   Us: Also note that in general, training a narrow architecture with a bottleneck (as we have in our networks) is more difficult than training a wide overparametrized network. This is in particular true for linear networks and especially for training algorithms that do not perform backpropagation. In this sense, we performed the experiments that we believed would be most challenging for our algorithm. We will add a paragraph explaining this in the experiment section.

Broader impact

We do not foresee any potential societal harm caused by our work. We hope our contribution moves forward the discussion on the relationship between multi-layer networks and cortical structure.

References

[1] Rajesh P.N. Rao and Dana H. Ballard. Predictive coding in the visual cortex: A functional interpretation of some extra-classical receptive-field effects. *Nature Neuroscience*, 1999.

[2] Karl Friston. A theory of cortical responses. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 2005.

[3] Mandyam Veerambudi Srinivasan, Simon Barry Laughlin, A. Dubs, and George Adrian Horridge. Predictive coding: a fresh view of inhibition in the retina. *Proceedings of the Royal Society of London. Series B. Biological Sciences*, 216(1205):427–459, 1982.

[4] James C.R. Whittington and Rafal Bogacz. An approximation of the error backpropagation algorithm in a predictive coding network with local hebbian synaptic plasticity. *Neural Computation*, 29(5):1229–1262, may 2017.

[5] Tommaso Salvatori, Luca Pinchetti, Beren Millidge, Yuhang Song, Tianyi Bao, Rafal Bogacz, and Thomas Lukasiewicz. Learning on arbitrary graph topologies via predictive coding, 2022.

[6] André Ofner and Sebastian Stober. Differentiable generalised predictive coding, 2021.

[7] Martin Boerlin, Christian K Machens, and Sophie Denève. Predictive coding of dynamical variables in balanced spiking networks. *PLoS computational biology*, 9(11):e1003258, 2013.

[8] Jonathan Kadmon, Jonathan Timcheck, and Surya Ganguli. Predictive coding in balanced neural networks with noise, chaos and delays. *Advances in neural information processing systems*, 33:16677–16688, 2020.

[9] Ryan Smith, Paul Badcock, and Karl J Friston. Recent advances in the application of predictive coding and active inference models within clinical neuroscience. *Psychiatry and Clinical Neurosciences*, 75(1):3–13, 2021.

[10] Paul F. Kinghorn, Beren Millidge, and Christopher L. Buckley. Habitual and reflective control in hierarchical predictive coding, 2021.

[11] Tommaso Salvatori, Yuhang Song, Yujian Hong, Lei Sha, Simon Frieder, Zhenghua Xu, Rafal Bogacz, and Thomas Lukasiewicz. Associative Memories via Predictive Coding. In M Ranzato, A Beygelzimer, Y Dauphin, P S Liang, and J Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 3874–3886. Curran Associates, Inc., 2021.

[12] Beren Millidge, Anil Seth, and Christopher L Buckley. Predictive coding: a theoretical and experimental review, 2021.

[13] Beren Millidge, Tommaso Salvatori, Yuhang Song, Rafal Bogacz, and Thomas Lukasiewicz. Predictive coding: Towards a future of deep learning beyond backpropagation?, 2022.

[14] André M. Bastos, W. Martin Usrey, Rick A Adams, George R. Mangun, Pascal Fries, and Karl J. Friston. Canonical microcircuits for predictive coding. *Neuron*, 76:695–711, 2012.

[15] Micha Heilbron and Maria Chait. Great Expectations: Is there Evidence for Predictive Coding in Auditory Cortex? *Neuroscience*, 389:54–73, 2018.

[16] Naoki Kogo and Chris Trengove. Is predictive coding theory articulated enough to be testable? *Frontiers in Computational Neuroscience*, 9, 2015.

[17] Nelson Spruston. Pyramidal neurons: dendritic structure and synaptic integration. *Nature Reviews Neuroscience*, 9(3):206–221, 2008.

[18] Matthew Larkum. A cellular mechanism for cortical associations: an organizing principle for the cerebral cortex. *Trends in Neurosciences*, 36(3):141–151, 2013.

[19] Per Jesper Sjöström and Michael Häusser. A cooperative switch determines the sign of synaptic plasticity in distal dendrites of neocortical pyramidal neurons. *Neuron*, 51(2):227–238, 2006.

[20] Nace L Golding, Nathan P Staff, and Nelson Spruston. Dendritic spikes as a mechanism for cooperative long-term potentiation. *Nature*, 418(6895):326–331, 2002.

[21] Frédéric Gambino, Stéphane Pagès, Vassilis Kehayas, Daniela Baptista, Roberta Tatti, Alan Carleton, and Anthony Holtmaat. Sensory-evoked LTP driven by dendritic plateau potentials in vivo. *Nature*, 515(7525):116–119, 2014.

[22] Guy Major, Matthew E. Larkum, and Jackie Schiller. Active properties of neocortical pyramidal neuron dendrites. *Annual Review of Neuroscience*, 36(1):1–24, 2013. PMID: 23841837.

[23] Katie C Bittner, Christine Grienberger, Sachin P Vaidya, Aaron D Milstein, John J Macklin, Junghyup Suh, Susumu Tonegawa, and Jeffrey C Magee. Conjunctive input processing drives feature selectivity in hippocampal ca1 neurons. *Nature neuroscience*, 18(8):1133, 2015.

[24] Katie C Bittner, Aaron D Milstein, Christine Grienberger, Sandro Romani, and Jeffrey C Magee. Behavioral time scale synaptic plasticity underlies ca1 place fields. *Science*, 357(6355):1033–1036, 2017.

[25] Jeffrey C. Magee and Christine Grienberger. Synaptic plasticity forms and functions. *Annual Review of Neuroscience*, 43, 2020.

[26] Jason Hardie and Nelson Spruston. Synaptic depolarization is more effective than back-propagating action potentials during induction of associative long-term potentiation in hippocampal pyramidal neurons. *Journal of Neuroscience*, 29(10):3233–3241, 2009.

[27] Konrad P Körding and Peter König. Supervised and unsupervised learning with two sites of synaptic integration. *Journal of Computational Neuroscience*, 11(3):207–215, 2001.

[28] Robert Urbanczik and Walter Senn. Learning by the Dendritic Prediction of Somatic Spiking. *Neuron*, 81(3):521–528, 2014.

[29] Jordan Guergiuev, Timothy P Lillicrap, and Blake A Richards. Deep learning with segregated dendrites. *arXiv preprint arXiv:1610.00161*, 2016.

[30] João Sacramento, Rui Ponte Costa, Yoshua Bengio, and Walter Senn. Dendritic cortical microcircuits approximate the backpropagation algorithm. *Advances in Neural Information Processing Systems*, 2018-Decem(Nips):8721–8732, oct 2018.

[31] Alexandre Payeur, Jordan Guerguiev, Friedemann Zenke, Blake A Richards, and Richard Naud. Burst-dependent synaptic plasticity can coordinate learning in hierarchical circuits. *bioRxiv*, 2020.

[32] Albert Gidon, Timothy Adam Zolnik, Pawel Fidzinski, Felix Bolduan, Athanasia Papoutsi, Panayiota Poirazi, Martin Holtkamp, Imre Vida, and Matthew Evan Larkum. Dendritic action potentials and computation in human layer 2/3 cortical neurons. *Science*, 367(6473):83–87, jan 2020.

[33] Aaron D. Milstein, Yiding Li, Katie C. Bittner, Christine Grienberger, Ivan Soltesz, Jeffrey C. Magee, and Sandro Romani. Bidirectional synaptic plasticity rapidly modifies hippocampal representations independent of correlated activity. *bioRxiv*, 2020.

[34] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning internal representations by error propagation. In David E. Rumelhart and James L. Mcclelland, editors, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Volume 1: Foundations*, pages 318–362. MIT Press, Cambridge, MA, 1986.

[35] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[36] Seungjin Choi, Andrzej Cichocki, Hyung-Min Park, and Soo-Young Lee. Blind source separation and independent component analysis: A review. *Neural Information Processing-Letters and Reviews*, 6(1):1–57, 2005.

[37] Cengiz Pehlevan and Dmitri B Chklovskii. Neuroscience-inspired online unsupervised learning algorithms: Artificial neural networks. *IEEE Signal Processing Magazine*, 36(6):88–96, 2019.

[38] Adrian A Wanner and Rainer W Friedrich. Whitening of odor representations by the wiring diagram of the olfactory bulb. *Nature neuroscience*, 23(3):433–442, 2020.

[39] Guillaume Desjardins, Karen Simonyan, Razvan Pascanu, and Koray Kavukcuoglu. Natural neural networks, 2015.

[40] Yuki Fujimoto and Toru Ohira. A neural network model with bidirectional whitening. In *Artificial Intelligence and Soft Computing*, pages 47–57. Springer International Publishing, 2018.

[41] Ping Luo. Learning deep architectures via generalized whitened neural networks. In *International Conference on Machine Learning*, pages 2238–2246. PMLR, 2017.

[42] P. J. Werbos. *Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences*. PhD thesis, Harvard University, 1974.

[43] Francis Crick. The recent excitement about neural networks. *Nature*, 337(6203):129–132, 1989.

[44] Yann Le Cun. Learning Process in an Asymmetric Threshold Network. In *Disordered Systems and Biological Organization*, pages 233–240. Springer, 1986.

[45] Y. Le Cun and Franccoise Fogelman-Soulié. Modèles connexionnistes de l'apprentissage. *Intellectica. Revue de l'Association pour la Recherche Cognitive*, 1987.

[46] John F. Kolen and Jordan B. Pollack. Backpropagation without weight transport. *Proceedings of 1994 IEEE International Conference on Neural Networks (ICNN'94)*, 3:1375–1380 vol.3, 1994.

[47] Dong-Hyun Lee, Saizheng Zhang, Antoine Biard, and Yoshua Bengio. Target Propagation. *Iclr 15*, 1(3):1–16, 2014.

[48] Timothy P. Lillicrap, Daniel Cownden, Douglas B. Tweed, and Colin J. Akerman. Random synaptic feedback weights support error backpropagation for deep learning. *Nature Communications*, 7:1–10, 2016.

[49] Benjamin Scellier and Yoshua Bengio. Equilibrium propagation: Bridging the gap between energy-based models and backpropagation. *Frontiers in Computational Neuroscience*, 11(May), 2017.

[50] Dmitry Krotov and John J. Hopfield. Unsupervised learning by competing hidden units. *Proceedings of the National Academy of Sciences*, 116(16):7723–7731, 2019.

[51] Mohamed Akrout, Collin Wilson, Peter C. Humphreys, Timothy Lillicrap, and Douglas Tweed. Deep Learning without Weight Transport. *arXiv*, apr 2019.

[52] Jordan Guerguiev, Konrad Kording, and Blake Richards. Spike-based causal inference for weight alignment. *arXiv preprint arXiv:1910.01689*, 10 2019.

[53] Maxence Ernoult, Julie Grollier, Damien Querlioz, Yoshua Bengio, and Benjamin Scellier. Equilibrium Propagation with Continual Weight Updates. *arXiv preprint arXiv:2005.04168*, pages 1–37, 2020.

[54] Will Xiao, Honglin Chen, Qianli Liao, and Tomaso A. Poggio. Biologically-plausible learning algorithms can scale to large datasets. *CoRR*, abs/1811.03567, 2018.

[55] Siavash Golkar, David Lipshutz, Yanis Bahroun, Anirvan Sengupta, and Dmitri Chklovskii. A simple normative network approximates local non-Hebbian learning in the cortex. In H Larochelle, M Ranzato, R Hadsell, M F Balcan, and H Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 7283–7295. Curran Associates, Inc., 2020.

[56] Siavash Golkar, David Lipshutz, Yanis Bahroun, Anirvan M. Sengupta, and Dmitri B. Chklovskii. A biologically plausible neural network for local supervision in cortical microcircuits, 2020.

[57] Yoshua Bengio. Deriving differential target propagation from iterating approximate inverses, 2020.

[58] Jyotibdha Acharya, Arindam Basu, Robert Legenstein, Thomas Limbacher, Panayiota Poirazi, and Xundong Wu. Dendritic Computing: Branching Deeper into Machine Learning. *Neuroscience*, 489:275–289, 2022.

[59] Timothy P Lillicrap, Adam Santoro, Luke Marris, Colin J Akerman, and Geoffrey Hinton. Backpropagation and the brain. *Nature Reviews Neuroscience*, 21(6):335–346, 2020.

[60] João Sacramento, Rui Ponte Costa, Yoshua Bengio, and Walter Senn. Dendritic error backpropagation in deep cortical microcircuits. *arXiv preprint arXiv:1801.00062*, pages 1–37, 2017.

[61] Beren Millidge, Alexander Tschantz, and Christopher L. Buckley. Predictive coding approximates backprop along arbitrary computation graphs, 2020.

[62] Yuhang Song, Thomas Lukasiewicz, Zhenghua Xu, and Rafal Bogacz. Can the Brain Do Backpropagation? — Exact Implementation of Backpropagation in Predictive Coding Networks. In H Larochelle, M Ranzato, R Hadsell, M F Balcan, and H Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 22566–22579. Curran Associates, Inc., 2020.

[63] Tommaso Salvatori, Yuhang Song, Thomas Lukasiewicz, Rafal Bogacz, and Zhenghua Xu. Predictive coding can do exact backpropagation on convolutional and recurrent neural networks. *CoRR*, abs/2103.03725, 2021.

[64] Robert Rosenbaum. On the relationship between predictive coding and backpropagation, 2021.

[65] Matthew E. Larkum, Thomas Nevian, Maya Sandler, Alon Polsky, and Jackie Schiller. Synaptic integration in tuft dendrites of layer 5 pyramidal neurons: A new unifying principle. *Science*, 325(5941):756–760, 2009.

[66] Hiroto Takahashi and Jeffrey C Magee. Pathway interactions and synaptic plasticity in the dendritic tuft regions of CA1 pyramidal neurons. *Neuron*, 62(1):102–111, 2009.

[67] Thomas Klausberger, Peter J Magill, László F Márton, J David B Roberts, Philip M Cobden, György Buzsáki, and Peter Somogyi. Brain-state-and cell-type-specific firing of hippocampal interneurons in vivo. *Nature*, 421(6925):844–848, 2003.

[68] Therese Riedemann. Diversity and function of somatostatin-expressing interneurons in the cerebral cortex. *International Journal of Molecular Sciences*, 20(12):2952, 2019.

[69] Karl Friston. The free-energy principle: A unified brain theory?, 2010.

[70] Ryota Kanai, Yutaka Komura, Stewart Shipp, and Karl Friston. Cerebral hierarchies: predictive processing, precision and the pulvinar. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 370(1668):20140169, 2015.

[71] David Lipshutz, Yanis Bahroun, Siavash Golkar, Anirvan M. Sengupta, and Dmitri B. Chklovskii. A Biologically Plausible Neural Network for Multichannel Canonical Correlation Analysis. *Neural Computation*, 33(9):2309–2352, 08 2021.

[72] Cengiz Pehlevan and Dmitri Chklovskii. A Normative Theory of Adaptive Dimensionality Reduction in Neural Networks. In C Cortes, N Lawrence, D Lee, M Sugiyama, and R Garnett, editors, *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015.

[73] Charles D Gilbert and Wu Li. Top-down influences on visual processing. *Nature Reviews Neuroscience*, 14(5):350–363, 2013.

[74] Georg B Keller and Thomas D Mrsic-Flogel. Predictive processing: a canonical cortical computation. *Neuron*, 100(2):424–435, 2018.

[75] Björn M. Kampa, Johannes J. Letzkus, and Greg J. Stuart. Dendritic mechanisms controlling spike-timing-dependent synaptic plasticity. *Trends in Neurosciences*, 30(9):456–463, 2007.

[76] Andrew M Saxe, James L McClelland, and Surya Ganguli. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. *arXiv preprint arXiv:1312.6120*, 2013.

[77] Sanjeev Arora, Nadav Cohen, and Elad Hazan. On the optimization of deep networks: Implicit acceleration by overparameterization. In *International Conference on Machine Learning*, pages 244–253. PMLR, 2018.

[78] Simon S Du, Wei Hu, and Jason D Lee. Algorithmic regularization in learning deep homogeneous models: Layers are automatically balanced. *Advances in Neural Information Processing Systems*, 31, 2018.

[79] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [Yes]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [Yes]
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

Supplementary Materials

This is the supplementary materials section for the NeurIPS 2022 submission titled "Constrained Predictive Coding as a Biologically Plausible Model of the Cortical Hierarchy".

A Derivation of upper bound

In this section we give a more detailed derivation of our framework, using a slightly modified order of steps compared to the main text. This should help clarify some aspects of the final algorithm.

Similar to the main text, we start with the predictive coding objective and introduce the weight doubles. We arrive at

$$\min_{\mathbf{Z},\mathbf{W}} L = \min_{\mathbf{Z},\mathbf{W}_a,\mathbf{W}_b} \frac{1}{2}\sum_{l=1}^{n-1}\Big[g_b^{(l)}\big\|\mathbf{Z}^{(l)} - \mathbf{W}_b^{(l-1)}\mathbf{Z}^{(l-1)}\big\|_F^2 + g_a^{(l+1)}\big\|\mathbf{Z}^{(l+1)} - \mathbf{W}_a^{(l)}\mathbf{Z}^{(l)}\big\|_F^2 + c^{(l)}\big\|\mathbf{Z}^{(l)}\mathbf{Z}^{(l)\top}\big\|_F^2\Big], \quad (16)$$

where we have also introduced a quadratic cost (a prior on $\mathbf{Z}$) with coefficient $c^{(l)}$. We now impose the constraint on the empirical covariance, $\frac{1}{T}\mathbf{Z}^{(l)}\mathbf{Z}^{(l)\top} \preceq \rho^{(l)}\mathbf{I}$, of the internal variables $\mathbf{Z}^{(l)}$[5]:

$$\min_{\mathbf{Z},\mathbf{W}} L \leq \min_{\substack{\mathbf{Z},\mathbf{W}_a,\mathbf{W}_b \\ \frac{1}{T}\mathbf{Z}^{(l)}\mathbf{Z}^{(l)\top}\preceq\rho^{(l)}\mathbf{I}}} \frac{1}{2}\sum_{l=1}^{n-1}\Big[g_b^{(l)}\big\|\mathbf{Z}^{(l)} - \mathbf{W}_b^{(l-1)}\mathbf{Z}^{(l-1)}\big\|_F^2 + g_a^{(l+1)}\big\|\mathbf{Z}^{(l+1)} - \mathbf{W}_a^{(l)}\mathbf{Z}^{(l)}\big\|_F^2 + c^{(l)}\big\|\mathbf{Z}^{(l)}\mathbf{Z}^{(l)\top}\big\|_F^2\Big]. \quad (17)$$

Expanding this and using the constraint we again upper bound the objective:

$$\min_{\mathbf{Z},\mathbf{W}} L \leq \min_{\substack{\mathbf{Z},\mathbf{W}_a,\mathbf{W}_b \\ \frac{1}{T}\mathbf{Z}^{(l)}\mathbf{Z}^{(l)\top}\preceq\rho^{(l)}\mathbf{I}}} \frac{1}{2}\sum_{l=1}^{n-1}\Big[g_b^{(l)}\big\|\mathbf{Z}^{(l)} - \mathbf{W}_b^{(l-1)}\mathbf{Z}^{(l-1)}\big\|_F^2 + g_a^{(l+1)}\big\|\mathbf{Z}^{(l+1)}\mathbf{Z}^{(l+1)\top}\big\|_F^2 + g_a^{(l+1)}\operatorname{Tr}\big(-2\mathbf{Z}^{(l+1)\top}\mathbf{W}_a^{(l)}\mathbf{Z}^{(l)} + T\rho^{(l)}\mathbf{W}_a^{(l)\top}\mathbf{W}_a^{(l)}\big) + c^{(l)}\big\|\mathbf{Z}^{(l)}\mathbf{Z}^{(l)\top}\big\|_F^2\Big], \quad (18)$$

We next implement the inequality constraint using positive-definite Lagrange multipliers:

$$\min_{\mathbf{Z},\mathbf{W}} L \leq \min_{\mathbf{Z},\mathbf{W}_a,\mathbf{W}_b} \max_{\mathbf{Q}} \sum_{l=1}^{n-1}\tilde{L}^{(l)}, \quad (19)$$

where

$$\begin{aligned}\tilde{L}^{(l)} = \frac{1}{2}\Big[&g_b^{(l)}\big\|\mathbf{Z}^{(l)} - \mathbf{W}_b^{(l-1)}\mathbf{Z}^{(l-1)}\big\|_F^2 + c_q^{(l)}\operatorname{Tr}\mathbf{Q}^{(l)\top}\mathbf{Q}^{(l)}\big(\mathbf{Z}^{(l)}\mathbf{Z}^{(l)\top} - T\rho^{(l)}\mathbf{I}\big) \\ &+ g_a^{(l+1)}\operatorname{Tr}\big(-2\mathbf{Z}^{(l+1)\top}\mathbf{W}_a^{(l)}\mathbf{Z}^{(l)} + T\rho^{(l)}\mathbf{W}_a^{(l)\top}\mathbf{W}_a^{(l)}\big) \\ &+ c^{(l)}\big\|\mathbf{Z}^{(l)}\mathbf{Z}^{(l)\top}\big\|_F^2 + g_a^{(l+1)}\big\|\mathbf{Z}^{(l+1)}\mathbf{Z}^{(l+1)\top}\big\|_F^2\Big],\end{aligned} \quad (20)$$

where we have added a positive constant $c_q^{(l)}$ defining the coefficient of the inequality constraint. This constant does not affect the solution of the optimization problem but does affect the details of the neural dynamics. Motivated by biological arguments[6], we set this constant equal to $g_a^{(l)}$. In doing so, the interneuron variables no longer need a $(1/g_a^{(l)})$ in their definition, as in the main text, and are instead given by $\mathbf{n}^{(l)} = \mathbf{Q}^{(l)}\mathbf{z}^{(l)}$.

[5]The factors $\rho^{(l)}$ which set the scale of the constraint were set to 1 in the derivation in the main text for brevity. However, it is sensible for this scale factor to vary from layer to layer and possibly depend on the layer width. In Sec. C, we explore the effects of this parameter on the performance of the network.

[6]Since the feedback of the interneurons is received and integrated in the apical compartment, we expect the apical-to-soma conductance to also be present here.

A final upper bound to (19) can be found if each $\tilde{L}^{(l)}$ is optimized individually. Explicitly, we perform gradient descent of each variable, $\mathbf{Z}^{(l)}, \mathbf{W}_a^{(l)}, \mathbf{W}_b^{(l)}$, only with respect to $\tilde{L}^{(l)}$ and not with respect to the entire objective L:

$$\begin{aligned} \delta \mathbf{Z}^{(l)} &= \nabla_{\mathbf{Z}^{(l)}} L \;\rightarrow\; \nabla_{\mathbf{Z}^{(l)}} \tilde{L}^{(l)}, \\ \delta \mathbf{W}_a^{(l)} &= \nabla_{\mathbf{W}_a^{(l)}} L \;\rightarrow\; \nabla_{\mathbf{W}_a^{(l)}} \tilde{L}^{(l)}, \\ \delta \mathbf{W}_b^{(l)} &= \nabla_{\mathbf{W}_b^{(l)}} L \;\rightarrow\; \nabla_{\mathbf{W}_b^{(l)}} \tilde{L}^{(l)}. \end{aligned} \tag{21}$$

This affords us an upper bound because we are no longer finding the actual minimum of L. Symbolically this can be written as in (9):

$$\min_{\mathbf{Z},\mathbf{W}} L \leq \sum_{l=1}^{n-1} \min_{\mathbf{Z},\mathbf{W}_a,\mathbf{W}_b} \max_{\mathbf{Q}} \tilde{L}^{(l)}, \tag{22}$$

where the precise definition of this equation is given by the update equations (21). Note that these updates are no longer curl-free and hence cannot be found as the gradient of a single objective function.

B Theoretical section proofs

Note that the definition of ϵ in Proposition 1 is different from the definition in other sections.

B.1 Proof of Proposition 1

To prove Proposition 1, we first show that the latent variables $\mathbf{z}^{(l)}$ have a specific form. As in the main text, we define $\epsilon^{(l)} = g_a^{(l)}/g_b^{(l)}$ and, for simplicity, we assume $\epsilon^{(l)} = \epsilon$ is the same for all layers. For simplicity also, we assume $c^{(l)}$ (equivalently $d^{(l)}$ in previous section) is equal to zero, and use $c_q^{(l)} = g_a^{(l)}$ as in the previous section. This choice makes the notation simpler without changing our results. Equation (10) becomes

$$\dot{\mathbf{z}}^{(l)} = g_b^{(l)} \mathbf{W}_b^{(l-1)} \mathbf{z}^{(l-1)} + g_a^{(l)} \mathbf{W}_a^{(l)\top} \mathbf{z}^{(l+1)} - g_b^{(l)} \mathbf{z}^{(l)} - g_a^{(l)} \mathbf{Q}^{(l)\top} \mathbf{Q}^{(l)} \mathbf{z}^{(l)}. \tag{23}$$

Proposition 2. *Assume the neural dynamics given in Eq. (23) are at equilibrium. Then the latent variables take the form*

$$\mathbf{z}^{(n-k)} = \mathbf{B}_k \left(\mathbf{W}_b^{(n-k-1)} \mathbf{z}^{(n-k-1)} + \epsilon^k \mathbf{A}_k \mathbf{W}_a^{(n-1)\top} \mathbf{y} \right), \tag{24}$$

where $\mathbf{B}_k$ and $\mathbf{A}_k$ are independent of y and defined recursively via

$$\begin{aligned} \mathbf{A}_{k+1} &= \mathbf{W}_a^{(n-k-1)\top} \mathbf{B}_k \mathbf{A}_k, \\ \mathbf{B}_{k+1} &= \left(1 - \epsilon \mathbf{F}^{(n-k-1)} \mathbf{W}_a^{(n-k-1)\top} \mathbf{B}_k \mathbf{W}_b^{(n-k-1)} \right)^{-1} \mathbf{F}^{(n-k-1)}, \end{aligned}$$

with $\mathbf{F}^{(l)} = \left(\mathbf{I} + \epsilon \mathbf{Q}^{(l)\top} \mathbf{Q}^{(l)} \right)^{-1}$ and

$$\mathbf{B}_1 = \mathbf{F}^{(n-1)}, \quad \mathbf{A}_1 = 1.$$

Proof. We prove this via induction. Starting from the neural dynamics Eq. (23), at equilibrium ($\dot{\mathbf{z}}^{(l)} = 0$) we have:

$$\mathbf{z}^{(l)} = \mathbf{F}^{(l)} \left(\mathbf{W}_b^{(l-1)} \mathbf{z}^{(l-1)} + \epsilon \mathbf{W}_a^{(l)\top} \mathbf{z}^{(l+1)} \right).$$

Setting $l = n-1$ we get:

$$\mathbf{z}^{(n-1)} = \mathbf{F}^{(n-1)} \left(\mathbf{W}_b^{(n-2)} \mathbf{z}^{(n-2)} + \epsilon \mathbf{W}_a^{(n-1)\top} \mathbf{y} \right),$$

which satisfies the proposed form in Eq. (24) for $k = 1$. Now, we assume that the form in Eq. (24) holds for $z^{(n-k)}$, and show that it also holds for $z^{(n-k-1)}$. We have

$$\begin{aligned}\mathbf{z}^{(n-k-1)} =& \mathbf{F}^{(n-k-1)} \left(\mathbf{W}_b^{(n-k-2)} \mathbf{z}^{(n-k-2)} + \epsilon \mathbf{W}_a^{(n-k-1)\top} \mathbf{z}^{(n-k)} \right) \\ =& \mathbf{F}^{(n-k-1)} \left(\mathbf{W}_b^{(n-k-2)} \mathbf{z}^{(n-k-2)} + \epsilon \mathbf{W}_a^{(n-k-1)\top} \mathbf{B}_k \left(\mathbf{W}_b^{(n-k-1)} \mathbf{z}^{(n-k-1)} + \epsilon^k \mathbf{A}_k \mathbf{W}_a^{(n-1)\top} \mathbf{y} \right) \right).\end{aligned}$$

Collecting $\mathbf{z}^{(n-k-1)}$ and multiplying by the inverse on the left we arrive at:

$$\begin{aligned}\mathbf{z}^{(n-k-1)} = \left(1 - \epsilon \mathbf{F}^{(n-k-1)} \mathbf{W}_a^{(n-k-1)\top} \mathbf{B}_k \mathbf{W}_b^{(n-k-1)} \right)^{-1} \mathbf{F}^{(n-k-1)} \times \\ \times \left(\mathbf{W}_b^{(n-k-2)} \mathbf{z}^{(n-k-2)} + \epsilon^{k+1} \mathbf{W}_a^{(n-k-1)\top} \mathbf{B}_k \mathbf{A}_k \mathbf{W}_a^{(n-1)\top} \mathbf{y} \right),\end{aligned}$$

as proposed. □

Note that, whereas Proposition 1 holds only in the $\epsilon \to 0$ limit, Proposition 2 holds for general ϵ. Using this result we show that the relationship between the activity, feedback and feedforward inputs of the lth module has a simple form given in the following corollary:

Corollary 1. *The leading order in ϵ of the dependence of $\mathbf{z}^{(l)}$ and $\mathbf{z}^{(l+1)}$ on $\mathbf{z}^{(l-1)}$ and $\mathbf{y}$, which has non-zero dependence on $\mathbf{y}$, is given by*

$$\begin{aligned}\mathbf{z}^{(l)} &= \mathbf{C}_1^{(l)} \mathbf{z}^{(l-1)} + \mathcal{O}(\epsilon^{n-l}), \\ \mathbf{z}^{(l+1)} &= \mathbf{C}_2^{(l)} \mathbf{z}^{(l-1)} + \epsilon^{n-l-1} \mathbf{W}_a^{(l+1)\top} \cdots \mathbf{W}_a^{(n-1)\top} \mathbf{y} + \mathcal{O}(\epsilon^{n-l}),\end{aligned}$$

where $\mathbf{C}_1^{(l)}$ and $\mathbf{C}_2^{(l)}$ are respectively the expansions of $\mathbf{B}_{n-l} \mathbf{W}_b^{(l-1)}$ and $\mathbf{B}_{n-l-1} \mathbf{W}_b^{(l)} \mathbf{B}_{n-l} \mathbf{W}_b^{(l-1)}$ up to order ϵ^{n-l-1}.

Proof. The result follows directly from applying Eq. (24) twice and then dropping all but the leading dependence on ϵ in the coefficient of the $\mathbf{y}$ term. □

Now, we restate and prove Proposition 1, which can be viewed as a generalization of the results in [55]. Note that in this proposition, the exact form of $\mathbf{C}_1^{(l)}$ and $\mathbf{C}_2^{(l)}$ is not important. All that matters is that both $\mathbf{z}^{(l)}$ and $\mathbf{z}^{(l+1)}$ have a linear dependence on $\mathbf{z}^{(l-1)}$. Since the dependence of $\mathbf{z}^{(l)}$ on $\mathbf{y}$ is subleading compared to the dependence of $\mathbf{z}^{(l+1)}$ on $\mathbf{y}$, this can be dropped in the limit of $\epsilon \to 0$.

Proposition 1. *Assume that the learning rules (Eqs. 11) are at equilibrium and we receive a new datapoint given by $\mathbf{x}_{T+1}, \mathbf{y}_{T+1}$. For $c^{(l)} = 0$ and in the limit of $\epsilon \equiv g_a / g_b \to 0$, the leading term in ϵ for the forward weight updates $\delta \mathbf{W}_b^{(l-1)}$ for this new sample is given by*

$$\delta \mathbf{W}_b^{(l-1)} \propto \mathbf{v}_{a,T+1}^{(l)} \mathbf{z}_{T+1}^{(l-1)\top} = \epsilon^{n-l} \left[\mathbf{W}_a^{(l)\top} \cdots \mathbf{W}_a^{(n-1)\top} (\mathbf{y}_{T+1} - \tilde{\mathbf{y}}_{T+1}) \right] \mathbf{z}_{T+1}^{(l-1)\top} + \mathcal{O}(\epsilon^{n-l+1}),$$

where $\tilde{\mathbf{y}}_{T+1} = \mathbf{Y} \mathbf{Z}^{(l-1)\top} \left(\mathbf{Z}^{(l-1)} \mathbf{Z}^{(l-1)\top} \right)^{-1} \mathbf{z}_{T+1}^{(l-1)}$ is the optimal linear inferred value for $\mathbf{y}_T$.

Proof. The statement follows from two applications of the update rule for $\mathbf{W}_b$ given in Eq. (11a) for $\mathbf{W}_b^{(l-1)}$. Putting $c^{(l)} = 0$ we have:

$$\delta \mathbf{W}_b^{(l-1)} \propto \epsilon \big(\mathbf{W}_a^{(l)\top} \mathbf{z}^{(l+1)} - \mathbf{Q}^{(l)\top} \mathbf{Q}^{(l)} \mathbf{z}^{(l)} \big) \mathbf{z}^{(l-1)\top}. \tag{25}$$

The equilibrium condition given by $\delta \mathbf{W}_b = 0$ is given by averaging over the entire dataset:

$$\mathbf{Q}^{(l)\top} \mathbf{Q}^{(l)} \mathbf{Z}^{(l)} \mathbf{Z}^{(l-1)\top} = \mathbf{W}_a^{(l)\top} \mathbf{Z}^{(l+1)} \mathbf{Z}^{(l-1)\top}. \tag{26}$$

Now we plug the derived values in Corollary 1. After right-multiplying by $(\mathbf{Z}^{(l-1)} \mathbf{Z}^{(l-1)\top})^{-1}$ we get:

$$\mathbf{Q}^{(l)\top} \mathbf{Q}^{(l)} \mathbf{C}_1 = \mathbf{W}_a^{(l)\top} \mathbf{C}_2 + \epsilon^{n-l-1} \mathbf{W}_a^{(l)\top} \cdots \mathbf{W}_a^{(n-1)\top} \mathbf{Y} \mathbf{Z}^{(l-1)\top} \left(\mathbf{Z}^{(l-1)} \mathbf{Z}^{(l-1)\top} \right)^{-1}. \tag{27}$$

Now, we look at the update for a new sample $T+1$. Plugging the above as well as the equations from Corollary 1 into the update rule we see that the $\mathbf{C}_2^{(l)}$ dependent term cancels and we are only left with the term depending on $\mathbf{y}$:

$$\begin{aligned}\delta \mathbf{W}_b^{(l-1)} &\propto \epsilon^{n-l}\big(\mathbf{W}_a^{(l)\top}\mathbf{z}_{T+1}^{(l+1)} - \mathbf{Q}^{(l)\top}\mathbf{Q}^{(l)}\mathbf{z}_{T+1}^{(l)}\big)\mathbf{z}_{T+1}^{(l-1)\top} \\ &= \epsilon^{n-l}\left[\mathbf{W}_a^{(l)\top}\cdots\mathbf{W}_a^{(n-1)\top}(\mathbf{y}_{T+1}-\tilde{\mathbf{y}}_{T+1})\right]\mathbf{z}^{(l-1)\top},\end{aligned}$$

with $\tilde{\mathbf{y}}_{T+1}$ as defined in the proposition. □

B.2 Proof of Theorem 1 and consequences

In this section we look at some properties of the within-layer computations of CCPC. We will focus on the computations within the lth layer and for clarity of notation, for this subsection only, we rename $\mathbf{Z}^{(l)} \to \mathbf{Z}$, $\mathbf{Z}^{(l+1)} \to \mathbf{Y}$, and $\mathbf{Z}^{(l-1)} \to \mathbf{X}$. We will assume $\rho^{(l)} = 1$ for brevity. The results have straight-forward generalizations for the case where $\rho^{(l)} \neq 1$.

Theorem 1. *Let the concatenated matrix $\mathbf{\Xi} = \mathbf{U}\mathbf{\Lambda}\mathbf{V}^\top = [\epsilon^{1/2}\mathbf{C}_x^{-1/2}\mathbf{X}, \mathbf{Y}] \in \mathbb{R}^{(d_X+d_Y)\times T}$ have singular value decomposition $\mathbf{\Xi} = \sum_{\alpha=1}^{(d_X+d_Y)} \mathbf{u}_\alpha \lambda_\alpha \mathbf{v}_\alpha^\top$, with $\{\mathbf{u}_\alpha \in \mathbb{R}^{(d_X+d_Y)}\}$ and $\{\mathbf{v}_\alpha \in \mathbb{R}^T\}$ both being sets of orthonormal vectors, and singular values (λ_α) sorted in decreasing order. We consider the minimization*

$$\min_{\substack{\mathbf{Z}\in\mathbb{R}^{d\times N} \\ \mathbf{Z}\mathbf{Z}^\top \preccurlyeq I_d}} \operatorname{Tr}\left[(c+\epsilon)\mathbf{Z}^\top\mathbf{Z} - \mathbf{Z}^\top\mathbf{Z}\big[\epsilon\mathbf{X}^\top(\mathbf{X}\mathbf{X}^\top)^{-1}\mathbf{X} + \mathbf{Y}^\top\mathbf{Y}\big]\right]. \tag{28}$$

Then one of the optimal solutions is given by $\hat{\mathbf{Z}} = \sum_{\alpha=1}^{D} \mathbf{w}_\alpha \mathbb{1}(\lambda_\alpha > c+\epsilon)\mathbf{v}_\alpha^\top$ where $\{\mathbf{w}_\alpha \in \mathbb{R}^d\}$ is an arbitrary set of orthonormal vectors, with $D = \min(d, d_X + d_Y)$.

Proof. Let the SVD of $\mathbf{Z}$ be $\mathbf{W}\mathbf{\Lambda}_\mathbf{Z}\mathbf{V}_\mathbf{Z}^\top$. Minimizing over $\mathbf{Z}$ amounts to finding the matrices $\mathbf{W}$, $\mathbf{\Lambda}_\mathbf{Z}$, and $\mathbf{V}_\mathbf{Z}$ that give the smallest value of the objective. Because of the constraint $\mathbf{Z}\mathbf{Z}^\top \preccurlyeq I_d$, $\mathbf{\Lambda}_\mathbf{Z}$ satisfies $0 \leq \lambda_{\mathbf{Z}\alpha} \leq 1$. Plugging this form into the optimization objective we have

$$\operatorname{Tr}\left[(c+\epsilon)\mathbf{Z}^\top\mathbf{Z} - \mathbf{Z}^\top\mathbf{Z}\big[\epsilon\mathbf{X}^\top(\mathbf{X}\mathbf{X}^\top)^{-1}\mathbf{X} + \mathbf{Y}^\top\mathbf{Y}\big]\right] = \operatorname{Tr}\left[(c+\epsilon)\mathbf{\Lambda}_\mathbf{Z}^\top\mathbf{\Lambda}_\mathbf{Z} - \mathbf{\Lambda}_\mathbf{Z}^\top\mathbf{\Lambda}_\mathbf{Z}\tilde{\mathbf{V}}^\top\mathbf{\Lambda}^\top\mathbf{\Lambda}\tilde{\mathbf{V}}\right]$$

where $\tilde{\mathbf{V}} = \mathbf{V}^\top\mathbf{V}_\mathbf{Z}$. Note that $\mathbf{W}$ does not appear in this expression and therefore remains undetermined in this optimization objective.

$\mathbf{V}$ only appears in the second term and hence can be found by optimizing the second term alone. This term is of the form $-\sum_{\alpha,\beta}\lambda_{\mathbf{Z}\alpha}^2|\tilde{V}_{\alpha\beta}|^2\lambda_\beta^2$. The $\tilde{\mathbf{V}}$ that minimizes the term is the one that pairs λ_β and $\lambda_{\mathbf{Z}\alpha}$, in the sorted order. Assuming $(\lambda_{\mathbf{Z}\alpha})$ are also sorted in the decreasing order, this is achieved by setting $\mathbf{V}_\mathbf{Z} = \mathbf{V}$.

Plugging this in, the entire expression simplifies to

$$-\sum_\alpha(\lambda_\alpha - c - \epsilon)\lambda_{\mathbf{Z}\alpha}.$$

Since $0 \leq \lambda_{\mathbf{Z}\alpha} \leq 1$, it is clear that, for a minimum, if $(\lambda_\alpha - c - \epsilon) < 0$, $\lambda_{\mathbf{Z}\alpha} = 0$, while $(\lambda_\alpha - c - \epsilon) > 0$ implies $\lambda_{\mathbf{Z}\alpha} = 1$. In case of equality, the choice of $\lambda_{\mathbf{Z}\alpha}$ does not matter. Hence $\lambda_{\mathbf{Z}\alpha} = \mathbb{1}(\lambda_\alpha > c+\epsilon)$ is a valid minimum. The vectors $\{\mathbf{w}_\alpha\}$ are the columns of $\mathbf{W}$, while $\{\mathbf{v}_\alpha\}$ are the columns of $\mathbf{V}$. Hence, the result. □

From this theorem, we derive two corollaries, describing the different limits of $\epsilon \to 0$ and $\epsilon \to \infty$. The proofs of these corollaries follow directly from Theorem 1 which maps the within-layer computation of CCPC onto a PCA problem, and the known properties of PCA.

Corollary 2. *If the input sample covariance matrix, $\mathbf{C}_x$, is full rank, the internal representation $\mathbf{Z}$ in the limit $\epsilon \to 0$ is driven by the principal components (scores) of $\mathbf{Y}$, i.e.,*

$$\mathbf{Z}^* = \mathbf{W}_a^\top\mathbf{Y} + \epsilon\mathbf{W}_b\mathbf{X}, \tag{29}$$

with $\mathbf{W}_a$ the top principal directions of $\mathbf{Y}$ and $\mathbf{W}_b$ the maximally correlated directions between $\mathbf{X}$ and the columns of $\mathbf{W}_a$.

Corollary 3. *If the input sample covariance matrix, $\mathbf{C}_x$, is full rank, the internal representation $\mathbf{Z}$ in the limit $\epsilon \to \infty$ is driven by the maximally correlated directions between $\mathbf{C}_x^{-1/2}\mathbf{X}$ and $\mathbf{Y}$, i.e.,*

$$\mathbf{Z}^* = \mathbf{W}_a^\top \mathbf{Y} + \mathbf{W}_b \mathbf{X}, \tag{30}$$

with $\mathbf{W}_a, \mathbf{W}_b$ projecting onto a common lower-dimensional subspace so that the projections are maximally correlated.

B.3 Rank structure of CCPC

In this section again we set $c^{(l)} = 0$ and $\rho^{(l)} = 1$, and assume $g_a = 1, g_b = \epsilon$ is the same for all layers. The generalization to other cases is straightforward. We demonstrate a series of rank constraints on the intermediate layers of CCPC. The first of these is very general and requires no assumptions. The following statements become gradually more tight but also require more assumptions.

In this section, we define r_x and r_y to be the ranks of the the covariances of $\mathbf{X}$ and $\mathbf{Y}$, and $r^{(l)}$ to be the rank of the covariance of $\mathbf{Z}^{(l)}$ at the equilibrium of the CCPC update equations. We take $r^{(0)} = r_x$ and $r^{(n)} = r_y$.

Corollary 4. *The rank of the lth layer is bounded by $r^{(l+1)} + r^{(l-1)}$:*

$$\forall l \ : \ r^{(l)} \leq r^{(l+1)} + r^{(l-1)}. \tag{31}$$

Proof. This follows directly from Theorem 1 which describes $Z^{(l)}$ as the whitened PCA of the concatenation of $\mathbf{Z}^{(l-1)}$ and $\mathbf{Z}^{(l+1)}$. □

When the adjacent layers are whitened, i.e., the non-zero eigenvalues of $Z^{(l+1)}Z^{(l+1)\top}$ and $Z^{(l-1)}Z^{(l-1)\top}$ are all one, we can prove a tighter form of the rank constraint. This is relevant for the intermediate layers of CCPC other than the final hidden layer. This proposition is relevant for the final hidden layer only if the supervisory input $\mathbf{Y}$ is white.

Proposition 3. *Let $r_c^{(l)}$ be the rank of the covariance matrix $\mathbf{Z}^{(l-1)}\mathbf{Z}^{(l+1)\top}$. The rank of intermediate layers $l \leq n-2$ is given by:*

$$\forall l \leq n-2 \ : \ r^{(l)} \leq \begin{cases} r_c^{(l)} \leq \min\{r^{(l+1)}, r^{(l-1)}\} & \epsilon \geq 1, \\ r^{(l+1)} & \epsilon < 1. \end{cases} \tag{32}$$

Proof. This is a slight generalization of [71]. It follows from the eigenvalue structure of the concatenated data matrix given in Theorem 1, where only a subset of the eigenvalues exceed the threshold given by ϵ. The number of eigenvalues that exceed this threshold is given by Eq. (32). □

Applying this proposition to all the layers of a CCPC network, we arrive at the following corollary:

Corollary 5. *Assume that $\mathbf{Y}$ is white and $\epsilon \geq 1$. In a CCPC network with $c^{(l)} \geq 0$, $\rho^{(l)} = 1$, and layer widths $w^{(l)}$, the rank of the latent variables $\mathbf{z}^{(l)}$ is less than or equal to $\min\{r_x, r_y, w^{(1)}, \cdots, , w^{(n-1)}\}$.*

In other words, the rank of the latent variables is bounded by the minimum of the input/output ranks and the bottleneck of the network.

B.4 Alignment and relation to CCA

In the limit where the internal loss terms in Eq. (6) are zero, we effectively have a deep linear network [76]. A lot is known about deep linear networks, trained by gradient descent of weights. One of the remarkable properties of such networks is that the weights remain balanced, namely $\mathbf{W}^{(l)\top}\mathbf{W}^{(l)} - \mathbf{W}^{(l-1)}\mathbf{W}^{(l-1)\top}$ remains a constant, under gradient descent. In the case this difference is zero, the left singular vectors of $\mathbf{W}^{(l-1)}$ match the right singular vectors of $\mathbf{W}^{(l)}$ [80]. In addition, for two class linear discriminants, $\mathbf{W}^{(l-1)}$ ultimately become rank 1, under gradient descent [80].

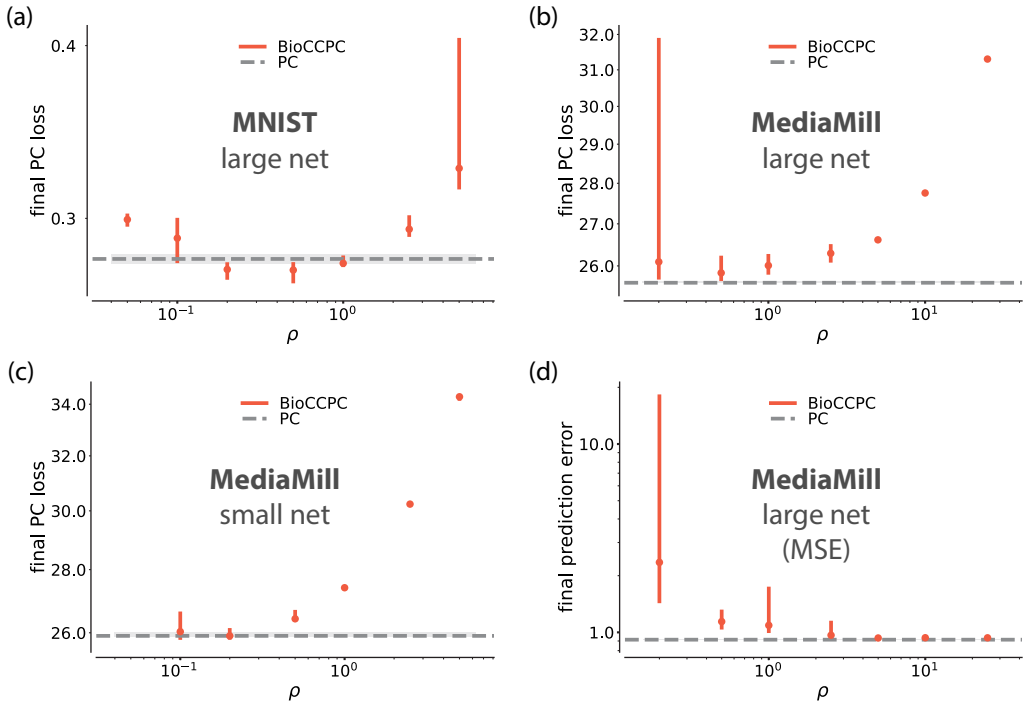

Figure 4: Dependence of BioCCPC performance on constraint scale. Unless otherwise noted, this is the predictive-coding loss (Eq. 6) after 500,000 iterations. (a) A large network with two hidden layers of sizes 50 and 5 trained on MNIST. (b) The same large network as in (a) run on the Mediamill dataset. See text. (c) A small net (single hidden layer of size 5) on Mediamill. (d) Mean-squared prediction error for the large net on Mediamill, showing that the best prediction performance may occur at a different choice of ρ compared to the lowest predictive-coding loss.

C Numerical simulations

C.1 Effect of constraint scale

The BioCCPC network uses an inequality constraint on the covariance of the hidden variables, $\frac{1}{T}\mathbf{Z}^{(l)}Z^{(l)\top} \preceq \rho^{(l)}\mathbf{I}$, and this constraint is saturated at convergence.[7] The scale, $\rho^{(l)}$, is therefore important: a scale that is too small or too big forces the hidden-layer activities into a range that can hinder learning, as seen in Fig. 4a. Interestingly, the BioCCPC algorithm with a well-chosen constraint scale converges faster than a plain PC network, that is, it can reach a lower PC loss given the same number of training samples (Fig. 4a).

The predictive-coding loss, Eq. (6), penalizes prediction errors in all layers. If we think about the task as supervised learning, however, it might make more sense to focus only on the accuracy of the prediction in the last layer. The best output prediction error may require a different constraint scale compared to the lowest PC loss, cf. Fig. 4b and Fig. 4d.

C.2 Effect of dataset choice

While we derived our algorithm by starting from the supervised PC framework of [4], the same derivation would hold for the unsupervised or self-supervised PC networks. In these settings, PC can be thought of as a framework which combines information from the inputs at the first and last layer, rather than a method for predicting an output from an input. In the linear case, this has much in common with biologically plausible circuits that perform canonical correlation

[7] More precisely, the constraint is saturated only within the subspace that $\mathbf{Z}^{(l)}$ occupies. In general, this is only the case in the layer with the smallest dimension, as the activity in the other layers is restricted to a subspace of dimension given by the smallest layer; see Corollary 5.

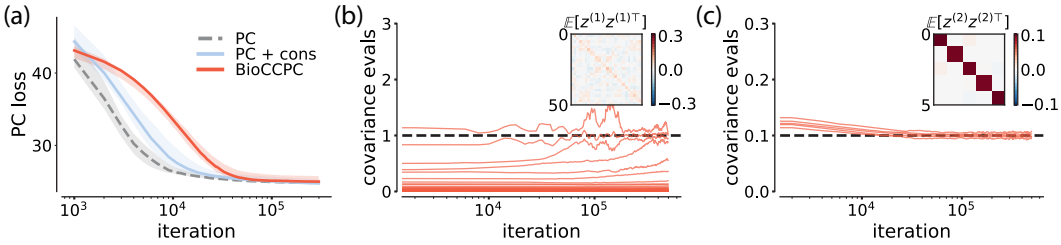

Figure 5: Comparison of our algorithm with PC on the Mediamill dataset. Simulations run using a network with two hidden layers, sizes 50 and 5. Shaded area represents the variation over 10 runs. (a) Evolution of the predictive-coding loss, Eq. (6), on a validation set during learning. (b) and (c) Evolution of the eigenvalues of the covariance matrix of hidden-layer activations in the two hidden layers. The dashed lines show the scale of the constraint; see SM Sec. A. Inset: covariance matrix at end of training. Note how activity is whitened in the second hidden layer (c), and low-rank in the first hidden layer (b). Note also that the network performance is good even though the constraint is not yet fully converged.

analysis (CCA) [55, 71]. We therefore tested the algorithm on Mediamill [81], a dataset consisting of two views of the same scene, one given by the video and the other by text annotation. The results (Figs. 4b-d and Fig. 5) show similar trends as those obtained on MNIST. In particular, an appropriate choice of constraint scale leads to results that are competitive with less biologically plausible implementations of PC.

In Fig. 6 we show the evolution of the validation-set loss for four other datasets: FashionMNIST, CIFAR10, CIFAR100, and the LFW (Labeled Faces in the Wild). The first three are supervised-learning scenarios and we use a network similar to the one we employed for MNIST. The LFW dataset contains pairs of views of the same subject, and so we employ a symmetric autoencoder-like architecture for this self-supervised task. As before, we see that the BioCCPC algorithm converges to a solution that has predictive-coding loss comparable to the minimum obtained from traditional PC implementations (see Fig. 6).

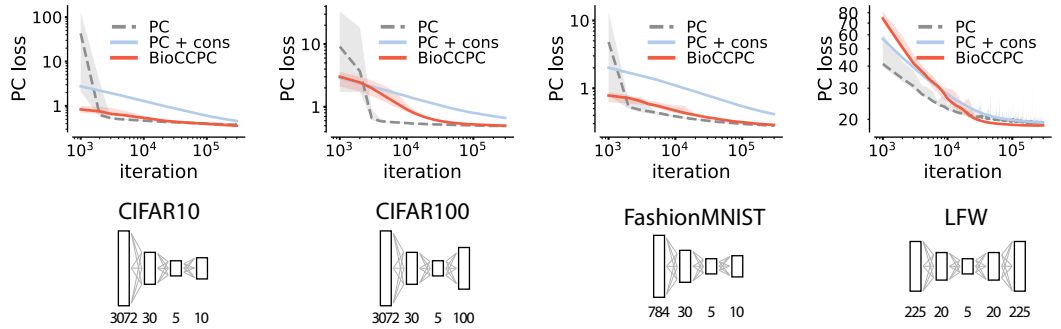

Figure 6: Comparison of our algorithm with PC on a variety of datasets. The name of the dataset and the architecture we employed are shown below each figure. Shaded area represents the variation over 10 runs.

C.3 Network architecture

We tested BioCCPC across different architectures, such as a shallow network with a small hidden-layer dimension (Fig. 4) and a deeper network with two hidden layers, one of which is large-dimensional (Fig. 4a,b,d). In both cases BioCCPC compares well to PC.

C.4 Lack of symmetry in feedforward and feedback weights

One of the difficulties with a biological implementation of predictive coding is that it requires symmetric weights across layers. This is not so much an issue of plausibility—indeed, symmetric weights can be obtained simply by using the same Hebbian rules to train both feedforward and feedback connections—but rather a problem with realism: weights in many areas of the brain are not in fact seen to be symmetric.

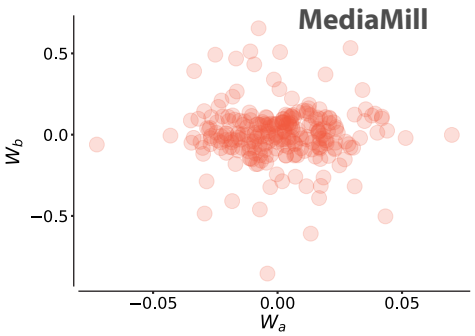

Figure 7: Comparison of feedforward and feedback weights after learning on a network with hidden layers of sizes 50 and 5 trained on the Mediamill dataset.

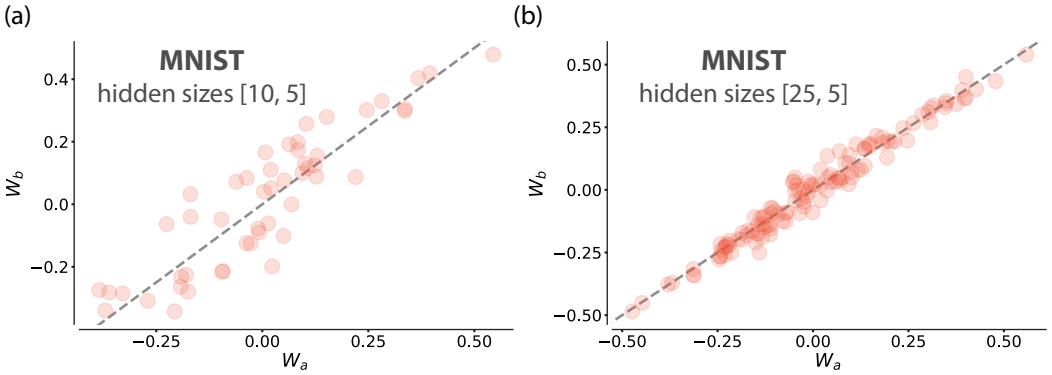

Figure 8: Comparison of feedforward and feedback weights after learning for different network sizes. This is trained on MNIST. (a) There is less symmetry in a smaller network with hidden layers of sizes 10 and 5. (b) There is more symmetry in a larger network, with hidden layers 25 and 5.

Our algorithm explicitly breaks the symmetry between the weights, leading to solutions that generically alleviate this difficulty. Indeed, in some cases there is very little correlation between feedforward and feedback weights; see Fig. 7. However, in some cases we see that the forward and backward weights converge to similar values even though the learning rules are not symmetric. We suspect that this can occur in cases where there is a large imbalance between the width of adjacent layers, the larger the imbalance (i.e. the lower rank the wider layer is) the more correlation we see in our experiments; see Fig. 8. In biologically relevant neural networks, we do not generally see this large imbalance in the width of adjacent layers and therefore we do not expect to see high correlation between the forward and backward weights.

C.5 Predictive coding with constraint

In the main text we considered a predictive-coding network that imposes a covariance constrained but we did not make explicit what this entails. Explicitly, we augmented the Whittington-Bogacz objective, Eq. (2), with a Lagrange multiplier enforcing an inequality constraint on the covariance of hidden variables, similar to what we do in BioCCPC:

$$L_{\text{cons}} = \frac{1}{2}\sum_{l} \frac{\left\|\mathbf{Z}^{(l)} - \mathbf{W}^{(l-1)}\mathbf{Z}^{(l-1)}\right\|_F^2}{\sigma^{(l)2}} + \operatorname{Tr}\mathbf{Q}^{(l)\top}\mathbf{Q}^{(l)}(\mathbf{Z}^{(l)}\mathbf{Z}^{(l)\top} - T\rho^{(l)}\mathbf{I}), \tag{33}$$

and optimized using stochastic gradient descent. This lets us distinguish the effect of the constraint from the effect of the approximations we used in the derivation.

C.6 Details of simulation runs

We initialized our the weights in our networks using uniform random numbers in the range $[-a, a]$, where $a = 1/\sqrt{\text{number of columns}}$. This is similar to Xavier initialization [82] except the scale of initial weights depends only on the number of columns instead of both rows and columns. We have found this initialization to provide slightly better results.

We optimized hyperparameters using Optuna [83] with the tree-structured Parzen Estimator (TPE) algorithm [84]. We ran the optimization for 100 trials and repeated it 10 times with different random seeds for every combination of dataset, network architecture, and constraint scale ρ (with the exception of the plain PC algorithm which does not have a constraint). In each case we optimized the learning rate η used in the weight updates and the learning rate τ^{-1} of the fast dynamics (see Algorithm 1). For networks that have a constraint, we allowed a different learning rate $f\eta$ for the $\mathbf{Q}$ dynamics, and optimized for the factor f. Finally, for BioCCPC network, we allowed for different learning rates for the forward and backward weights, $\mathbf{W}_b$ and $\mathbf{W}_a$, and optimized for the ratio between these learning rates. Hyperparameter optimization runs were trained for 500 batches with batch size 100 and were evaluated on a held-out validation set comprising 500 samples. Each run was repeated 4 times with different random initialization to account for stochasticity in the learning dynamics.

The simulation runs that were used to make the figures in the paper were based on the hyperparameter optimization results that yielded the lowest predictive-coding loss. Since hyperparameter optimization tends to push the learning rate up to the brink of instability, the learning rate used in the simulations was lowered by multiplying by a certain "safety" factor α. We generally used $\alpha = 0.8$.

We ran our simulations on the CPU. Each run took from 1 to 4 minutes when run single-threaded on a 24-core Intel Xeon CPU E5-2643 at 3.40GHz running Linux. We also used a cluster to run the hyperparameter optimization for different architectures in parallel.

D Generalization error

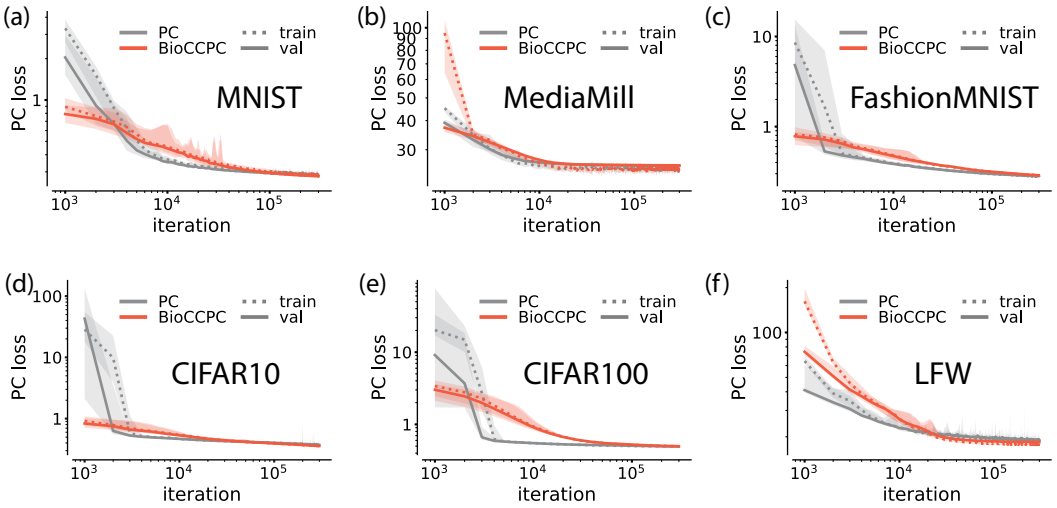

Figure 9: There is little overfitting in the linear setting that we are investigating. The plots compare the training- and validation-set predictive-coding losses for both the less biologically plausible predictive-coding implementation from [4] (PC) and for our algorithm (BioCCPC).

The linear constraint that we impose on our network is a strong regularizer. Figure 9 compares the training and validation losses during training on six different datasets (as described above). We see that there is very little overfitting in all cases.

E BioCCPC Pseudocode

Algorithm 1: Covariance Constrained predictive coding algorithm (BioCCPC)

input: $\mathbf{x}_t$, $\mathbf{y}_t$ ▷ new sample and previous weight matrices

$\mathbf{W}_a^{(l)}, \mathbf{W}_b^{(l-1)}, \mathbf{Q}^{(l)} \qquad \forall l \in \{1, \ldots, n-1\}$

$\mathbf{z}_t^{(0)} \leftarrow \mathbf{x}_t,\ \mathbf{z}_t^{(l)} \leftarrow \mathbf{W}_b^{(l-1)}\mathbf{z}_t^{(l-1)}$ ▷ initialize latents via forward pass

run until convergence for all $l \in \{1, n-1\}$ ▷ neural dynamics

$$\mathbf{n}^{(l)} \leftarrow (1/g_a^{(l)})\mathbf{Q}^{(l)}\mathbf{z}_t^{(l)}$$

$$\mathbf{v}_a^{(l)} \leftarrow \mathbf{W}_a^{(l)\top}\mathbf{z}_t^{(l+1)} - \mathbf{Q}^{(l)\top}\mathbf{n}^{(l)} \quad , \qquad \mathbf{v}_b^{(l)} \leftarrow \mathbf{W}_b^{(l-1)}\mathbf{z}_t^{(l-1)}$$

$$\mathbf{z}_t^{(l)} \leftarrow \mathbf{z}_t^{(l)} + \tau^{-1}\left[-g_{\mathrm{lk}}^{(l)}\mathbf{z}_t^{(l)} + g_a^{(l)}(\mathbf{v}_a^{(l)} - \mathbf{z}_t^{(l)}) + g_b^{(l)}(\mathbf{v}_b^{(l)} - \mathbf{z}_t^{(l)})\right]$$

update for all $l \in \{1, n-1\}$ ▷ synaptic weight updates

$$\mathbf{W}_a^{(l)} \leftarrow \mathbf{W}_a^{(l)} + \eta\big(\mathbf{z}_t^{(l+1)}\mathbf{z}_t^{(l)\top} - \mathbf{W}_a^{(l)}\big)$$

$$\mathbf{W}_b^{(l-1)} \leftarrow \mathbf{W}_b^{(l-1)} + \eta\Big[g_a^{(l)}\big(\mathbf{W}_a^{(l)\top}\mathbf{z}_t^{(l+1)} - \mathbf{Q}^{(l)\top}\mathbf{n}^{(l)}\big) - c^{(l)}\mathbf{z}_t^{(l)}\Big]\mathbf{z}_t^{(l-1)\top}$$

$$\mathbf{Q}^{(l)} \leftarrow \mathbf{Q}^{(l)} + \eta\big(\mathbf{n}^{(l)}\mathbf{z}_t^{(l)\top} - \mathbf{Q}^{(l)}\big)$$

output: $\mathbf{z}_t^{(l)} \qquad \forall l \in \{0, \ldots, n\}$ ▷ internal representations and updated weights

$\mathbf{W}_a^{(l)}, \mathbf{W}_b^{(l-1)}, \mathbf{Q}^{(l)} \qquad \forall l \in \{1, \ldots, n-1\}$

Supplementary References

[80] Ziwei Ji and Matus Telgarsky. Gradient descent aligns the layers of deep linear networks. In *7th International Conference on Learning Representations, ICLR 2019*, 2019.

[81] Cees GM Snoek, Marcel Worring, Jan C Van Gemert, Jan-Mark Geusebroek, and Arnold WM Smeulders. The challenge problem for automated detection of 101 semantic concepts in multimedia. In *Proceedings of the 14th ACM international conference on Multimedia*, pages 421–430, 2006.

[82] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256. JMLR Workshop and Conference Proceedings, 2010.

[83] Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2019.

[84] James Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. Algorithms for hyper-parameter optimization. *Advances in neural information processing systems*, 24, 2011.