LETTER Communicated by Blake Richards

Multiple Timescale Online Learning Rules for Information Maximization with Energetic Constraints

Peng Yi
yipeng@wustl.edu
ShiNung Ching
shinung@wustl.edu
Department of Electrical and Systems Engineering, Washington University in St. Louis, St. Louis, MO, 63130, U.S.A.

A key aspect of the neural coding problem is understanding how representations of afferent stimuli are built through the dynamics of learning and adaptation within neural networks. The infomax paradigm is built on the premise that such learning attempts to maximize the mutual information between input stimuli and neural activities. In this letter, we tackle the problem of such information-based neural coding with an eye toward two conceptual hurdles. Specifically, we examine and then show how this form of coding can be achieved with online input processing. Our framework thus obviates the biological incompatibility of optimization methods that rely on global network awareness and batch processing of sensory signals. Central to our result is the use of variational bounds as a surrogate objective function, an established technique that has not previously been shown to yield online policies. We obtain learning dynamics for both linear-continuous and discrete spiking neural encoding models under the umbrella of linear gaussian decoders. This result is enabled by approximating certain information quantities in terms of neuronal activity via pairwise feedback mechanisms. Furthermore, we tackle the problem of how such learning dynamics can be realized with strict energetic constraints. We show that endowing networks with auxiliary variables that evolve on a slower timescale can allow for the realization of saddle-point optimization within the neural dynamics, leading to neural codes with favorable properties in terms of both information and energy.

1 Introduction

In computational neuroscience, a fundamental concern is to deduce how neurons represent sensory stimuli via their responses. This coding problem has been examined for over a half-century (Barlow, 1961; Attneave, 1954) and is still popular in neuroscience (Brette, 2017). To understand the mechanisms of neural coding, many normative theories have been constructed that start from an optimality hypothesis and criterion, then

Neural Computation **31, 943–979** (2019)
doi:10.1162/neco_a_01182

predict how neural coding should be enacted. One well-accepted hypothesis, infomax (Linsker, 1988, 1989; Park & Pillow, 2017), posits that the neural system adapts its encoding model by maximizing the information transmission—mutual information (MI), between stimuli and the ensuing neural response. Infomax has been widely adopted to explain various neural experimental data or make quantitative predictions of neural coding mechanisms, including population coding (Tkačik, Prentice, Balasubramanian, & Schneidman, 2010), spatial-temporal correlation coding (Pillow et al., 2008), redundancy-reduction (decorrelation) coding (Pitkow & Meister, 2012), and on-off pathway splitting (Gjorgjieva, Sompolinsky, & Meister, 2014). Furthermore, infomax-based learning has been proposed as a self-organization principle for sensory neural systems (Linsker 1988, 1989) and has been adopted as an objective function in feature learning, notably for independent component analysis (Bell & Sejnowski, 1995). In this letter, we build on the infomax paradigm by showing how optimal neural codes can be learned through pairwise interactions between units by processing streaming input while also satisfying strict energetic constraints.

1.1 Challenges of Infomax Learning. From a computational perspective, optimizing mutual information (MI) is intractable for all but special cases, such as gaussian or finite discrete-state stimuli. This is because computing MI requires marginalization of the stimulus distribution, a cumbersome procedure for high-dimensional data with complicated distributions. Moreover, the exact stimulus distribution is not known in advance, since sensory inputs are encountered in an online fashion. Thus, enacting infomax in a neural circuit presents several challenges: (1) representations should be obtained online, with only sample-by-sample processing of stimuli, (2) learning (e.g., via synaptic plasticity) should rely on primarily local (pairwise) neural interactions, and (3) learning and the ensuing obtained representation should not be energetically wasteful.

1.2 Approximation Methods for Infomax. The computational challenges are incompletely addressed by current infomax learning frameworks. With an as-if gaussian approximation, Linsker (1992) proposed local learning rules for the case of a linear encoder and gaussian stimuli. Brunel and Nadal (1998) proposed approximating the mutual information with Fisher information, an approximation that has recently been refined by Huang and Zhang (2018). A similar Fisher information approximation was also adopted in Wei and Stocker (2016) and Karklin and Simoncelli (2011). However, the learning rules based on gaussian approximation or Fisher information are batched algorithms since the full stimulus data are needed in computation. Online learning rules for information preservation have been proposed for recurrent networks (Tanaka, Kaneko, & Aoyagi, 2009; Liu & Ching, 2017) when both the stimuli and the neural response are

discrete state variables, but these cannot be trivially extended to the case when stimuli are continuous.

1.3 Infomax with Energetic Constraints. The issue of energetic efficiency is less well studied in the context of infomax, despite the intuitive premise that neural circuits balance computational efficiency with energy costs (Levy & Baxter, 1996; Laughlin, 2001; Gershman, Horvitz, & Tenenbaum, 2015). Results in this domain include Wang, Wei, Stocker, and Lee (2016), which derived infomax-based efficient neural coding with a hard, energetic constraint for a two-neuron case with an analytic method. Further, Karklin and Simoncelli (2011) added energy regularization and performed computations using gaussian approximation and batched gradient algorithms. However, whether infomax neural coding with hard energy constraints is achievable through online learning is not known.

1.4 Contributions. In this work, we address the online learning of infomax neural coding through pairwise neural interactions by combining a variational mutual information approximation and a sampled-based gradient approximation with neural activities. We adopt the variational lower bound of MI proposed in Barber and Agakov (2003) and propose a gradient-based online learning scheme with streaming stimuli samples in section 2. The variational MI is introduced by using a variational decoder distribution as an approximation of the true Bayesian posterior. The variational decoder can be used for probabilistic reconstruction of stimuli given the encoder neural responses and, hence, models how downstream neurons use the coded sensory information. However, the gradient of the variational MI may still contain nonlocal terms (i.e., overall synaptic weights). To overcome this obstacle, we replace nonlocal terms with sample-based approximations derived from encoder and decoder neurons' activities, such that the derived learning rules are only based on pairwise neural interactions. With the gradient-based online learning scheme and a gaussian linear decoder, we obtain a learning rule for a rate-based linear gaussian encoder in section 3. In section 4, we derive a multiple timescale online learning rule for infomax when an energetic constraint is imposed on the response of the linear gaussian encoder. We obtain the learning rule for a discrete-state binary spiking encoder in section 5. The efficiency of the learning rules is demonstrated through several numerical examples.

2 Neural Coding with Infomax and Online Variational Learning Scheme

In this section, we introduce the problem of efficient neural coding with infomax and then present our online learning scheme based on variational methods.

2.1 Efficient Neural Coding with Infomax. Consider a stimulus $x \in \mathbb{R}^N$ from a real-valued high-dimensional space $\mathbb{R}^N$ following an unknown distribution $p(x)$. The neural system response r to this stimulus, which is here assumed to be a continuous random variable, is modeled with a constrained conditional probability density function $p(r|x;\theta)$. That is, $p(r|x;\theta)$, known as the neural stimuli-response function or encoder model, comes from a specified family of distributions with the encoding parameter θ.

Infomax neural coding hypothesizes that the neural system finds the parameter θ by optimizing the information transmission between the stimuli x and the response r,

$$\max_{\theta} I(x,r),\ \ I(x,r) = H(x) - H(x|r) = KL(p(x,r),\, p(x)p(r)), \tag{2.1}$$

where

$$H(x) = -\int_x dx p(x) \log p(x), \tag{2.2}$$

$$H(x|r) = -\int_r dr p(r) \int_x dx p(x|r) \log p(x|r), \tag{2.3}$$

$$KL(p(x,r),\, p(x)p(r)) = \int_x \int_r dx dr p(x,r) \log \frac{p(x,r)}{p(x)p(r)}. \tag{2.4}$$

Here, $H(x)$ is the entropy of x, $H(x|r)$ is the conditional entropy of the posterior, and $KL(p(x,r),\, p(x)p(r))$ is the Kullback–Leibler divergence between $p(x,r)$ and $p(x)p(r)$. Notice that

$$p(r) = \int_x dx p(x,r) = \int_x dx p(x) p(r|x;\theta) \tag{2.5}$$

and

$$p(x|r) = \frac{p(r|x;\theta)p(x)}{p(r)}. \tag{2.6}$$

Since $p(x|r)$ is the posterior of the stimulus given the neuron response, neural coding with infomax falls within the Bayesian brain premise that the brain forms perception and cognition in a Bayes optimal manner (Park & Pillow, 2017; Doya, 2007).

In the previous equations, the probability density function $p(r|x;\theta)$ can be changed to the probability mass function $P(r|x;\theta)$ when the neural response r is a discrete-state random variable and the corresponding integral becomes a sum. We will use this modification in section 5, where a discrete neural response model is adopted.

A key challenge in handling equation 2.1 comes from the computation of the posterior in equation 2.6, which is needed in equation 2.3. Moreover, the distribution $p(x)$ is not known in advance by the encoder neurons. Even if $p(x)$ could be approximated by its empirical distribution (which inevitably leads to batched algorithms), marginalization over x in equation 2.3 is still intractable when x is high dimensional, and there is still no closed form of equation 2.3. To overcome the obstacle, we introduce our online learning scheme by variational methods.

2.2 Online Learning Scheme with Variational Mutual Information. Given any distribution $q(x|r)$,

$$\begin{aligned} &KL(p(x|r), q(x|r)) = \int_x dx p(x|r) \log \tfrac{p(x|r)}{q(x|r)} \\ &= \int_x dx p(x|r) \log p(x|r) - \int_x dx p(x|r) \log q(x|r) \geq 0. \end{aligned} \tag{2.7}$$

Therefore,

$$\begin{aligned} &-H(x|r) = \int_r dr p(r) \int_x dx p(x|r) \log p(x|r) \\ &\geq \int_r dr p(r) \int_x dx p(x|r) \log q(x|r) = \int_{x,r} dx dr p(x, r) \log q(x|r). \end{aligned} \tag{2.8}$$

Therefore, given the encoder model $p(r|x; \theta)$ parameterized with θ, and a family of variational distributions $q(x|r; \phi)$ parameterized with ϕ, we have the following variational lower bound of MI:

$$I(x, r) \geq H(x) + \int_x dx p(x) \int_r dr p(r|x; \theta) \log q(x|r; \phi) \triangleq \tilde{I}(x, r). \tag{2.9}$$

Instead of optimizing the exact $I(x, r)$ defined in equation 2.1, we optimize $\tilde{I}(x, r)$ in equation 2.9. Notice that $H(x)$ is independent of parameters θ, ϕ; hence, infomax neural coding is solved by the following optimization problem:

$$\max_{\theta \in \Theta, \phi \in \Phi} \int_x dx p(x) \int_r dr p(r|x; \theta) \log q(x|r; \phi). \tag{2.10}$$

The lower bound in equation 2.9 is called the variational mutual information (VMI), first proposed in Barber and Agakov (2003). Here, $q(x|r; \phi)$ is a variational approximation for $p(x|r)$ in equation 2.6. Since $p(r|x; \theta)$ transforms stimuli to neural representation, it is called an encoder; similarly, $q(x|r; \phi)$ is called a decoder. The constraint $\theta \in \Theta, \phi \in \Phi$ is introduced because $p(r|x; \theta)$ and $q(x|r; \phi)$ are usually limited within specific families of distributions.

The variational method has been shown to be an efficient computational method when Bayesian inference is involved as a subroutine and has been

Algorithm 1: Online Infomax Learning Scheme.

Data: Samples of stimuli $p(x)$, learning rate η_k

Result: Encoder parameter θ and decoder parameter ϕ

initialization;

while *A stimulus* $x_k \in \mathbb{R}^N$ *is sampled from* $p(x)$ **do**

- Given the decoder parameter ϕ, the encoder computes an approximation of $\nabla_\theta \int_r p(r|x_k;\theta)\log q(x_k|r;\phi)dr$ denoted as $\Delta\theta$, and updates θ as

$$\theta \leftarrow \mathcal{P}_\Theta(\theta + \eta_k \Delta\theta). \tag{2.11}$$

- Given the encoder parameter θ, the decoder computes an approximation of $\nabla_\phi \int_r p(r|x_k;\theta)\log q(x_k|r;\phi)dr$ denoted as $\Delta\phi$, and update ϕ as

$$\phi \leftarrow \mathcal{P}_\Phi(\phi + \eta_k \Delta\phi). \tag{2.12}$$

end

verified recently in deep learning applications (Kingma & Max, 2013; Shakir & Rezende, 2015; Chalk, Marre, & Tkacik, 2018; Rezende, Wierstra, & Gerstner, 2011). If $q(x|r;\phi)$ is endowed with condition $\forall\theta, \exists\phi, q(x|r;\phi) = p(x|r)$, then the variational bound $\tilde{I}(x,r)$ is exactly the same as $I(x,r)$, in contrast to a Fisher information approximation, which is only exact asymptotically(Brunel & Nadal, 1998; Huang & Zhang, 2018).

Although algorithms have been developed for VMI optimization, most of them are nonlocal or batched. For example, Barber and Agakov (2003) maximized the lower bound $\tilde{I}(x,r)$ by alternatively optimizing $p(r|x;\theta)$ and $q(x|r;\phi)$ with respect to θ and ϕ when a batch of stimulus data is given. Similar batched algorithms are adopted in Chalk et al. (2018) and Shakir and Rezende (2015). In contrast, we propose the following online variational learning scheme with streaming stimuli (see algorithm 1).

Here, $\mathcal{P}_\Omega(x)$ is the projection of x onto the set Ω: $\mathcal{P}_\Omega(x) = \arg\min_{y\in\Omega} ||x - y||$. From an optimization perspective, the proposed learning scheme can be viewed as a stochastic gradient algorithm. In fact, the objective function in equation 2.10 is

$$E_x\left[\int_r dr p(r|x;\theta)\log q(x|r;\phi)\right]. \tag{2.13}$$

Instead of optimizing with the exact gradient $\nabla E_x[\int_r dr p(r|x;\theta)\log q(x|r;\phi)]$, we can utilize a sample $x_k \sim p(x)$ to estimate the gradient

$$\nabla E_x[\int_r drp(r|x;\theta)\log q(x|r;\phi)] = E_x[\nabla \int_r drp(r|x;\theta)\log q(x|r;\phi)]$$
$$\approx \nabla \int_r drp(r|x_k;\theta)\log q(x_k|r;\phi). \tag{2.14}$$

Our scheme follows by updating θ and ϕ with the approximated gradient of

$$\nabla \int_r drp(r|x_k;\theta)\log q(x_k|r;\phi).$$

But if a batch of stimulus data $\{x_1,\ldots,x_{\mathbf{T}}\}$ are given as independent and identically distributed (i.i.d.) samples of the unknown distribution $p(x)$, then we can approximate $p(x)$ with the empirical distribution $p(x) = \frac{1}{\mathbf{T}}\sum_{k=1}^{\mathbf{T}}\delta(x-x_k)$. In this case, equation 2.10 becomes

$$\max_{\theta\in\Theta,\phi\in\Phi}\frac{1}{\mathbf{T}}\sum_{k=1}^{\mathbf{T}}\int_r drp(r|x_k;\theta)\log q(x_k|r;\phi). \tag{2.15}$$

Note that for an optimization problem with a sum of objective functions, $\min_{x\in\Omega}\sum_{i=1}^{N} f_i(x)$, instead of utilizing gradients of all $f_i(x)$ at each iteration, the incremental gradient method (Bertsekas, 2011) picks one f_i (denoted, f_{i_k}) and updates as $x_{k+1} = \mathcal{P}_\Omega(x_k - \eta_k\nabla f_{i_k}(x_k))$. Therefore, algorithm 1 can also be treated as an incremental gradient method for equation 2.15.

The proposed scheme is able to process streaming stimuli in an online fashion. However, we still seek a gradient computation or approximation in equations 2.11 and 2.12 that is implementable with pairwise interactions. To obtain such an approximation, we introduce the variational distribution or decoder, $q(x|r;\phi)$, that will be used. Decoder neurons take the encoder neuron response r as input and then provide a probabilistic "reconstruction" of input stimuli. In this letter, we use a linear gaussian decoder $q(x|r;\phi) = \mathcal{N}(Ur,\Lambda)$, even though the stimulus distribution and the encoder need not be gaussian. Suppose there are m encoder neurons; then $U = [u_1,\ldots,u_N]^T = [u_{ij}] \in \mathbb{R}^{N\times m}$ and $u_i = (u_{i1},\ldots,u_{im})^T \in \mathbb{R}^m$ is the synaptic weight from all encoder neurons to the ith decoder neuron. The decoder parameter ϕ corresponds to the synaptic weights $\{u_1,\ldots,u_N\}$. $\Lambda = diag\{\lambda_1,\ldots,\lambda_N\}$ is a diagonal matrix with positive diagonal elements $\lambda_i > 0$. Hence, the ith decoder neuron follows a gaussian distribution $q(x_i|r;\phi) = \mathcal{N}(u_i^T r,\lambda_i)$. We impose the following constraint on the decoder synaptic weight vector:

$$||u_i||_2 \le 1, i = 1,\ldots,N. \tag{2.16}$$

Further, we denote the unit norm ball in $\mathbb{R}^m$ as $\mathcal{B}_2^m = \{x\in\mathbb{R}^m|\ ||x||_2 \le 1\}$, and the constraint that $\phi\in\Phi$ is $u_i \in \mathcal{B}_2^m, i = 1,\ldots,N$.

Remark 1. In this work, we only consider an ℓ_2 constraint, equation 2.16, on the decoder parameter. This constraint implies that the overall synaptic weights of each decoder neuron should not be overly strong and the sum of squared incoming synaptic weights should be less than 1. The projection onto the unit norm ball, $P_{\mathcal{B}_2^m}$, required in equation 2.12 ($u_i \leftarrow \frac{u_i}{||u_i||_2}$ when $||u_i||_2 > 1$) can be performed in a local manner. In fact, the decoder neuron i can first calculate $||u_i||_2$ and send the signal back to each synapse u_{ij}, since it knows all the incoming synaptic weights u_{ij}, $j = 1, \ldots, m$. Sparsifying constraints on decoder parameters, such as $||u_i||_0 \leq l_0$, could similarly be considered.

3 Linear Gaussian Encoding Model

In this section, we give a local learning rule for the encoder based on the proposed online learning scheme and sample-based gradient approximation techniques.

Consider m sensory neurons whose responses are a real-valued vector $r = (r_1, \ldots, r_m)^T \in \mathbb{R}^m$. Here, the ith encoder neuron response r_i is treated as a deviation from baseline and thus can assume both positive and negative values. Given a stimulus $x \in \mathbb{R}^N$, the r follows a linear gaussian distribution $r \sim p(r|x;\theta) = \mathcal{N}(Wx, \Sigma)$ where $W = [w_1, \ldots, w_m]^T = [w_{ij}] \in \mathbb{R}^{m \times N}$ and $\Sigma = diag\{\sigma_1, \ldots, \sigma_N\}$. $w_i = (w_{i1}, \ldots, w_{iN})^T \in \mathbb{R}^N$ can be regarded as the synaptic weight of the ith encoder neuron and $\sigma_i > 0$ describes its encoding noise. Here, σ_i is assumed to be fixed, while W is the encoder parameter that should be learned via infomax. We also impose a norm constraint on the encoder synaptic weight:

$$w_i \in \mathcal{B}_2^N, i = 1, \ldots, N, \quad \mathcal{B}_2^N = \{x \in \mathbb{R}^N | \ ||x||_2 \leq 1\}. \tag{3.1}$$

Notice that even though the encoder is assumed to be gaussian, directly optimizing MI in equation 2.1 is still intractable since the stimulus distribution $p(x)$ is unknown to the neurons and in general nongaussian. In this scenario, we introduce a synaptic learning rule as follows.

The encoder-decoder structure and information flow of algorithm 2 is illustrated in Figure 1. Let us examine the learning rules 3.2 and 3.3 to show that they are indeed updated with pairwise information.

First, note that equation 3.2 can be regarded as a three-term Hebbian learning rule. Here, $(x_k - \hat{x}_k)^T$ is locally available to each encoder neuron since the reconstructed sample $\hat{x}_k$ is fed backward from the decoder neurons to the encoder neurons. Since u_{ji} is the synaptic weight from encoder i to decoder neuron j, the term $U_{:,i} = (u_{1i}, \ldots, u_{Ni})^T$ is a vector describing how the response of encoder i influences the decoder neurons, hence, could be treated as pairwise interacting information that is available to the ith

Algorithm 2: Sample-Based Infomax Learning for Linear Encoder and Decoder.

Data: Samples of underlying stimuli $p(x)$

Result: Synaptic weights of encoder and decoder, W, U

initialization;

while *A stimulus* $x_k \in \mathbb{R}^N$ *is sampled from* $p(x)$ **do**

- The stimulus signal x_k is forward processed by the encoder and decoder to get samples. First, the encoder gets a sample r_k according to its current encoding model, $r_k \sim \mathcal{N}(Wx_k, \Sigma)$. Then r_k is sent to the decoder, which gets a sample $\hat{x}_k$ with its decoding model, $\hat{x}_k \sim \mathcal{N}(Ur_k, \Lambda)$. The reconstructed sample $\hat{x}_k$ is compared with the stimulus x_k to give an error signal, $x_k - \hat{x}_k$. The error signal is fed backward such that it is available to both encoder and decoder neurons.

- Encoder neuron i adapts its synaptic weight w_i:

$$w_i \leftarrow \mathcal{P}_{\mathcal{B}_2^N}\left\{w_i + \eta_k[\left(\frac{u_{1i}}{\lambda_1}, \cdots, \frac{u_{Ni}}{\lambda_N}\right)(x_k - \hat{x}_k)]x_k\right\}. \tag{3.2}$$

- Decoder neuron j adapts its synaptic weight u_j:

$$u_j \leftarrow \mathcal{P}_{\mathcal{B}_2^m}\left\{u_j + \eta_k \frac{1}{\lambda_j}[(x_k - \hat{x}_k)_j r_k - (\sigma_1 u_{j1}, \cdots, \sigma_m u_{jm})^T]\right\}. \tag{3.3}$$

end

encoder neuron. It is important to note that even through these interactions are pairwise, the encoder rule itself is not fully local because it relies on knowledge of pre- and postsynaptic weights (i.e., the issue of weight transport). We elaborate on this further in section 3.2.

For the decoder synaptic weight learning rule, equation 3.3, the jth decoder neuron needs only the encoder sample r_k and the jth element of reconstruction error $(x_k - \hat{x}_k)_j$ (not the entire vector $x_k - \hat{x}_k$). Since $u_j = (u_{j1}, \ldots, u_{jm})^T$ is simply the synaptic weight of decoder j, the last term in equation 3.3 is locally computable. Moreover, since the variance parameters σ_i and λ_j are fixed, they can be locally held by each encoder or decoder neuron. Finally, the projection in equations 3.3 and 3.2 can be performed at the level of individual neurons with locally held synaptic information. A detailed derivation of learning rules 3.3 and 3.2 is given in the appendix.

3.1 Infomax Learning of MNIST Data Set. We illustrate the rules in algorithm 2 by showing that an infomax-based efficient representation can be learned when high-dimensional stimuli are compressed to low-dimensional encoder neuron responses. Here, we consider the MNIST

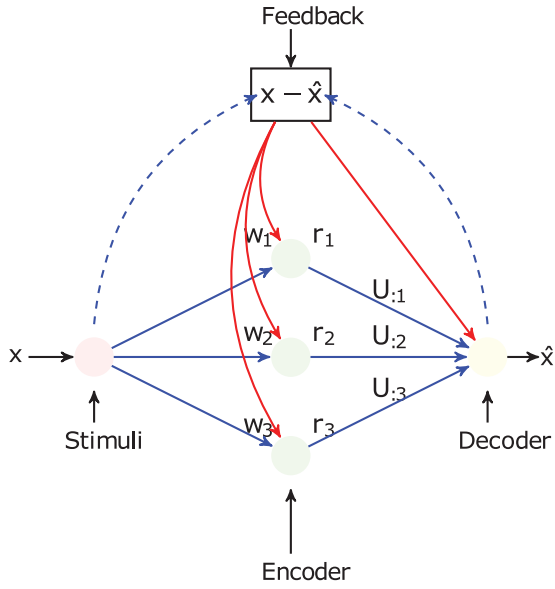

Figure 1: The encoder-decoder structure and information flow.

data set, where each sample is a 28×28 image of a handwritten digit. Hence, the stimulus x is a 784-dimensional real-valued vector: $N = 784$. We set the number of encoder neurons m to be 36. During learning, we sequentially present a randomly selected image (out of 6×10^4 images in the MNIST database) to algorithm 2. The gaussian noise parameters $\Sigma = diag\{\sigma_1, \ldots, \sigma_m\}$ and $\Lambda = diag\{\lambda_1, \ldots, \lambda_N\}$ are randomly drawn from [0.01, 0.02]. Since we have the constraints on synaptic weights $W = [w_1, \ldots, w_m]^T$, $U = [u_1, \ldots, u_N]^T$ that $w_i \in \mathcal{B}_2^N$, $i = 1, \ldots, m$ and $u_j \in \mathcal{B}_2^m$, $j = 1, \ldots, N$ and each pixel of MNIST images is bounded in [0, 1], both of the averaged signal-to-noise ratios (SNR), $\langle \frac{w_i^T x}{\sigma_i} \rangle$ and $\langle \frac{u_j^T r}{\lambda_j} \rangle$, will be bounded, even though we adopt a high-SNR setting. In other words, the learning rule cannot just increase synaptic weights W, U to combat the neural noise. The initial synaptic weights of W and U are randomly drawn from [0, 0.001]. The learning rate is fixed as $\eta_k = 1 \times 10^{-6}$.

The algorithm performance is evaluated by epochs, while in each epoch, 1000 randomly selected images are presented to the learning rule one by one. After each epoch, we first use the current parameters W, U to evaluate the VMI $\tilde{I}(x, r)$, and then reconstruct a testing set from its encoder representation. To evaluate the VMI, we draw 1000 images for each digit from the training set and then compute with equation A.1 and the current encoder and decoder parameters, while the constants in equation A.1 are

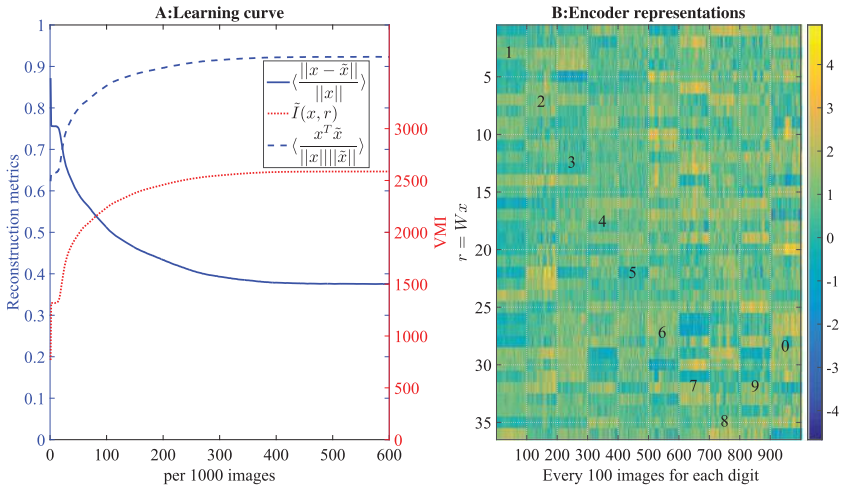

Figure 2: (A) The learning curve of averaged reconstruction error and the angle error over the testing set, and the variational mutual information at the end of each epoch. (B) The stochastic encoder neuron representations for 100 images of each digit after learning.

ignored. We also generate a testing set of 200 images by randomly drawing 20 images for each digit that was excluded from the training set. The reconstruction of the testing set is based on samples from encoder and decoder neurons as follows. Given an image x from the testing set, we first sample from the encoder $\mathcal{N}(Wx, \Sigma)$ to get a realization of its stochastic representation $\hat{r}$, and then sample $\hat{x}$ from the decoder $\mathcal{N}(U\hat{r}, \Lambda)$ by presenting $\hat{r}$ to the decoder. We repeat the sampling process 100 times for each image and average the sampled $\hat{x}$ to form a final reconstruction of x, denoted $\tilde{x}$. We can evaluate the reconstruction by the relative error $\frac{||x-\tilde{x}||}{||x||}$ and also by the similarity, $\cos\theta = \frac{x^T\tilde{x}}{||x||\cdot||\tilde{x}||}$. The learning curves (see Figure 2A) of VMI and the two metrics over the testing set show that algorithm 2 has a quick initial phase and gradually slows down as the synaptic weights approach the optimal ones. These curves imply that the MI between encoder response r and stimuli x is increased due to learning and that the information contained in the encoder representation r can be well utilized by the downstream neurons (decoder).

After learning, we resample 100 images for each digit from the testing database to get a new testing set. We reconstruct them with the learned encoder and decoder using the same procedure described before. Figure 2B shows the stochastic encoder representations of every 100 images for each digit. It is notable that the encoder representation is coherent for each digit among those 100 images while incoherent across different digits. The

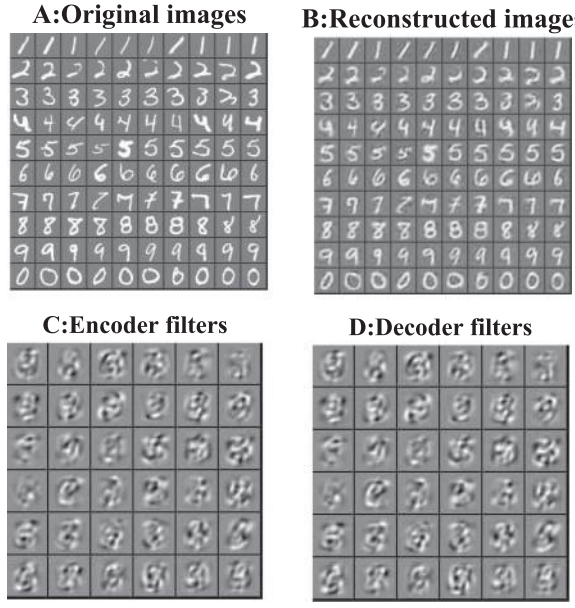

Figure 3: (A) The randomly selected images from MNIST testing database. (B) The reconstructed images with the learned linear gaussian encoder and decoder. (C) Each row of encoder synaptic weight $W = [w_1, \ldots, w_m]^T \in \mathbb{R}^{36\times 784}$ is an encoder filter. (D) Each column of decoder synaptic weight $U = [u_1, \ldots, u_N]^T \in \mathbb{R}^{784\times 36}$ is a decoder filter.

averaged reconstruction error $\langle \frac{||x-\tilde{x}||}{||x||} \rangle$ of those 1000 images is 0.3767, and the averaged $\langle \cos\theta \rangle$ is 0.9224. Figures 3A and 3B show 10 randomly selected images for each digit in the testing set and their reconstructed images. They all verify that the encoder neurons preserve the necessary information when compressing the high-dimensional images to low-dimensional representations, while the decoder can extract the information for downstream processing. Finally, Figures 3C and 3D show the learned synaptic weight patterns of encoder and decoder neurons. Each patch in Figure 3C is a visualization of a row of encoder weights $W = [w_1, \ldots, w_{36}]^T \in \mathbb{R}^{36\times 784}$ (i.e., an encoder filter), and each patch in Figure 3D is a visualization of a column of decoder weights $U = [u_1, \ldots, u_{784}]^T \in \mathbb{R}^{784\times 36}$.

3.2 Infomax Learning with Fixed Random Feedback Weights. We note that the encoder learning rule 3.2 in algorithm 2 requires each encoder neuron to know the weights of the synapses to which it is presynaptic. However, in biological neurons, the synaptic weight is largely determined by postsynaptic mechanisms (Grossberg, 1987). Therefore, the learning rule

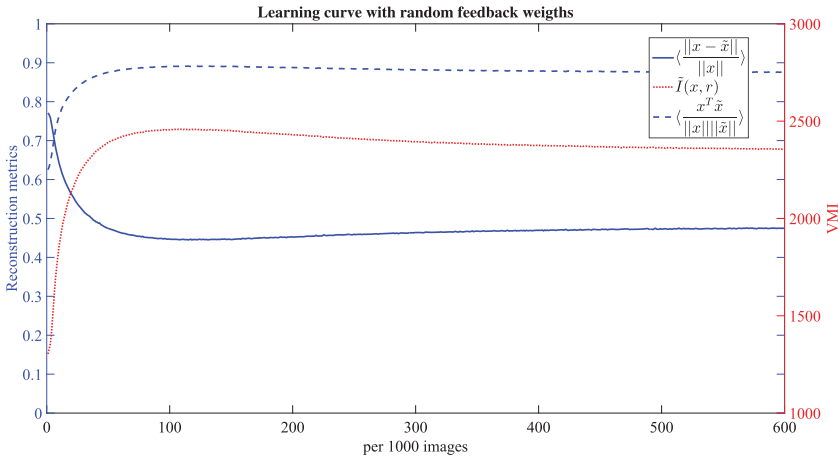

Figure 4: The learning curve of averaged reconstruction error and the angle error over the testing set, and the variational mutual information at the end of each epoch, with random fixed feedback weights in equation 3.4.

requires the weight of the synapse to be transported back to the neurons presynaptic to it. Biological mechanisms that achieve such "weight transport" are unclear (Lillicrap, Cownden, Tweed, & Akerman, 2016). In this sense, the proposed learning rule is only semilocal even though the learning rule relies on only pairwise neural interactions.

Recently, a strategy has been proposed to obviate the weight transport conundrum through a strategy referred to as as feedback alignment (Lillicrap et al., 2016). The key idea here is that rather than using exact parameterization of synaptic weights in terms of postsynaptic targets, a random projection of the error is used. We explored this strategy in our context, through modification of learning rule 3.2 of algorithm 2 to the following one:

$$w_i \leftarrow \mathcal{P}_{\mathcal{B}_2^N}\left\{ w_i + \eta_k \left[\left(\frac{u_{1i}^f}{\lambda_1}, \ldots, \frac{u_{Ni}^f}{\lambda_N} \right) (x_k - \hat{x}_k) \right] x_k \right\}. \tag{3.4}$$

Here $(u_{1i}^f, \ldots, u_{Ni}^f)$ are randomly generated but fixed feedback weights. In this case, the error signal is projected onto a random vector and sent back to encoder neurons; therefore, there is no need for weight transport.

To verify the infomax learning with random feedback weights, we adopt the same experiment setting as in section 3.1 but change the encoder learning rule, equation 3.2, of algorithm 2 to equation 3.4. The learning curves (see Figure 4) of VMI and the two metrics over the testing set show that

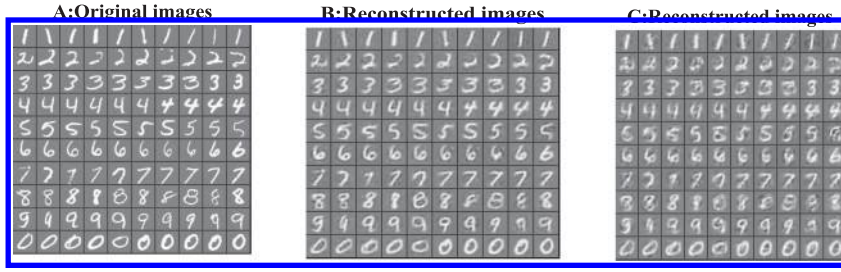

Figure 5: (A) Randomly selected images from MNIST testing database. (B) Reconstructed images with the encoder/decoder learned in section 3.1 with the encoder learning rule 3.2. (C) The reconstructed images with the encoder/decoder learned with the encoder rule 3.4, which uses fixed random feedback weights.

even with random feedback weights, algorithm 2 with the encoder learning rule, equation 3.4, can still increase the variational mutual information and the downstream neurons (decoder) can still improve the reconstruction of the original stimulus from its encoder representation. After learning, we resample 100 images for each digit from the testing database to get a new testing set. We reconstruct them with the encoder or decoder learned with the encoder rule, equation 3.4, that utilizes fixed random feedback weights. The averaged reconstruction error $\langle \frac{||x-\tilde{x}||}{||x||} \rangle$ of those 1000 images is 0.4700, and the averaged $\langle \cos\theta \rangle$ is 0.8782. Figures 5A to 5C show 10 randomly selected images for each digit in the testing set and their reconstructed images with the encoder and decoder learned with equations 3.2 and 3.4, respectively. Compared with the encoder/decoder learned in section 3.1, the encoder/decoder learned with fixed random feedback weights only slightly deteriorates algorithm performance. This shows neurons with random feedback weights can achieve fast but efficient infomax learning.

3.3 Adaptive Infomax with Samples of Dynamical Stimuli. In the natural world, the distribution $p(x)$ (i.e., the underlying stimulus statistics) changes persistently. Sensory neural circuits show strong adaptivity for such dynamic stimuli (Fairhall, Lewen, Bialek, & van Steveninck, 2001). By virtue of its online nature, our algorithms should demonstrate such adaptivity when stimuli are sampled from time-varying distributions.

We proceed to verify the adaptivity of algorithm 2 with the MNIST data set. The algorithm setting is the same as section 3.1, except that the stimulus distribution changes at some time point. Initially, we sample only images of the digits 1, 7, and 9 from the testing database and present them to algorithm 2 one by one. The learning is divided into epochs, while each epoch contains 100 images per digit. After 600 epochs, we start to sample images of all digits from the testing database and present them to algorithm 2 one by one.

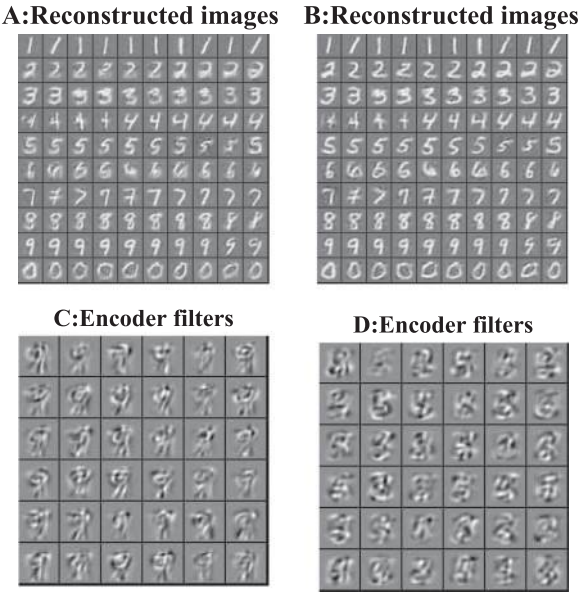

Figure 6: (A) Reconstruction of images with learned encoder/decoder at the 600th epoch. Note that only digits 1, 7 and 9 are presented. (B) Reconstruction of images with learned encoder/decoder at the 1200th epoch. (C) Learned encoder filters of W at the 600th epoch. (D) Learned encoder filters of W at the 1200th epoch.

Therefore, the underlying stimulus distribution changes, and the learning continues for another 600 epochs.

We compare the learned encoder/decoder at the 600th and 1200th epochs by reconstructing a set of test images, which contains 100 images for each digit. The averaged reconstruction error $\langle \frac{||x-\tilde{x}||}{||x||} \rangle$ is 0.4378 at the 600th epoch but 0.4017 at the 1200th epoch. Figures 6A and 6B show the reconstructed images for 10 randomly chosen images of each digit with the learned encoder/decoder at the 600th and 1200th epochs, respectively. Figure 6A shows that images of digits 1, 7, and 9 can be well reconstructed with the learned encoder/decoder at the 600th epoch, but the images of other digits cannot. After presenting images of all digits to the learning rule, the reconstruction of images of all digits has been improved, as shown by Figure 6B. Figure 6C presents the learned encoder filters at the 600th epoch, and Figure 6D presents the learned encoder filters at the 1200th epoch. Figure 7 shows the learning curve of the reconstruction of digits 7, 9, 2, and 6, respectively. They show that different features are adaptively learned when the stimulus distribution is changing.

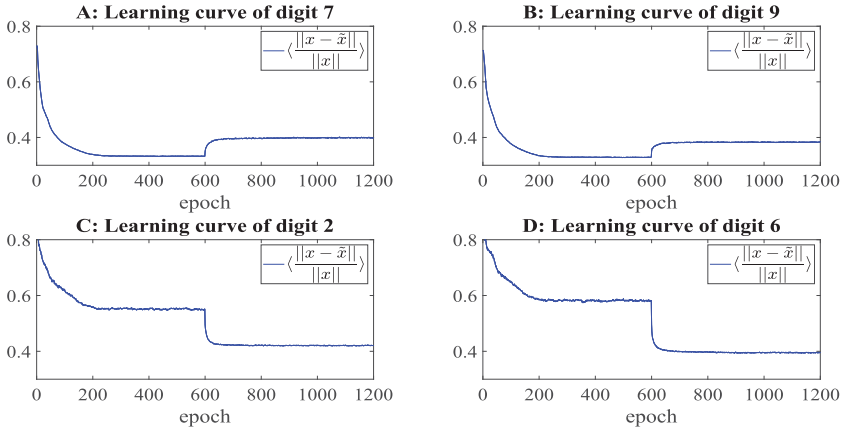

Figure 7: (A) Learning curve of the reconstruction error of digit 7. (B) Learning curve of the reconstruction error of digit 9. Since only the images of digits 1, 7, and 9 are presented at the first 600 epoches, their reconstruction errors decrease with learning. However, their reconstruction errors increase after 600 epochs since the images of more digits are presented and the "encoder resource" has to be redistributed. (C) Learning curve of the reconstruction error of digit 2. (D) Learning curve of the reconstruction error of digit 6. The reconstruction error decreases once these images are presented as stimuli.

4 Multiple Timescale Learning of Infomax with Energetic Constraints

The variational online learning framework is also flexible enough to balance information processing efficiency with energetic (analogous to physiological metabolic) costs.

To illustrate the idea, we retain the linear gaussian encoder model in section 3, $p(r|x;\theta) = \mathcal{N}(Wx;\Sigma)$, and the decoder is $q(x|r;\phi) = \mathcal{N}(Ur,\Lambda)$. Consider the infomax problems with a metabolic constraint that the mean squared encoder neuron response (firing rate) should be less than some level:

$$\max_{p(r|x;\theta)} I(x,r), s.t., E(r^T r) = \int_r p(r) r^T r dr \leq M. \tag{4.1}$$

Infomax with energetic constraints has been previously proposed (Wang et al., 2016; Karklin & Simoncelli, 2011) but handled using the addition of regularizing penalties to the main MI cost. The difference in our formulation, equation 4.1, is that we limit the mean squared encoder responses (which also incorporates the encoder noise variance) explicitly. In

algorithm 3, we solve equation 4.1 for a population of neurons by a learning algorithm with samples of unknown $p(x)$.

Compared with equation 3.2 in algorithm 2, a term modulated by a global signal ζ is added to the update rule of encoder synaptic weights $w_i, i = 1, \ldots, m$. The signal ζ monitors overall encoder response strength and provides homeostatic feedback that downregulates responses in order to satisfy the constraint. Hence, the term with ζ can be regarded as a three-term Hebbian learning term. The dynamical evolution of ζ is on a slower timescale, since it is updated every $\mathbf{T}$ samples. Therefore, learning rules 4.2 and 4.3 for synaptic weights w_i, u_j can be implemented with locally available information but modulated by a population-level slow signal. This learning rule amounts to a saddle point dynamics that seeks an equilibrium between conflicting objectives (Arrow, Hurwicz, & Uzawa, 1958). A detailed derivation of algorithm 3 is presented in section A.2.

4.1 Example: Infomax Learning from Natural Images under Energetic Constraint. We compare the infomax learning with and without energetic constraints—that is, algorithms 2 and 3 in terms of processing 8×8 patches sampled from a natural image database (Hyvarinen, Hurri, & Hoyer, 2009). We remove the DC component of the patches but do not perform any further preprocessing. The stimulus set has a total of 2×10^4 patches. During learning, image patches are randomly selected (sequentially) and simultaneously presented to algorithms 2 and 3.

The number of encoder neurons is set as $m = 400$, so we are seeking an overcomplete representation with infomax learning. The linear gaussian encoder $\mathcal{N}(W, \Sigma)$ and decoder $\mathcal{N}(U, \Lambda)$ have synaptic weights $W \in \mathbb{R}^{400\times 64}$, $U \in \mathbb{R}^{64\times 400}$ to be learned, while Σ, Λ are fixed diagonal matrices. The initial parameters of algorithms 2 and 3 are the same with W, U randomly drawn from [0, 0.1] and nonzero elements of Σ, Λ drawn from [0.05, 0.1]. Since the the stimuli and encoder responses are approximately bounded within $[-1, 1]$, the SNR is low: $SNR \leq \frac{1}{0.075} \approx 15$. We still impose the constraint that the synaptic weight of each encoder or decoder neuron is limited within the unit Euclidean ball: $w_i \in \mathcal{B}_2^{64}$ and $u_j \in \mathcal{B}_2^{400}$. For algorithm 3, the energetic constraint is $M = 80$ with $\zeta \in \mathbb{R}^+$ initially set to be 0.1. The learning process is divided into epochs while each epoch has $\mathbf{T} = 1000$ image patches. Therefore, ζ is adapted after each epoch, while W and U are adapted with each image patch. The learning rate of W, U is set as $\eta_k = 1 \times 10^{-6}$, and the learning rate of ζ is set as $\nu = 1 \times 10^{-4}$.

The results of algorithm 3 with energy constraints are depicted in Figures 8A and 8C, which show the evolution of the global modulator signal ζ and the violations of the constraint $\hat{sr} - M$, respectively. We observe that the global signal ζ is dynamically adapted and increases once the constraint is violated, as intended. Moreover, since $\hat{sr} - M$ asymptotically approaches zero in Figure 8C, the energy budget is fully used by the learned encoder/decoder. In other words, algorithm 3 has fully allocated the energy

Algorithm 3: Sample-Based Online Learning for Infomax Under Energetic Constraints.

Data: Samples of stimuli $p(x)$, the average energetic level M
Result: Encoder and decoder synaptic weights W, U
initialization;
while *A stimulus* $x_k \in \mathbb{R}^N$ *is sampled and presented to the neural network.* **do**

- The stimulus signal x_k is forward-processed by the encoder and decoder to get samples. First, the encoder gets a sample r_k according to its current encoding model, $r_k \sim \mathcal{N}(Wx_k, \Sigma)$. Then r_k is sent to the decoder, which gets a sample $\hat{x}_k$ with its decoding model $\hat{x}_k \sim \mathcal{N}(Ur_k, \Lambda)$. The reconstruction sample $\hat{x}_k$ is compared with x_k to give an error signal $x_k - \hat{x}_k$. The error signal is fed backward such that it is available to both encoder and decoder neurons.

- Encoder neuron i adapts its synaptic weight w_i:

$$w_i \leftarrow \mathcal{P}_{\mathcal{B}_2^N}\Big\{w_i + \eta_k\Big[\Big(\frac{u_{1i}}{\lambda_1}, \cdots, \frac{u_{Ni}}{\lambda_N}\Big)\Big(x_k - \hat{x}_k\Big)\Big]x_k - \eta_k \zeta r_{k,i} x_k\Big\}. \tag{4.2}$$

- Decoder neuron j adapts its synaptic weight u_j:

$$u_j \leftarrow \mathcal{P}_{\mathcal{B}_2^m}\Big\{u_j + \eta_k \frac{1}{\lambda_j}[(x_k - \hat{x}_k)_j r_k - (\sigma_1 u_{j1}, \cdots, \sigma_m u_{jm})^T]\Big\}. \tag{4.3}$$

- A counter variable counts the number of samples received: $counter = counter + 1$, and the averaged sum of squared encoder response is updated $\hat{sr} = \hat{sr} + \frac{1}{\mathbf{T}} r_k^T r_k$

if *counter*==**T then**

- The global modulator signal is updated:

$$\zeta \leftarrow \mathcal{P}_{\mathbb{R}_+}[\zeta + \nu(\hat{sr} - M)] \tag{4.4}$$

- The counter is reset: $counter = 0$, and the averaged sum of squared encoder responses is reset: $\hat{sr} = 0$.

end
end

budget to achieve the most information-efficient stimulus representation. We also evaluate the VMI after each epoch with a Monte Carlo method. We sample 10^4 patches and compute VMI with equation A.1 while the constants are ignored. Figure 8B shows the VMI after each epoch that are computed with an encoder/decoder learned with and without constraints, which demonstrates VMI is increased in both cases. Figure 8D gives the violations of the constraint $\hat{sr} - M$ when learning without energy constraints. Therefore, we conclude that the mutual information learned with an energy constraint could be almost the same as the case without energy constraint. This means that by efficiently allocating the energy budget, we can achieve the same information transmission while keeping the energy consumption within the constraint.

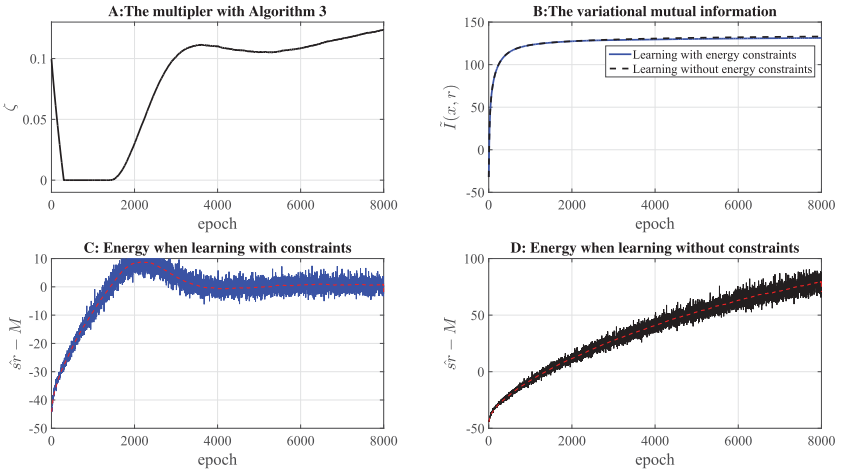

Figure 8: Comparing infomax learning with and without energetic constraints. (A) The trajectories of the modulator signal ζ with algorithm 3 after each epoch. (B) The trajectories of variational mutual information after each epoch that are computed with the encoder or decoder learned with and without energy constraints, respectively. It shows that the mutual information learned with an energy constraint is almost the same as the case without an energy constraint. (C) The trajectory of the violation of energy constraint $\hat{s}r - M$ when learning with energy constraints, where the dashed red line is the smoothed one. (D) The trajectory of the violation of energy constraint $\hat{s}r - M$ when learning without energy constraints, where the dashed red line is the smoothed one.

To further compare the performance of algorithms 3 and 2, that is, infomax learning with and without energy constraints, we reconstruct 5000 image patches from the database using the same sampling method described in section 3.1. Figures 9A and 9B show the distribution of randomly selected 20 encoder neurons' activities over the testing set, computed with the encoder/decoder learned with and without energy constraints, respectively. They shows that with energy constraints, the neural activities are more concentrated around zero. Figures 9C and 9D show the histogram of reconstruction error $\frac{||x-\tilde{x}||}{||x||}$ over the testing set where $\tilde{x}$ is reconstructed with the encoder/decoder learned with and without energetic constraints. The averaged $\langle \frac{||x-\tilde{x}||}{||x||} \rangle$ over the testing set with energy constraint is 0.2009, while it is 0.1927 without energy constraints. Hence, the reconstruction performance is indistinguishable for the two cases. However, the mean $\langle r^T r - M \rangle$ over the testing set for the encoder/decoder learned with energy constraints is 0.1526, while it is 76.6665 for the encoder/decoder learned without energy constraints. Figures 9E and 9F show the histograms of $r^T r$ computed with the encoder/decoder learned with and without energy constraints with a

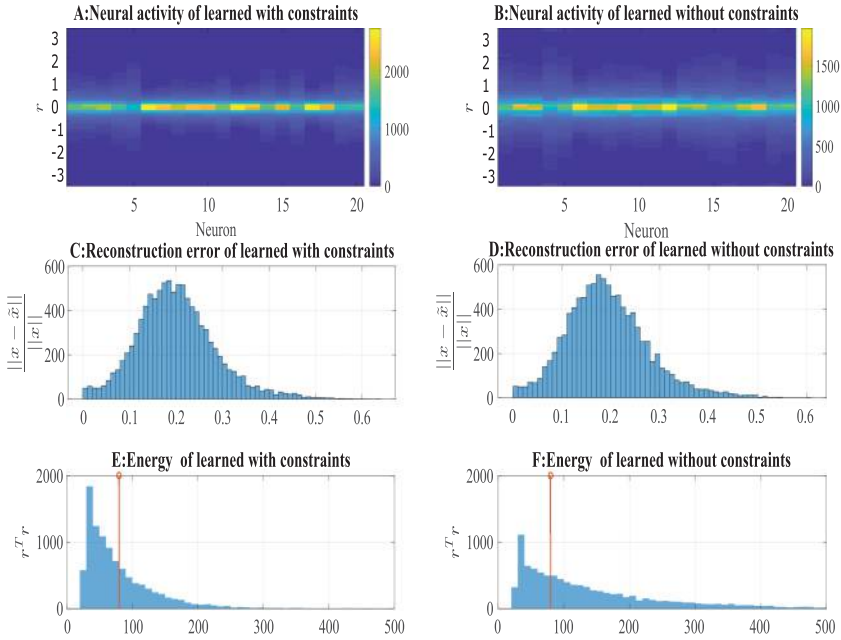

Figure 9: Reconstruction of image patches with the encoder decoder learned with and without energy constraints on the same testing set. (A) The image of the distribution of the selected encoder neuron's activities among $[-3, 3]$ with 50 bins, while neural activities are computed with an encoder/decoder learned with energy constraints. (B) The image of the distribution of the selected encoder neuron's activities among $[-3, 3]$ with 50 bins, while neural activities are computed with an encoder/decoder learned without energy constraints. (C) The histogram of reconstructed errors with the encoder/decoder learned with energy constraints. (D) The histogram of reconstructed errors with the encoder/decoder learned without energy constraints. (E) The histogram of neural activities energy computed with the encoder/decoder learned with energy constraints. (F) The histogram of neural activities energy computed with the encoder/decoder learned without energy constraints.

vertical line marking the location of the energy budget. They show that with the learning rules in algorithm 3, the energy constraint is satisfied for more image patches and the energy consumption is much less than the case without energy constraints.

As these results suggest, despite using substantially less energy, the encoder/decoder resulting from algorithm 3 achieves comparable performance in terms of information efficiency and reconstruction error. Indeed, by incorporating a slower timescale modulatory signal, neural coding can be made efficient in term of both information processing and energy costs.

5 Infomax Learning with Bernoulli Neurons

In previous sections, we adopted the linear gaussian encoder that transforms continuous stimuli to continuous neural responses r, analogous to a neural circuit that encodes stimuli with its firing rate. However, it is also interesting to investigate how a population of neurons efficiently encodes continuous stimuli with discrete responses analogous to neural spikes. Both Poisson and Bernoulli neural response models have been proposed for discrete encoding (Meyer, Williamson, Linden, & Sahani, 2017), and population coding of spiking neurons based on infomax has been investigated in Tkačik et al. (2010). In this section, we show that the proposed online learning scheme, algorithm 2, can also be applied to infomax learning of a population of Bernoulli spiking neurons.

Suppose the stimulus is still taken from a continuous space $\mathbb{R}^N$ and follows an unknown distribution $p(x)$. Given a sample x, each encoder neuron's response is taken from a set of two symbols $\Delta = \{1, -1\}$. The linear-nonlinear Bernoulli encoding model of neuron i is

$$\begin{aligned} P(r_i = 1|x; w_i) &= \frac{1}{1 + \exp^{-w_i^T x}}, \\ P(r_i = -1|x; w_i) &= 1 - \frac{1}{1 + \exp^{-w_i^T x}}, \end{aligned} \tag{5.1}$$

where $w_i \in \mathbb{R}^N$ is the synaptic weight of encoder neuron i. Hence, a population of m encoder neurons transforms the stimuli x to a symbol assembly, such as $(1, -1, 1, -1, -1, \ldots)$. We denote the set of all possible symbol assemblies as $\Delta^m = \prod_{i=1}^m \Delta$. Suppose the encoder neurons are conditionally independent of each other given the stimuli. Then the encoder model of the population of m neurons is

$$P(r|x; \theta) = \prod_{i=1}^{m} P(r_i|x; w_i), \quad r \in \Delta^m, r_i \in \Delta, \theta = \{w_1, \ldots, w_m\}. \tag{5.2}$$

The goal of the encoder neurons is to learn optimal synaptic weights $\theta = \{w_1, \ldots, w_m\}$ by solving the infomax problem, equation 2.1, which is now given as

$$\begin{aligned} &\max_{\theta} I(x, r), \\ &I(x, r) = H(r) - H(r|x) \\ &= -\sum_{r \in \Delta^m} P(r; \theta) \log P(r; \theta) + \int_x dx p(x) \sum_{r \in \Delta^m} P(r|x; \theta) \log P(r|x; \theta). \end{aligned} \tag{5.3}$$

Notice that $P(r; \theta) = \int_x p(x) P(r|x; \theta) dx$, and then $p(x|r; \theta) = \frac{P(r|x;\theta)p(x)}{P(r;\theta)}$.

Remark 2. Here, each encoder neuron is a computational unit that may be implemented with a neural circuit of two biological neurons. Those two biological neurons have strongly mutual inhibitory connections such that only one of them will fire given a stimulus input. Therefore, we can take $r_i = 1$ if one of the two neurons fires and $r_j = -1$ if the other fires.

As before, we aim to give an online learning algorithm wherein each neuron uses only pairwise neural interaction and neural activity to adapt its synaptic weight. We again employ the scheme of algorithm 2 to derive the online learning rule and specifically assume the variational linear gaussian decoder $q(x|r; \phi) = \mathcal{N}(Ur, \Lambda)$, introduced in section 2.2, to approximate the true posterior.

This results in an online learning rule for Bernoulli neurons in algorithm 4, derived in full in section A.3.

Learning rule 5.5 can be enacted with local computations. For the encoder synaptic weight w_i, both $\hat{\text{Var}}(r_i|x_k)$ and $\hat{E}(r_i|x_k)$ can be computed with locally available samples. Moreover, u_{ji}, the synaptic weight from the ith encoder neuron to the jth decoder neuron, can also be regarded as local information to the encoder neuron i, even though the transport of synapse weights from postsynaptic to presynaptic neurons is still needed. Since $\hat{x}_k$ is assumed to be fed backward from decoder to each encoder neuron, $x_k - \hat{x}_k$ is known by the ith encoder neuron. Similar arguments hold for learning rule 5.6 for decoder synaptic weight u_j.

5.1 Example: Learning to Encode Gaussian Mixture Distribution with Binary Spikes. We use gaussian mixture stimuli to show that the sample-based online learning rule in algorithm 4 can find efficient stochastic discrete representations that preserve as much information as possible when transforming continuous stimuli to binary spikes. Here, the stimulus $x \in \mathbb{R}^2$ follows a gaussian mixture distribution $p(x) = \sum_{i=1}^{8} \pi_i \mathcal{N}(\mu_i, \Sigma_i)$. $p(x)$ has a total of eight clusters in $\mathbb{R}^2$, where each cluster is a gaussian distribution with its center $\mu_i \in \mathbb{R}^2$ and variance $\Sigma_i \in \mathbb{R}^{2\times 2}$. The weight of each cluster, π_i, determines the frequency with which a stimulus is sampled from a cluster i. Each cluster's mean vector $\mu_i \in \mathbb{R}^2$ is uniformly generated from $[-4, 4] \times [-4, 4]$, and its covariance Σ_i is also randomly generated. The weight of each cluster π_i is randomly drawn from $[0.3, 1]$, and then all π_i are normalized to be a probability vector: $0 \le \pi_i \le 1$, $\sum_{i=1}^{8} \pi_i = 1$. Each nonzero element of Λ is randomly taken from $[0.1, 0.3]$, resulting in $SNR \approx \frac{2}{0.2} = 10$. The number of encoder neurons is 32. We utilize the learning rules in algorithm 4 to update $\theta = \{w_1, \ldots, w_{32}\}$, $w_i \in \mathbb{R}^{2\times 1}$ and $U \in \mathbb{R}^{2\times 32}$ with the constraints $u_j \in \mathcal{B}_2^{32}$, $j = 1, 2$. Each element of the initial w_i is randomly drawn from $[-0.1, 0.1]$, and each element of initial U is randomly drawn from a normal distribution. The number of samples in algorithm 4 used for approximating $\hat{\sigma}_{i,k}$ and $\hat{x}_k$ is $\mathbf{T} = 200$. The learning rate is $\eta_k = 1 \times 10^{-6}$. The learning trajectory for the encoder synaptic weight $w_1 \in \mathbb{R}^2$ is given in

Algorithm 4: Sample-Based Infomax Learning for Bernoulli Spiking Neurpus.

Data: Samples of underlying stimuli $p(x)$

Result: Synaptic weights of encoder and decoder, $\theta = \{w_1, \cdots, w_m\}$, $U = [u_1, \cdots, u_N]^T$

initialization;

while *A stimulus* $x_k \in \mathbb{R}^N$ *is sampled from* $p(x)$ **do**

- The stimulus x_k is forward-processed by the encoder and decoder to get the sample-based estimations. Set $\hat{\sigma}_{i,k} = 0, i = 1, \cdots, m$ and $\hat{x}_k = \mathbf{0}$.

  for $l = 1, \cdots, \mathbf{T}$ **do**

  - Each encoder neuron i gets a sample $r^l_{i,k}$ with its current model $r^l_{i,k} \sim P(r_i|x_k; w_i)$, and $\hat{\sigma}_{i,k} \leftarrow \hat{\sigma}_{i,k} + \frac{1}{2\mathbf{T}}\big(r^l_{i,k} + 1\big)$.
  - The sample $r^l_k = (r^l_{1,k}, \cdots, r^l_{m,k})^T$ is forward sent to the decoder, which gets a sample $\hat{x}^l_k \sim \mathcal{N}(Ur^l_k, \Lambda)$, and $\hat{x}_k \leftarrow \hat{x}_k + \frac{1}{\mathbf{T}}\hat{x}^l_k$.

  end

- The encoder and decoder neurons calculate

$$\hat{E}(r_i|x_k) = 2\hat{\sigma}_{i,k} - 1, \quad \hat{\mathrm{Var}}(r_i|x_k) = 4\hat{\sigma}_{i,k}(1 - \hat{\sigma}_{i,k}), \tag{5.4}$$

  with $\hat{E}(r|x_k) = (\hat{E}(r_1|x_k), \cdots, \hat{E}(r_m|x_k))^T$, and $\hat{x}_k$ is fed backward to encoder neurons.

- Encoder neuron i adapts its synaptic weight w_i,

$$w_i \leftarrow w_i + \frac{1}{2}\eta_k\left\{\Big(\frac{u_{1i}}{\lambda_1}, \cdots, \frac{u_{Ni}}{\lambda_N}\Big)\Big(x_k - \hat{x}_k\Big) + \hat{E}(r_i|x_k)\sum_{j=1}^{N}\frac{u^2_{ji}}{\lambda_j}\right\}\hat{\mathrm{Var}}(r_i|x_k)x_k. \tag{5.5}$$

- Decoder neuron j adapts its synaptic weight u_j,

$$u_j \leftarrow \mathcal{P}_{\mathcal{B}^m_2}\left\{u_j + \eta_k\frac{1}{\lambda_j}\left[(x_k - \hat{x}_k)_j\hat{E}(r|x_k) - (\hat{\mathrm{Var}}(r_1|x_k)u_{j1}, \cdots, \hat{\mathrm{Var}}(r_m|x_k)u_{jm})^T\right]\right\}. \tag{5.6}$$

end

Figure 10A. We also evaluate the variational mutual information $\tilde{I}(x, r)$ once 1000 samples are presented with a Monte Carlo method and equation A.17, while the constants are ignored. The trajectory of $\tilde{I}(x, r)$ is shown in Figure 10B, which implies that the mutual information is increased with the learning rule.

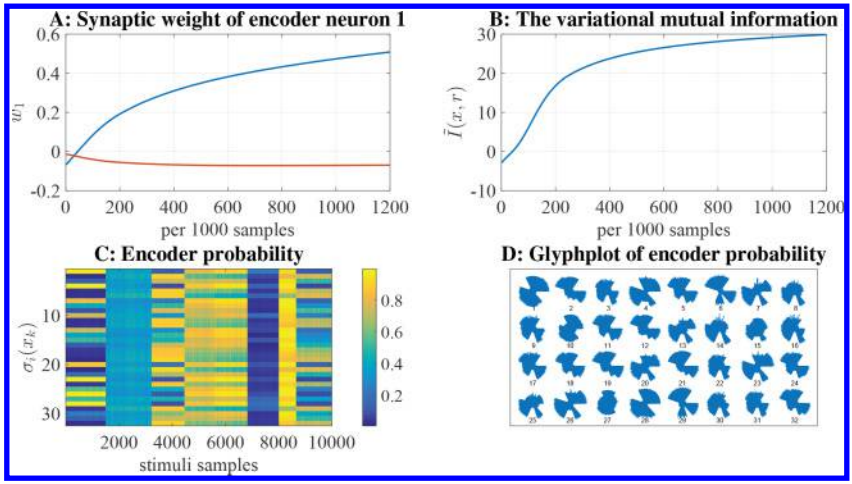

Figure 10: (A) Learning trajectory of synaptic weight w_1. (B) Learning trajectory of the variational mutual information $\tilde{I}(x, r)$. (C) Image of the probablity that encoder neurons take a value of 1 (i.e., $\sigma_i(x_k)$) for samples from different clusters. (D) Glyph plot of the probablity that encoder neurons take a value of 1 for the testing set, where the samples from the same cluster are put adjacent to each other.

After learning, we regenerate 1×10^4 samples from the mixture gaussian and test the learned encoder/decoder by reconstructing those samples. The averaged reconstruction error $\langle \frac{||x-\tilde{x}||}{||x||} \rangle$ over the testing set is 0.1717, while the averaged similarity $\langle \cos\theta \rangle$ between samples and their reconstruction over the testing set is 0.9965. Figures 11A and 11B show the heat map of the testing set and its reconstruction with the learned encoder/decoder. They verify that the cluster information of the gaussian mixture stimuli is well preserved by the binary stochastic encoder representations, and the encoder information can be used by the downstream neurons to recover the original stimuli. Figure 10C shows the probability that encoder neurons take a value of 1 given samples from different clusters. Figure 10D gives the glyph plot of the probability that encoder neurons take a value of 1 given samples from different clusters. They demonstrate that a specific stochastic binary representation of encoder neurons emerges for samples of different clusters after learning.

6 Discussion and Conclusion

In this letter, we have addressed the problem of neural coding from the perspective of mutual information maximization and variational online

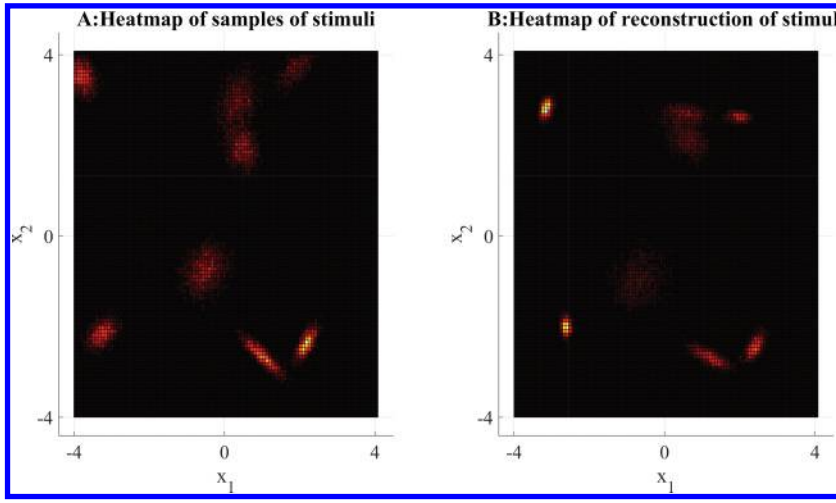

Figure 11: (A) Heat map of the testing set of stimuli samples. (B) Heat map of the reconstruction of testing samples with the learned encoder/decoder.

learning. While our model formulation is abstract, the proposed learning rules are biologically plausible in the following sense:

- Stimulus samples are processed in an online manner rather than in a batch manner. This obviates the need to "store" data—a biologically problematic premise—within a batched scheme. With online learning rules, sensory neurons adapt their encoding parameters in real time when the underlying stimulus distribution is changing.
- The nonlocal terms in the gradient of variational MI can be approximated with samples from decoder/encoder neurons. In this way, the variational decoder plays a crucial role during approximated gradient computation beyond the derivation of a lower MI bound. The use of decoder samples in this scheme has not been explored in previous variational MI works. Specifically, the sample-based learning rule removes the requirement that each neuron has global network awareness. This supposition is consistent with a recent hypothesis that neural circuits perform Bayesian computation with samples rather than probability (Sanborn & Chater, 2016). The current learning rule still requires each encoder neuron to know the weights of synapses that it is presynaptic to, while the biological mechanism to achieve such a weight transport from postsynaptic neuron to presynaptic neuron is unclear. This issue is a substantial and active research topic in biological learning.
- With the variational online learning scheme, an energetic constraint on encoder neural responses can be incorporated with infomax by

adding a modulatory signal with slower timescale dynamics. The scheme is different from the energy-penalty-based infomax in Karklin and Simoncelli (2011), which employs a fixed regularizer added to the infomax cost. In our case, we use a hard constraint, leading to a multiplier that adapts based on encoder neural response history on a slow timescale.

Our numerical examples illustrate the efficacy of the proposed rules and also highlight the manifestation of neural response selectivity and continual adaption to dynamic stimuli. Thus, these phenomena are compatible with online, sample-based infomax learning.

The learning rule is based on the variational method, an efficient approximate Bayesian inference method. The variational approach is closely related to the free energy hypothesis that the brain minimizes free energy for perception and action (Friston, 2010). In fact,

$$I(x,r) - \tilde{I}(x,r) = \int_r dr p(r) \int_x p(x|r;\theta) \log \frac{p(x|r;\theta)}{q(x|r;\phi)}. \tag{6.1}$$

The right term in equation 6.1 is an expectation of conditional free energy (Friston, 2012). Hence, maximizing $\tilde{I}(x,r)$ could be regarded as a minimizing of free energy. Moreover, the variational decoder $q(x|r;\phi)$ can be interpreted as a cognitive model in the sense of Helmholtz. In this case, the encoder $p(x|r;\theta)$ is a model of the sensory neural system, while $q(x|r;\phi)$ is a model of a higher-level cognitive system. The co-optimization of encoder $p(x|r;\theta)$ and decoder $q(x|r;\phi)$ then embodies a lower-level sensory neural system modulated by feedback from higher-level cognition functions (Gilbert & Li, 2013).

The normative view of theoretical neuroscience posits neural circuits as optimization/decision entities. In this context, there are two principles to be clarified: the optimization objective function and the optimization algorithm. In our work, a lower bound of MI is taken as the objective function. The proposed online learning rule is similar to the wake-and-sleep algorithm for a Helmholtz machine (Hinton, Dayan, Frey, & Neal, 1995), since it learns both the perception (encoding) and recognition (decoding) model. The wake-and-sleep algorithm is conjectured to be the mechanism for free energy minimization in the brain (Friston, 2010), and it is shown to be implementable by spiking neural circuits (Pavel & Miller, 2015). However, the wake-and-sleep algorithm is ad hoc since it optimizes different objective functions during the wake phase and the sleep phase (Rezende et al., 2011). Our work provides comparable results using a consistent, information-based objective function. From the optimization algorithm perspective, the learning rule could be treated as a stochastic gradient algorithm or incremental gradient algorithm. Both algorithms are understood well in

the operations research community, while their convergence is usually guaranteed for convex problems with diminishing step sizes. For biological learning, the nondiminishing step size enables the system to achieve continual learning even in a nonstationary environment and hence enables the neural system to adapt to dynamical stimuli.

Variational learning performance is limited by the model capacity of the decoder distribution. Taking the multimodal mixture gaussian distribution as an example, the exact posterior $p(x|r)$ should also be a multimodal distribution. When $|r| \gg |x|(m \gg N)$, the posterior $p(x|r)$ will be concentrated around a few peaks. In this case, $q(x|r;\phi) = \mathcal{N}(Ur, \Lambda)$ used in this work can serve as a good approximation of the true posterior $p(x|r)$. Even when we have more encoder neurons than stimulus dimension $m > N$, in which case $p(x|r)$ will be highly peaked, approximating a multimodal distribution with a unimodal gaussian distribution $q(x|r;\phi)$ will inevitably bring errors. If the decoder distribution $q(x|r;\phi)$ is flexible enough such that for any encoding parameter θ, there exists ϕ to ensure $q(x|r;\phi) = p(x|r)$, it is possible to learn infomax with the variational method. Since $q(x|r;\phi)$ is realized by the higher-level cortical networks we conjecture that it is not limited to gaussian distributions but could approximate more complicated ones. This investigation is suitable for future work.

The variational infomax method is based on the approximation of the posterior with gaussian distributions. Certainly there is a theoretical gap between the true posterior and this approximation, though the tightness of this gap is not yet well understood (Blei, Kucukelbir, & McAuliffe (2017)). Moreover, because the proposed learning algorithm is based on stochastic gradient descent for a nonconvex problem, we cannot guarantee convergence to the global optimum. Thus, there are at least two levels of suboptimality to which the converged solution may be subjected. Understanding this degradation from an analytical perspective is thus an important question, though one that we do not tackle here.

The balance between information processing efficiency and energetic cost in neural coding has been extensively discussed (Levy & Baxter, 1996; Laughlin, 2001; Balasubramanian, Kimber, & Berry, 2001), though how the neuron system learns such a balance with biological rules is unknown. Our results show that an averaged energy cost (in terms of the sum of squared encoder responses) can be held within a desirable range by using a multiple timescale learning rule with modulatory signals. In principle, this framework could be extended to other constraints, leading to even more diverse auxiliary dynamics. For example, it is desirable that the encoder neural response be as independent as possible to reduce redundancy and exhibit sparse spatiotemporal activation.

The proposed variational infomax learning resembles the variational autoencoder (VAE), a recent fixture in unsupervised learning (Kingma & Max, 2013). Both VMI and VAE adopt variational Bayesian inference as a

subroutine. Furthermore, VAEs can be treated in terms of minimizing a reconstruction error plus a regularization term. However, there are important differences between the strategy proposed here (infomax with hard energy constraints) and regularized VAEs.

VAEs aim to learn a probabilistic generative model by maximizing the data likehood. That is, VAEs assume a generative model $q(x) = \int_r q(x|r)q(r)dr$ for the observed data and then maximize the likehood of $\log q(x)$ over independent samples. The hidden variable r is assumed with a fixed prior distribution $q(r)$ (often a gaussian), and then the generative mapping $q(x|r)$ is learned to maximize the likelihood of $q(x)$. In contrast, infomax assumes an analytic model $p(x|r)$ and maximizes the mutual information $I(x, r)$ between observation x and the hidden variable r. The distribution of r is $p(r) = \int_x p(x|r)p(x)dx$ (after learning). No prior distribution is assumed, thus allowing for general hidden variable distributions.

Beyond this conceptual difference, there are differences in mathematical formulation, solution strategy, and ensuing interpretation. In section A.4, we detail the mathematical relationship between the objective functions for VAE and the proposed infomax problem. As shown there, there is indeed commonality in the two formulations. Specifically, we note the presence of a term $\sum_{k=1}^{N} \int_r p(r|x_k; \theta)rr^T dr$ that appears in both objectives and is central to achieve coding with efficient energy utilization.

The key development in our results pertains to the treatment of this term: the use of a Lagrangian saddle point optimization strategy with a multiplier ζ (see sections A.2 and A.4). As a result, ζ adapts according to the neural responses and the available energy budget, while VAEs and recent variants thereof (β-VAE in Higgins et al., 2017) use a fixed weight on $\sum_{k=1}^{N} \int_r drp(r|x_k; \theta)rr^T$. The adaptation of ζ amounts to the use of modulatory processes within the overall learning dynamics that carry a normative biological interpretation. In particular, these processes suggest a functional role for slower signaling pathways (e.g., widespread regulation of neuromodulators). This amounts to balancing information processing efficiency and energy consumption within the network. Indeed, the learning rule for infomax with energy constraints has a three-term Hebbian form and multiple timescale dynamics, providing a normative explanation for why such dynamics may be observed biologically.

Appendix: Derivations of Learning Rules and Discussions on VAE

A.1 Derivation of Learning Rule for Linear Gaussian Encoder. In this part, the encoder is $p(r|x; \theta) = \mathcal{N}(Wx, \Sigma)$, and the decoder is $q(x|r; \phi) = \mathcal{N}(Ur, \Lambda)$, where W, U are the neuron synaptic weights to be learned, while Σ, Λ are fixed neuron noise parameters.

To apply the online learning scheme in section 2.2, we need to calculate an approximated gradient of $L_k(W, U) = \int_r p(r|x_k; \theta) \log q(x_k|r; \phi)dr$ after a

stimulus x_k is sampled and presented to the neural network:

$$\begin{aligned} L_k(W,U) &= \int_r dr p(r|x_k;\theta)\log q(x_k|r;\phi) \\ &= \int_r dr\mathcal{N}(Wx_k,\Sigma)\log\frac{1}{\sqrt{2\pi|\det\Lambda|}} \\ &\quad\times\exp\left\{-\frac{1}{2}x_k^T\Lambda^{-1}x_k+x_k^T\Lambda^{-1}Ur-\frac{1}{2}r^TU^T\Lambda^{-1}Ur\right\} \\ &= constant+\int_r dr\mathcal{N}(Wx_k,\Sigma)\left[x_k^T\Lambda^{-1}Ur-\frac{1}{2}r^TU^T\Lambda^{-1}Ur\right] \\ &= constant+x_k^T\Lambda^{-1}UWx_k \\ &\quad-\frac{1}{2}\left[Tr(U^T\Lambda^{-1}U\Sigma)+x_k^TW^TU^T\Lambda^{-1}UWx_k\right]. \end{aligned} \tag{A.1}$$

Then we have

$$\nabla_W L_k(W,U) = U^T\Lambda^{-1}x_kx_k^T - U^T\Lambda^{-1}UWx_kx_k^T, \tag{A.2}$$

$$\nabla_U L_k(W,U) = \Lambda^{-1}x_kx_k^TW^T - \Lambda^{-1}U\Sigma - \Lambda^{-1}UWx_kx_k^TW^T. \tag{A.3}$$

The exact gradients A.2 and A.3 are too complicated to be calculated by the encoder or decoder neurons, since all synaptic weights are needed for each neuron. We approximate the terms related with the expectation of neuron response or stimuli reconstruction by means of stochastic sampling. This is consistent with the hypothesis that the brain computes by sampling rather than with exact probability distributions (Sanborn & Chater, 2016), which also leads to biologically implementable rules. We could approximate Wx_k with a sampled response r_k of encoder network $\mathcal{N}(Wx_k,\Sigma)$ when the stimulus x_k is given. We could also approximate UWx_k by a sample $\hat{x}_k$ from the decoder $\mathcal{N}(Ur_k,\Lambda)$ after the sample r_k from the encoder has been given. In fact, $r_k = Wx_k+\varepsilon_1$ and $\hat{x}_k = UWx_k+U\varepsilon_1+\varepsilon_2$, where ε_1, ε_2 are zero-mean gaussian processes. Hence, r_k and $\hat{x}_k$ can be regarded as the unbiased estimates of Wx_k and UWx_k. Then we have the approximated gradients of equations A.2 and A.3 as follows:

$$\nabla_W L_k(W,U) \approx U^T\Lambda^{-1}x_kx_k^T - U^T\Lambda^{-1}\hat{x}_kx_k^T, \tag{A.4}$$

$$\nabla_U L_k(W,U) \approx \Lambda^{-1}x_kr_k^T - \Lambda^{-1}U\Sigma - \Lambda^{-1}\hat{x}_kr_k^T. \tag{A.5}$$

Then we write the above equations in term of synaptic weights w_i and u_j and get the learning rules in equations 3.2 and 3.3.

A.2 Derivation of Learning Rules for Energy-Constrained Infomax. Here we give the derivation of algorithm 3 for infomax with energetic

constraints (see equation 4.1) for the linear gaussian encoder and decoder. Suppose we have $\mathbf{T}$ samples of stimuli, $\{x_1, \ldots, x_{\mathbf{T}}\}$. Then we can approximate the unknown distribution $p(x)$ with an empirical distribution $\hat{p}(x) = \frac{1}{\mathbf{T}} \sum_{k=1}^{\mathbf{T}} \delta(x - x_k)$. Since $p(r) = \int_x p(x)p(r|x;\theta)dr$, we approximate it as $p(r) = \frac{1}{\mathbf{T}} \sum_{k=1}^{\mathbf{T}} p(r|x_k;\theta)$. The constraint term $E(r^T r) = \int_r p(r) r^T r dr = \int_r p(r) Tr(rr^T) dr$ is approximated as $\frac{1}{\mathbf{T}} \sum_{k=1}^{\mathbf{T}} \int_r p(r|x_k;\theta) Tr(rr^T) dr$. We also approximate the mutual information like equation 2.15, resulting in the following constrained optimization problem:

$$\begin{aligned} &\max_{\theta\in\Theta,\phi\in\Phi} \quad \frac{1}{\mathbf{T}} \sum_{k=1}^{\mathbf{T}} \int_r p(r|x_k;\theta) \log q(x_k|r;\phi) dr \\ &s.t. \quad \frac{1}{\mathbf{T}} \sum_{k=1}^{\mathbf{T}} \int_r p(r|x_k;\theta) Tr(rr^T) dr \le M. \end{aligned} \tag{A.6}$$

Performing Lagrangian saddle point optimization with a multiplier $\zeta \in \mathbb{R}_+$ produces (Arrow et al., 1958)

$$\begin{aligned} \min_{\zeta\in\mathbb{R}_+} \max_{\theta\in\Theta,\phi\in\Phi} \quad & \frac{1}{\mathbf{T}} \sum_{k=1}^{\mathbf{T}} \left\{ \int_r p(r|x_k;\theta) \log q(x_k|r;\phi) dr \right. \\ & \left. - \zeta \left(\int_r Tr(rr^T) p(r|x_k;\theta) \right) dr - M) \right\}. \end{aligned} \tag{A.7}$$

Given the encoder model $p(r|x;\theta) = \mathcal{N}(Wx, \Sigma)$ and the decoder model $q(x|r;\phi) = \mathcal{N}(Ur, \Lambda)$, we have

$$\begin{aligned} \int_r Tr(rr^T) p(r|x_k;\theta) dr &= \int_r Tr(rr^T) \mathcal{N}(Wx_k, \Sigma) dr \\ &= Tr\left(\int_r rr^T \mathcal{N}(Wx_k, \Sigma) dr \right) \end{aligned} \tag{A.8}$$

$$= Tr(\Sigma + Wx_k x_k^T W^T) = Tr(\Sigma) + x_k^T W^T W x_k. \tag{A.9}$$

Combined with equation A.1, the Lagrangian optimization, equation A.7, turns out to be

$$\min_{\zeta\in\mathbb{R}_+} \max_{\theta\in\Theta,\phi\in\Phi} \quad \frac{1}{\mathbf{T}} \sum_{k=1}^{\mathbf{T}} \{\mathcal{L}_k(W, U, \zeta)\} \tag{A.10}$$

$$\begin{aligned} \mathcal{L}_k(W, U, \zeta) = & x_k^T \Lambda^{-1} U W x_k \\ & - \frac{1}{2} [Tr(U^T \Lambda^{-1} U \Sigma) + x_k^T W^T U^T \Lambda^{-1} U W x_k] \\ & - \zeta (Tr(\Sigma) + x_k^T W^T W x_k - M). \end{aligned} \tag{A.11}$$

The learning algorithm is based on a two-timescale gradient algorithm for solving the saddle point problem, equation A.10. Suppose ζ is fixed as a constant over the $\mathbf{T}$ samples. Then we can calculate the gradient of $\mathcal{L}_k(W, U, \zeta)$ with respect to W and U. In fact, $\nabla_U \mathcal{L}_k(W, U, \zeta) = \nabla_U L_k(W, U)$ in equation A.3, and we have

$$\nabla_W \mathcal{L}_k(W, U, \zeta) = U^T \Lambda^{-1} x_k x_k^T - U^T \Lambda^{-1} U W x_k x_k^T - \zeta W x_k x_k^T. \tag{A.12}$$

We approximate the Wx_k and UWx_k in the gradient $\nabla_W \mathcal{L}_k(W, U, \zeta)$, $\nabla_U \mathcal{L}_k(W, U, \zeta)$ with samples r_k from the encoder and $\hat{x}_k$ from the decoder, respectively, and get the approximated gradients of w_i and u_i on the right-hand side of learning rules 4.2 and 4.3. The update of w_i and u_i is a projected gradient ascent with each sample x_k when ζ is treated as a constant.

Suppose the $\mathbf{T}$ samples $\{x_1, \ldots, x_T\}$ all have been presented. Then the gradient of $\frac{1}{\mathbf{T}} \sum_{k=1}^{\mathbf{T}} \mathcal{L}_k(W, U, \zeta)$ with respect to ζ is $\frac{1}{\mathbf{T}} \sum_{k=1}^{\mathbf{T}} (Tr(\Sigma) + x_k^T W^T W x_k) - M$. Noticing that $r_k = Wx_k + \varepsilon_k$, and $\varepsilon_k \sim \mathcal{N}(0, \Sigma)$, we can approximate $\frac{1}{\mathbf{T}} \sum_{k=1}^{\mathbf{T}} (Tr(\Sigma) + x_k^T W^T W x_k)$ with $\hat{rs} = \frac{1}{\mathbf{T}} \sum_{k=1}^{\mathbf{T}} r_k^T r_k$. Then the projected gradient descent on ζ performed every $\mathbf{T}$ samples is given as

$$\zeta \leftarrow \mathcal{P}_{\mathbb{R}_+}[\zeta + \eta_k(\hat{rs} - M)]. \tag{A.13}$$

The dynamics of the synaptic weights W, U and multiplier ζ are evolving on two different timescales. In fact, every time a sample x_k is given, the synaptic weights of w_i and u_j are updated since only local information with a globally known ζ is needed. On the other hand, only after a long period (say, $\mathbf{T}$) of samples presented is the multiplier ζ updated based on the encoder's neural activity history $\hat{sr}$. From a biological standpoint, the dynamics of ζ may be interpreted as a global neuromodulator that aggregates network activity into a widespread gating effect on individual neurons.

A.3 Derivation of Learning Rule for Bernoulli Spiking Model. In this section, we give the derivation for algorithm 4. Denote $\sigma_i(x) = \frac{1}{1+\exp^{-w_i^T x}}$. Then for the Bernoulli encoder, equation 5.1, we have

$$E(r_i|x; w_i) = 2\sigma_i(x) - 1, \quad \mathrm{Var}(r_i|x; w_i) = 4\sigma_i(x)(1 - \sigma_i(x)). \tag{A.14}$$

We denote $E(r|x_k) = (E(r_1|x_k; w_1), \ldots, E(r_m|x_k; w_m))^T$, and denote $Diag\{\mathrm{Var}(r_i|x_k)\}$ as a diagonal matrix with its diagonal element as $\mathrm{Var}(r_i|x_k; w_i)$. We also know that

$$\frac{d}{dz} \frac{1}{1 + \exp^{-z}} = \frac{1}{1 + \exp^{-z}} * \left(1 - \frac{1}{1 + \exp^{-z}}\right). \tag{A.15}$$

According to section 2.2, we need to calculate the approximated gradient of

$$L_k(\theta, U) = \sum_{r \in \Delta} P(r|x_k; \theta) \log \mathcal{N}(Ur, \Lambda) \tag{A.16}$$

with respect to $\theta = \{w_1, \ldots, w_m\}$ and U for a given sample x_k, since we adopt a linear gaussian decoder $\mathcal{N}(Ur, \Lambda)$. Similar to equation A.1,

$$\begin{aligned} L_k(\theta, U) &= \sum_{r \in \Delta^m} P(r|x_k; \theta_k) \log \frac{1}{\sqrt{2\pi |\det \Lambda|}} \\ &\quad \times \exp\left\{-\frac{1}{2} x_k^T \Lambda^{-1} x_k + x_k^T \Lambda^{-1} U r - \frac{1}{2} r^T U^T \Lambda^{-1} U r\right\} \\ &= constant + E_r\left[x_k^T \Lambda^{-1} U r - \frac{1}{2} r^T U^T \Lambda^{-1} U r\right] \\ &= constant + x_k^T \Lambda^{-1} U E(r|x_k) - \frac{1}{2} E^T(r|x_k; \theta) U^T \Lambda^{-1} U E(r|x_k) \\ &\quad -\frac{1}{2} Tr(U^T \Lambda^{-1} U Diag\{\mathrm{Var}(r_i|x_k)\}). \end{aligned} \tag{A.17}$$

Then the gradient of $L_k(\theta, U)$ with respect to w_i is

$$\begin{aligned} \nabla_{w_i} L_k(\theta, U) &= 2[U^T \Lambda^{-1} x_k]_i \sigma_1(x_k)(1 - \sigma_i(x_k)) x_k \\ &\quad -2[U^T \Lambda^{-1} U E(r|x_k)]_i \sigma_i(x_k)(1 - \sigma_i(x_k)) x_k \\ &\quad -2[U^T \Lambda^{-1} U]_{ii}(1 - 2\sigma_i(x_k))\sigma_i(x_k)(1 - \sigma_i(x_k)) x_k \end{aligned} \tag{A.18}$$

and the gradients of $L_k(\theta, U)$ with respect to U are

$$\begin{aligned} \nabla_U L_k(U, \Lambda) &= \Lambda^{-1}[x_k E^T(r|x_k) \\ &\quad - U Diag\{\mathrm{Var}(r_i|x_k)\} - U E(r|x_k) E^T(r|x_k)]. \end{aligned} \tag{A.19}$$

Finally, we discuss how to implement equations A.19 and A.18 with locally available samples. The key idea is to replace $UE(r|x_k)$ in equations A.19 and A.18 with a sample-based reconstruction $\hat{x}_k$. Given a stimulus x_k, we first get a sample r_k from encoder $P(r|x_k; \theta)$, and then we can sample the decoder $\mathcal{N}(Ur_k, \Lambda)$ to get a reconstruction $\hat{x}_k$. This sample procedure is repeated $\mathbf{T}$ times, yielding samples $\{r_k^l = (r_{1,k}^l, \ldots, r_{m,k}^l), x_k^l, l = 1, \ldots, \mathbf{T}\}$. The sample-based estimation of $UE(r|x_k)$ is $\hat{x}_k = \frac{1}{\mathbf{T}} \hat{x}_k^l$. Even though $\sigma_i(x_k)$ could be calculated by each encoder neuron, we still use a sample-based estimation to approximate $E(r|x_k; \theta)$ and $Diag\{\mathrm{Var}(r_i|x_k; w_i)\}$. This is because both Hebbian and spike timing dependent plasticity neural learning rules suggest that the synaptic weight adapts according to neural activities, while

here, samples can be regarded as neural activities. Moreover, $\sigma_i(x_k)$ cannot be known by the decoder neuron since it does not know the encoder synaptic weight w_i. We approximate $\sigma_i(x_k)$ as $\hat{\sigma}_i(x_k) = \frac{1}{2\mathbf{T}} \sum_{l=1}^{\mathbf{T}} (r_{i,k}^l + 1)$. Notice that $\hat{\sigma}_i(x_k)$ can be calculated by the decoder neuron since it accesses samples from encoder neurons. This leads to the approximated $\hat{E}(r_i|x_k; w_i)$, $\hat{E}(r|x_k)$ and $\hat{\mathrm{Var}}(r_i|x_k; w_i)$ according to equation A.14.

Then the gradient of $L_k(\theta, U)$ with respect to w_i and U is

$$\nabla_{w_i} L_k(\theta, U) \approx \frac{1}{2}\{[U^T \Lambda^{-1}(x_k - \hat{x}_k)]_i + [U^T \Lambda^{-1} U]_{ii} \hat{E}(r_i|x_k; w_i)\} \hat{\mathrm{Var}}(r_i|x_k; w_i) x_k, \tag{A.20}$$

$$\nabla_U L_k(\theta, U) \approx \Lambda^{-1}[(x_k - \hat{x}_k)\hat{E}^T(r|x_k) - U Diag\{\hat{\mathrm{Var}}(r_i|x_k)\}]. \tag{A.21}$$

We write equations A.20 and A.21 componentwise according to the synaptic weights of each encoder/decoder neuron and get the learning rules in algorithm 4.

A.4 Detailed Comparison between VAE and Infomax Learning Objectives, Optimization Strategies, and Interpretation.

A.4.1 Objective functions. Both variational infomax and VAE adopt the variational Bayesian inference techniques to derive tractable objective functions. However, their objective functions are fundamentally different.

Suppose we have a batch of sampled data $\{x_1, \ldots, x_N\}$. With the variational infomax, the objective function is

$$f_{vmi}(\theta, \phi) = \frac{1}{N} \sum_{k=1}^{N} \int_r dr p(r|x_k; \theta) \log q(x_k|r; \phi). \tag{A.22}$$

Here $q(x_k|r; \phi)$ is the variational approximation for the Bayesian posterior, which could be treated as the generative distribution in VAE.

On the other hand, by introducing a variational distribution $p(r|x_k; \theta)$, the objective function used in VAEs is

$$f_{vae}(\theta, \phi) = \frac{1}{N} \sum_{k=1}^{N} \int_r dr p(r|x_k; \theta)[-\log p(r|x_k; \theta) + \log q(x_k|r; \phi) q(r)] \tag{A.23}$$

$$= f_{vmi}(\theta, \phi) - \frac{1}{N} \sum_{k=1}^{N} KL(p(r|x_k; \theta), q(r)), \tag{A.24}$$

where $q(r)$ is a fixed prior distribution that is usually taken as a normal distribution $\mathcal{N}(0, I)$, and $p(r|x_k; \theta)$ is the channel conditional distribution in infomax. Therefore, the variational infomax objective function $f_{vmi}(\theta, \phi)$ is different from the VAE objective function $f_{vae}(\theta, \phi)$, since the latter has an additional KL divergence term.

A.4.2 Infomax with an Energy Constraint. The variational infomax with an energy constraint is also different from the VAE with a gaussian prior. The energy constraint $E_r(r^T r) < M$ can be written as

$$\frac{1}{N}\sum_{k=1}^{N}\int_r drp(r|x_k;\theta)rr^T \leq M.$$

Then the variational infomax with an energy constraint is leading to

$$\max_{\theta;\phi} \; f_{vmi}(\theta,\phi), \; s.t., \; \frac{1}{N}\sum_{k=1}^{N}\int_r drp(r|x_k;\theta)rr^T \leq M \tag{A.25}$$

(note that this equation is the same as equation A.6).

Now, for VAEs, with a gaussian prior $q(r) = \mathcal{N}(0, I)$, the KL term $-\frac{1}{N}\sum_{k=1}^{N} KL(p(r|x_k;\theta), q(r))$ in equation A.24 is

$$\begin{aligned} &-\frac{1}{N}\sum_{k=1}^{N}\int_r drp(r|x_k;\theta)\log(p(r|x_k;\theta)) \\ &-\frac{1}{2N}\sum_{k=1}^{N}\int_r drp(r|x_k;\theta)rr^T + constant. \end{aligned} \tag{A.26}$$

Therefore, the VAE problem becomes

$$\begin{aligned} &\max_{\theta;\phi} f_{vae}(\theta,\phi), \\ &f_{vae}(\theta,\phi) = f_{vmi}(\theta,\phi) - \frac{1}{N}\sum_{k=1}^{N}\int_r drp(r|x_k;\theta)\log(p(r|x_k;\theta)) \\ &-\frac{1}{2N}\sum_{k=1}^{N}\int_r drp(r|x_k;\theta)rr^T. \end{aligned} \tag{A.27}$$

The optimization problem, (equations A.25 and A.27), are obviously different, noting that the objective function of VAE also penalizes the entropy term $-\sum_{k=1}^{N}\int_r drp(r|x_k;\theta)\log(p(r|x_k;\theta))$.

A.4.3 Biological Implication of Three-Term Modulatory Processes for the Weight Updates. Even though the term $\sum_{k=1}^{N} \int_r drp(r|x_k;\theta)rr^T$ appears in both equations A.27 and A.25, its treatment differs. We develop a Lagrangian saddle point optimization with a multiplier ζ that produces

$$\min_{\zeta} \max_{\theta;\phi} \; f_{vmi}(\theta,\phi) - \zeta \frac{1}{N} \sum_{k=1}^{N} \int_r drp(r|x_k;\theta)rr^T + \zeta M \tag{A.28}$$

Hence, the weight ζ before $\sum_{k=1}^{N} \int_r drp(r|x_k;\theta)rr^T$ can adapt according to the neural responses and the available energy budget, while VAE (even β-VAE in Higgins et al., 2017) uses a fixed weight. This in turn leads to nontrivial biological interpretations, as discussed in section 6.

Acknowledgments

S. Ching holds a Career Award at the Scientific Interface from the Burroughs-Wellcome Fund. This work was partially supported by AFOSR 15RT0189, NSF 1537015, and NSF 1653589 from the U.S. Air Force Office of Scientific Research and the U.S. National Science Foundation, respectively.

References

Arrow, K. J., Hurwicz, L., & Uzawa, H. (1958). *Studies in linear and nonlinear programming*. Palo Alto, CA: Stanford University Press.

Attneave, F. (1954). Some informational aspects of visual perception. *Psychological Review*, *61*(3), 183–193.

Balasubramanian, V., Kimber, D., & Berry, M. J. (2001). Metabolically efficient information processing. *Neural Computation*, *13*(4), 799–815.

Barber, D., & Agakov, F. (2003). The IM algorithm: A variational approach to information maximization. In S. Thrun, L. K. Saul, & B. Schölkopf (Eds.), *Advances in neural information processing systems*, *16* (pp. 201–208). Cambridge, MA: MIT Press.

Barlow, Horace B. (1961). Possible principles underlying the transformations of sensory messages, In W. Rosenblith (Ed.), *Sensory communication* (pp. 217–234). Cambridge, MA: MIT Press.

Bell, A. J., & Sejnowski, T. J. (1995). An information-maximization approach to blind separation and blind deconvolution. *Neural Computation*, *7*(6), 1129–1159.

Bertsekas, D. P. (2011). Incremental gradient, subgradient, and proximal methods for convex optimization: A survey. *Optimization for Machine Learning*, *2010*(1–38), 3.

Blei, D. M., Kucukelbir, A., & McAuliffe, J. D. (2017). Variational inference: A review for statisticians. *Journal of the American Statistical Association*, *112*(518), 859–877.

Brette, R. (2017). *Is coding a relevant metaphor for the brain*? bioRxiv:168237.

Brunel, N., & Nadal, J. P. (1998). Mutual information, Fisher information, and population coding. *Neural Computation*, *10*(7), 1731–1757.

Chalk, M., Marre, O., & Tkacik, G. (2018). Toward a unified theory of efficient, predictive, and sparse coding. *PNAS*, *115*(1), 186–191.

Doya, K. (Ed.). (2007). *Bayesian brain: Probabilistic approaches to neural coding*. Cambridge, MA: MIT Press.

Fairhall, A. L., Lewen, G. D., Bialek, W., & van Steveninck, R. R. D. R. (2001). Efficiency and ambiguity in an adaptive neural code. *Nature*, *412*(6849), 787.

Friston, K. (2010). The free-energy principle: A unified brain theory? *Nature Reviews Neuroscience*, *11*(2), 127–138.

Friston, K. (2012). The history of the future of the Bayesian brain. *NeuroImage*, *62*(2), 1230–1233.

Gershman, S. J., Horvitz, E. J., & Tenenbaum, J. B. (2015). Computational rationality: A converging paradigm for intelligence in brains, minds, and machines. *Science*, *349*(6245), 273–278.

Gilbert, C. D., & Li, W. (2013). Top-down influences on visual processing. *Nature Reviews Neuroscience*, *14*(5), 350–363.

Gjorgjieva, J., Sompolinsky, H., & Meister, M. (2014). Benefits of pathway splitting in sensory coding. *Journal of Neuroscience*, *34*(36), 12127–12144.

Grossberg, S. (1987). Competitive learning: From interactive activation to adaptive resonance. *Cognitive Science*, *11*(1), 23–63.

Higgins, I., Matthey, L., Pal, A., Burgess, C., Glorot, X., Botvinick, M., . . . Lerchner, A. (2017). Beta-vae: Learning basic visual concepts with a constrained variational framework. In *Proceedings of the International Conference on Learning Representations*.

Hinton, G. E., Dayan, P., Frey, B. J., & Neal, R. M. (1995). The "wake-sleep" algorithm for unsupervised neural networks. *Science*, *268*, 1158–1161.

Huang, W., & Zhang, K. (2018). Information-theoretic bounds and approximations in neural population coding. *Neural Computation*, *30*(4), 885–944.

Hyvarinen, A. Hurri, J., & Hoyer, P. O. (2009). *Natural image statistics: A probabilistic approach to early computational vision*. New York: Springer Science & Business Media.

Karklin, Y., & Simoncelli, E. P. (2011). Efficient coding of natural images with a population of noisy linear-nonlinear neurons. In J. Shawe-Taylor, R. S. Zemel, & P. L. Bartlett (Eds.), *Advances in neural information processing systems*, *24* (pp. 999–1007). Red Hook, NY: Curran.

Kingma, D. P., & Max, W. (2013). Auto-encoding variational Bayes. arXiv:1312.6114.

Laughlin, S. B. (2001). Energy as a constraint on the coding and processing of sensory information. *Current Opinion in Neurobiology*, *11*(4), 475–480.

Levy, W. B., & Baxter, R. A. (1996). Energy efficient neural codes. *Neural Computation*, *8*(3), 531–543.

Lillicrap, T. P., Cownden, D., Tweed, D. B., & Akerman, C. J. (2016). Random synaptic feedback weights support error backpropagation for deep learning. *Nature Communications*, *7*, 13276.

Linsker, R. (1988). Self-organization in a perceptual network. *Computer*, *21*(3), 105–117.

Linsker, R. (1989). An application of the principle of maximum information preservation to linear systems. In D. S. Touretzky (Ed.), *Advances in neural information processing systems*, *2* (pp. 186–194). San Mateo, CA: Morgan Kaufmann.

Linsker, R. (1992). Local synaptic learning rules suffice to maximize mutual information in a linear network. *Neural Computation*, *4*(5), 691–702.

Liu, S., & Ching, S. (2017). Recurrent information optimization with local, metaplastic synaptic dynamics. *Neural Computation*, *29*(9), 2528–2552.

Meyer, A. F., Williamson, R. S., Linden, J. F., & Sahani, M. (2017). Models of neuronal stimulus-response functions: Elaboration, estimation, and evaluation. *Frontiers in Systems Neuroscience*, *10*, 109.

Park, M., & Pillow J. W. (2017). *Bayesian efficient coding*. bioRxiv:178418.

Pavel, S., & Miller., P. (2015). Spiking neuron network Helmholtz machine. *Frontiers in Computational Neuroscience*, *9*, 46.

Pillow, J. W., Shlens, J., Paninski, L., Sher, A., Litke, A. M., Chichilnisky, E. J., & Simoncelli, E. P. (2008). Spatio-temporal correlations and visual signalling in a complete neuronal population. *Nature*, *454*, 995–999.

Pitkow, X., & Meister, M. (2012). Decorrelation and efficient coding by retinal ganglion cells. *Nature Neuroscience*, *15*(4), 628.

Rezende, D. J., Wierstra, D., & Gerstner, W. (2011). Variational learning for recurrent spiking networks. In J. Shawe-Taylor, R. S. Zemel, & P. L. Bartlett (Eds.), *Advances in neural information processing systems*, *24* (pp. 136–144). Red Hook, NY: Curran.

Sanborn, A. N., & Chater, N. (2016). Bayesian brains without probabilities. *Trends in Cognitive Sciences*, *20*(12), 883–893.

Shakir, M., & Rezende, D. J. (2015). Variational information maximisation for intrinsically motivated reinforcement learning. In C. Cortes, N. D. Lawrance, D. D. Lee, M. Sugiyama, & R. Garnett (Eds.), *Advances in neural information processing systems*, *28* (pp. 2125–2133). Red Hook, NY: Curran.

Tanaka, T., Kaneko, T., & Aoyagi, T. (2009). Recurrent infomax generates cell assemblies, neuronal avalanches, and simple cell-like selectivity. *Neural Computation*, *21*(4), 1038–1067.

Tkačik, G., Prentice, J. S., Balasubramanian, V., & Schneidman, E. (2010). Optimal population coding by noisy spiking neurons. *Proceedings of the National Academy of Sciences*, *107*(32), 14419–14424.

Wang, Z., Wei, X. X., Stocker, A. A., & Lee, D. D. (2016). Efficient neural codes under metabolic constraints. In *Advances in neural information processing systems* (pp. 4619–4627). Red Hook, NY: Curran.

Wei, X. X., & Stocker, A. A. (2016). Mutual information, Fisher information, and efficient coding. *Neural Computation*, *28*(2), 305–326.

Received June 5, 2018; accepted January 7, 2019.