AN INFORMATION-THEORETIC FRAMEWORK FOR FAST AND ROBUST UNSUPERVISED LEARNING VIA NEURAL POPULATION INFOMAX

Wentao Huang & Kechen Zhang
Department of Biomedical Engineering
Johns Hopkins University School of Medicine
Baltimore, MD 21205, USA
{whuang21,kzhang4}@jhmi.edu

ABSTRACT

A framework is presented for unsupervised learning of representations based on infomax principle for large-scale neural populations. We use an asymptotic approximation to the Shannon’s mutual information for a large neural population to demonstrate that a good initial approximation to the global information-theoretic optimum can be obtained by a hierarchical infomax method. From the initial solution, an efficient algorithm based on gradient descent of the final objective function is proposed to learn representations from the input datasets, allowing complete, overcomplete, or undercomplete bases. As confirmed by numerical experiments, our method is robust and highly efficient for extracting salient features from image datasets. Compared with the main existing methods, our algorithm has a distinct advantage in both the training speed and the robustness of unsupervised representation learning.

1 INTRODUCTION

How to discover the unknown structures in data is a key task for machine learning. Learning good representations from observed data is important because a clearer description helps reveal the underlying structures. Representation learning has drawn considerable attention in recent years (Bengio et al., 2013). One category of algorithms for unsupervised learning of representations is based on probabilistic models (Lewicki & Sejnowski, 2000; Hinton, 2002; Hinton & Salakhutdinov, 2006; Lee et al., 2008), e.g. maximum likelihood (ML) estimation, maximum a posteriori (MAP) probability estimation, etc. Another category of algorithms is based on reconstruction error or generative criterion (Olshausen & Field, 1996; Aharon et al., 2006; Ranzato et al., 2007; Vincent et al., 2010; Mairal et al., 2009; 2010; Goodfellow et al., 2014), which usually uses the squared error and additional constraints for the objective functions. Sometimes the reconstruction error or generative criterion can be explained also by probabilistic models (Olshausen & Field, 1997; Vincent et al., 2010).

Shannon’s information theory is a powerful tool for description of stochastic systems and could be utilized to provide a characterization for good representations (Vincent et al., 2010). However, computational difficulties associated with Shannon’s mutual information (MI) (Shannon, 1948) have hindered its wider applications. The Monte Carlo (MC) sampling (MacKay, 2003; Yarrow et al., 2012) is a convergent method for estimating MI with arbitrary accuracy, but its computational inefficiency makes it unsuitable for difficult optimization problems especially in the cases of high-dimensional input stimuli and large population networks. Bell and Sejnowski (Bell & Sejnowski, 1995) have directly applied the infomax (Linsker, 1988) approach to independent component analysis (ICA) of data with independent non-Gaussian components, assuming additive noise, but their method requires that the number of outputs be equal to the number of inputs. The extensions of ICA that allows overcomplete bases incurs increased algorithm complexity and difficulty in learning of parameters (Lewicki & Sejnowski, 2000; Kreutz-Delgado et al., 2003; Hyvärinen, 2005; Le et al., 2011).
Since Shannon MI is closely related to ML and MAP (Clarke & Barron, 1990; Lee et al., 2000; Huang & Zhang, 2016), the algorithms of representation learning based on probabilistic models should be amenable to information-theoretic treatment. Representation learning based on reconstruction error could also be accommodated by information theory, because the inverse of Fisher information (FI) is the Cramér-Rao lower bound on the mean square decoding error obtained with any unbiased decoder (Rao, 1945). Hence minimizing the reconstruction error potentially maximizes a lower bound on the MI (Vincent et al., 2010).

Related problems also arise in neuroscience. It has long been suggested that the real nervous systems might approach an information-theoretic optimum for neural coding and computation (Barlow, 1961; Atick, 1992; Borst & Theunissen, 1999; Pouget et al., 2000). However, in the cerebral cortex, the number of neurons is huge, with about $10^7$ neurons under a square millimeter of cortical surface (Rockel et al., 1980; Carlo & Stevens, 2013). It has often been computationally intractable to precisely characterize the information coding and processing for large-scale populations.

To address all these issues, we present a framework for unsupervised learning of representations in a large-scale nonlinear feedforward model based on infomax principle with realistic biological constraints such as neuron models with Poisson noise. First we adopt an objective function based on an asymptotic formula in the large population limit for the MI between the stimuli and the neural population responses (Huang & Zhang, 2016). Since the objective function is usually nonconvex, choosing a good initial value is very important for its optimization. Starting from the initial value, we use a hierarchical infomax approach to quickly find a tentative global optimal solution for each layer by analytic methods. Finally, a fast convergence learning rule is presented for optimizing the final objective functions based on the tentative optimal solution. Our algorithm is robust and can learn complete, overcomplete or undercomplete basis vectors quickly from different image datasets. Experimental results showed that the convergence rate of our method was significantly faster than other existing methods, often by an order of magnitude. More importantly, the number of output units processed by our method can be very large, much larger than the number of inputs. As far as we know, no existing model can easily deal with this situation.

2 METHODS

Suppose the input $x$ is a $K$-dimensional vector, $x = (x_1, \ldots, x_K)^T$, the outputs of $N$ neurons are denoted by a vector, $r = (r_1, \ldots, r_N)^T$. In this paper we denote random variables by upper case letters, e.g., random variables $X$ and $R$, in contrast to their vector values $x$ and $r$. Then the MI between $X$ and $R$ is defined by (Cover & Thomas, 2006)

$$I(X; R) = \left\langle \ln \frac{p(x|r)}{p(x)} \right\rangle_{r,x}, \quad (2.1)$$

where $\langle \cdot \rangle_{r,x}$ denotes the expectation with respect to the probability density function (PDF) $p(r \mid x)$. A link between MI and FI was derived by some researchers (Brunel & Nadal, 1998). If the PDF $p(x)$ and $p(r \mid x)$ are twice continuously differentiable for almost every $x \in \mathbb{R}^K$, we can prove the following more general asymptotic approximation for large $N$ (e.g. $N \gg K$, Huang & Zhang, 2016).

$$I(X; R) \approx I_G = \frac{1}{2} \left\langle \ln \left( \det \left( \frac{G(x)}{2\pi e} \right) \right) \right\rangle_x + H(X), \quad (2.2)$$

where $\det (\cdot)$ denotes the matrix determinant and $H(X) = -\langle \ln p(x) \rangle_x$ is the entropy,

$$G(x) = J(x) + P(x), \quad (2.3)$$

$$J(x) = -\left\langle \frac{\partial^2 \ln p(r \mid x)}{\partial x \partial x^T} \right\rangle_{r|x}, \quad (2.4)$$

$$P(x) = -\frac{\partial^2 \ln p(x)}{\partial x \partial x^T}. \quad (2.5)$$

From (2.1) and (2.2) we get the conditional entropy

$$H(X|R) = -\left\langle \ln p(x|r) \right\rangle_{r,x} \approx -\frac{1}{2} \left\langle \ln \left( \det \left( \frac{G(x)}{2\pi e} \right) \right) \right\rangle_x. \quad (2.6)$$
Here the Fisher information matrix $J(x)$ is symmetric positive semidefinite and can be written also as

$$J(x) = \left\langle \frac{\partial \ln p(r|x)}{\partial x} \frac{\partial \ln p(r|x)}{\partial x^T} \right\rangle_{r|x}. \quad (2.7)$$

Suppose $p(r|x)$ is conditional independent, $p(r|x) = \prod_{n=1}^{N} p(r_n|x; \theta_n)$, where $\theta_n$ denotes a vector of parameters for the $n$-th neuron. Then we get (see Huang & Zhang [2016])

$$J(x) = N \int p(\theta) S(x; \theta) d\theta, \quad (2.8)$$

$$S(x; \theta) = \left\langle \frac{\partial \ln p(r|x; \theta)}{\partial x} \frac{\partial \ln p(r|x; \theta)}{\partial x^T} \right\rangle_{r|x}, \quad (2.9)$$

where $p(\theta)$ is the population density function of parameter $\theta$,

$$p(\theta) = \frac{1}{N} \sum_{n=1}^{N} \delta(\theta - \theta_n), \quad (2.10)$$

and $\delta(\cdot)$ denotes the Dirac delta function. It can be proved that the approximation function of MI $I_G\{p(\theta)\}$ (Eq. 2.2) is concave about $p(\theta)$ (Huang & Zhang [2016]). In Eq. (2.8), we can approximate the continuous integral by a discrete summation for numerical computation,

$$J(x) = N \sum_{k=1}^{K_1} \alpha_k S(x; \theta_k), \quad (2.11)$$

where the integer $K_1 \in \{1, \ldots, N\}$, $\sum_{k=1}^{K_1} \alpha_k = 1$, $\alpha_k > 0$, $k = 1, \ldots, K_1$. We know that the cerebral cortex usually forms functional column structures and each column is composed of neurons with the same properties (Hubel & Wiesel [1962]). Hence the positive integer $K_1$ can be regarded as the number of classes in the neural population. Poisson noise model and Gaussian noise model are detailed in Appendix.

### 2.1 The Objective Function

Given the tuning curve $f(x; \theta_n)$, our goal is to find the optimal population distribution density of parameter vector $\theta_n$ from which we can maximize the MI between the stimulus $x$ and response $r$. By Eq. (2.7), our optimization problem becomes:

$$\text{maximize } I_G\{\alpha_k\} = \frac{1}{2} \left\langle \ln \left( \frac{\det \left( G(x) \right)}{2\pi e} \right) \right\rangle_x + H(X) \quad (2.12)$$

or equivalently

$$\text{minimize } Q_G\{\alpha_k\} = -\frac{1}{2} \left\langle \ln \left( \det \left( G(x) \right) \right) \right\rangle_x, \quad (2.13)$$

and

$$\text{subject to } \sum_{k=1}^{K_1} \alpha_k = 1, \alpha_k > 0, \forall k = 1, \ldots, K_1, \quad (2.14)$$

where $G(x)$ is given by Eq. (2.3)–(2.5) and (2.11).

Sometimes we do not know the specific form of $p(x)$ and only know $M$ samples, $x_1, \ldots, x_M$, which are independent and identically distributed (i.i.d.) samples drawn from the distribution $p(x)$, then we can use the empirical average to approximate the integral in Eq. (2.13),

$$Q_G\{\alpha_k\} \approx -\frac{1}{2} \sum_{m=1}^{M} \ln \left( \det \left( G(x_m) \right) \right). \quad (2.15)$$

Since $Q_G\{\alpha_k\}$ is a convex function of $\{\alpha_k\}$ (Huang & Zhang [2016]), we can directly use efficient numerical methods to solve the optimal solution for small $K$. However, for large $K$, finding optimal solution by numerical methods becomes intractable. In the following we will propose another way to this problem. Instead of directly solving its density distribution $\{\alpha_k\}$, we optimize the parameters $\{\alpha_k\}$ and $\{\theta_k\}$ simultaneously under a framework of hierarchical infomax.
2.2 Hierarchical Infomax

For clarity, here we consider the Poisson neuron model and suppose the tuning curve $f(x; \theta_n)$ is a nonlinear function, although our method is easily applicable to other neuron models and different tuning functions, e.g. sigmoid and rectified linear unit (ReLU) (Nair & Hinton, 2010; Glorot et al., 2011).

Let us consider the nonlinear function for the $n$-th neuron has the form, $f(x; \theta_n) = \tilde{f}(y_n; \tilde{\theta}_n)$, $n = 1, \cdots, N$, where $\tilde{f}(y_n; \tilde{\theta}_n)$ is a nonlinear function, $\theta_n = (w_n^T, \tilde{\theta}_n^T)^T$ and $\tilde{\theta}_n$ are the parameter vectors, and $w_n$ is a $K$-dimensional weights vector,

$$y_n = w_n^T x.$$  

(2.16)

In general, it is very difficult to find the optimal parameters, $\theta_n$, $n = 1, \cdots, N$ for the following reasons. First, the number of output neurons $N$ is very large, usually $N \gg K$. Second, the tuning curve $f(x; \theta_n)$ is a nonlinear function, which usually leads to a nonconvex optimization problem. For nonconvex optimization problems, the selection of initial values often has a great influence on the final optimization results. Our approach meets these challenges by making better use of the large number of neurons and by finding good initial values by hierarchical infomax.

We divide the nonlinear transformation into two stages, mapping first from $x$ to $y_n$, and then from $y_n$ to $\tilde{f}(y_n; \tilde{\theta}_n)$, where $y_n$ can be regarded as the membrane potential of the $n$-th neuron, and $\tilde{f}(y_n; \tilde{\theta}_n)$ its firing rate. As with the real neurons, we assume the membrane potential is corrupted by noise:

$$\bar{Y}_n = Y_n + Z_n,$$  

(2.17)

where $Z_n \sim \mathcal{N}(0, \sigma^2)$ is a normal distribution with mean 0 and variance $\sigma^2$. Then the mean membrane potential of the $k$-th class subpopulation with $N_k = N\alpha_k$ neurons is given by

$$\bar{Y}_k = \frac{1}{N_k} \sum_{n=1}^{N_k} \bar{Y}_n = Y_k + \bar{Z}_k, k = 1, \cdots, K_1,$$  

(2.18)

$$\bar{Z}_k \sim \mathcal{N}(0, N_k^{-1} \sigma^2).$$  

(2.19)

Define vectors $\bar{y} = (\bar{y}_1, \cdots, \bar{y}_N)^T$, $\bar{y} = (\bar{y}_1, \cdots, \bar{y}_{K_1})^T$ and $y = (y_1, \cdots, y_{K_1})^T$, where $y_k = w_k^T x (k = 1, \cdots, K_1)$. Notice that $\bar{y}_n (n = 1, \cdots, N)$ is also divided into $K_1$ classes, the same as for $r_n$. If we assume $f(x; \theta_k) = \tilde{f}(\bar{y}_k; \tilde{\theta}_k)$, i.e. assuming an additive Gaussian noise for $y_n$ (see Eq. 2.18), then the random variables $X$, $Y$, $\bar{Y}$, $\tilde{Y}$ and $R$ form a Markov chain, denoted by $X \rightarrow Y \rightarrow \bar{Y} \rightarrow \tilde{Y} \rightarrow R$, which is demonstrated in Figure 1 and we have the following proposition (the proof is given in Appendix).

**Proposition 2.1.** With the random variables $X$, $Y$, $\bar{Y}$, $\tilde{Y}$, $R$ and Markov chain $X \rightarrow Y \rightarrow \bar{Y} \rightarrow \tilde{Y} \rightarrow R$, the following equations hold,

$$I(X; R) = I(Y; R) \leq I(\bar{Y}; R),$$  

(2.20)

$$I(X; R) \leq I(X; \bar{Y}) = I(X; \tilde{Y}),$$  

(2.21)

and for large $N_k (k = 1, \cdots, K_1)$,

$$I(Y; R) \simeq I(\bar{Y}; R).$$  

(2.22)

A major advantage of incorporating membrane noise is that it facilitates finding the optimal solution by using the infomax principle. When the noise variance $\sigma^2 \rightarrow 0$, we have $\bar{Y}_k \rightarrow Y_k$, $\tilde{f}(\bar{y}_k; \tilde{\theta}_k) \simeq \tilde{f}(\bar{y}_k; \tilde{\theta}_k) = f(x; \theta_k)$ and $I(\bar{Y}; R) \simeq I(Y; R) = I(X; R)$.

Hence, from the above we know that maximizing $I(X; R)$ will result in maximizing $I(Y; R)$ and maximizing $I(X; \bar{Y})$. Therefore the optimal solution of $\max I(X; \bar{Y})$ and $\max I(Y; R)$ will give a good initial approximation that will be very close to the optimal solution of $\max I(X; R)$.

Similarly, we can extend this method to multilayer neural population networks. For example, the two layers neural populations networks with outputs $R^{(1)}$ and $R^{(2)}$ form a Markov chain, $X \rightarrow$
$\hat{R}^{(1)} \rightarrow R^{(1)} \rightarrow \hat{R}^{(1)} \rightarrow R^{(2)}$, where the random variable $\hat{R}^{(1)}$ is similar to $Y$, the random variable $R^{(1)}$ is similar to $\hat{Y}$ and $R^{(1)}$ is similar to $Y$ in the above. Then we can prove that the optimal solution of $\max I(X; R^{(2)})$ approximates the solutions of $\max I(X; R^{(1)})$ and $\max I(\hat{R}^{(1)}; R^{(2)})$, where $I(\hat{R}^{(1)}; R^{(2)}) \simeq I(R^{(1)}; R^{(2)})$.

More generally, consider a highly nonlinear feedforward neural network that maps the input $x$ to output $z$, with $z = F(x; \theta) = h_L \circ \cdots \circ h_1(x)$, where $h_l$ ($l = 1, \cdots, L$) is a linear or nonlinear function (Montufar et al., 2014). We aim to find the optimal parameter $\theta$ by maximizing $I(X; Z)$. It is usually difficult to solve the optimization problem when there are many local extremums for $F(x; \theta)$. However, if each function $h_l$ is easy to optimize, then we can use the above method (i.e. hierarchical infomax for neural population) to get a good initial approximation to its global optimization solution, and go from there to find the final optimal solution. This information-theoretic consideration from the neural population coding point of view may help explain why deep structure networks with unsupervised pre-training have a powerful ability for learning representations.

In the following, we will discuss the optimization procedure in two stages, namely maximizing $I(X; \tilde{Y})$ and maximizing $I(Y; R)$.

### 2.2.1 The 1st Stage

In the first stage, our goal is to maximize the MI $I(X; \tilde{Y})$ and get the optimal parameters $w_n$ ($n = 1, \cdots, N$). Let us assume the stimulus $x$ has zero mean (if not, then let $x \leftarrow x - \langle x \rangle_x$) and covariance matrix $\Sigma_x$. It follows from eigendecomposition that

$$\Sigma_x = \langle xx^T \rangle_x \approx \frac{1}{M-1}XX^T = U\Sigma U^T,$$

(2.23)

where $X = [x_1, \cdots, x_M]$ (see 2.15), $U = [u_1, \cdots, u_K] \in \mathbb{R}^{K \times K}$ is an unitary orthogonal matrix and $\Sigma$ is a diagonal matrix with positive real numbers on the diagonal, $\Sigma = \text{diag} (\sigma_1^2, \cdots, \sigma_K^2)$, $\sigma_1 \geq \cdots \geq \sigma_K > 0$.

Define

$$\tilde{x} = \Sigma^{-1/2} U^T x,$$

(2.24)

$$\tilde{w}_k = \Sigma^{1/2} U^T w_k,$$

(2.25)

$$y_k = \tilde{w}_k^T \tilde{x},$$

(2.26)
where \( k = 1, \cdots, K_1 \). The covariance matrix of \( \tilde{x} \) is given by
\[
\Sigma_{\tilde{x}} = \left\langle \tilde{x}\tilde{x}^T \right\rangle_{\tilde{x}} \approx I_K, \quad (2.27)
\]
and \( I_K \) is a \( K \times K \) identity matrix, then from (2.2) and (A.5) we have that
\[
I(X; \tilde{Y}) = I(\tilde{X}; \tilde{Y})
\]
and
\[
I(\tilde{X}; \tilde{Y}) \approx I^G = \frac{1}{2} \ln \left( \frac{\det(\tilde{G})}{2\pi e} \right) + H(\tilde{X}), \quad (2.28)
\]
where
\[
\tilde{G} = N \sigma^{-2} \sum_{k=1}^{K_1} \alpha_k \tilde{w}_k \tilde{w}_k^T + I_K. \quad (2.29)
\]

Generally we can use the following approximation (see [Huang & Zhang, 2016]),
\[
p(\tilde{x}) \approx N(0, I_K), \quad (2.30)
\]
\[
P'(\tilde{x}) = -\frac{\partial^2 \ln p(\tilde{x})}{\partial \tilde{x} \partial \tilde{x}^T} \approx I_K. \quad (2.31)
\]

By the central limit theorem, the distribution of random variable \( \tilde{X} \) is closer to a normal distribution than the distribution of the original random variable \( X \). On the other hand, the PCA models assume multivariate gaussian data whereas the ICA models assume multivariate non-gaussian data. Hence by a PCA-like whitening transformation (2.24) we can use the approximation (2.31) with the Laplace’s method asymptotic expansion, which only requires that the peak be close to its mean and not that the random variable \( \tilde{X} \) be exactly Gaussian.

If there is no other constraints on the Gaussian channels of neural populations, especially the peak firing rate, then the capacity of this channel may grow indefinitely, i.e., \( I(\tilde{X}; \tilde{Y}) \to \infty \). The most common limitation on the neural populations is an energy or power constraint which can also be regarded as a limitation on the signal-to-noise ratio (SNR). The SNR for the output \( \tilde{y}_n \) of the \( n \)-th neuron is given by
\[
\text{SNR}_n = \frac{1}{\sigma^2} \left\langle \left( w_n^T x \right)^2 \right\rangle_x \approx \frac{1}{\sigma^2} \tilde{w}_n^T \tilde{w}_n, \quad n = 1, \cdots, N. \quad (2.32)
\]
We require that
\[
\frac{1}{N} \sum_{n=1}^{N} \text{SNR}_n \approx \frac{1}{\sigma^2} \sum_{k=1}^{K_1} \alpha_k \tilde{w}_k^T \tilde{w}_k \leq \rho, \quad (2.33)
\]
where \( \rho \) is a positive constant. Then by Eq. (2.28), (2.29) and (2.33), we have the following optimization problem:
\[
\text{minimize} \quad Q^G[\hat{W}] = -\frac{1}{2} \ln \left( \det \left( N \sigma^{-2} \hat{W} \hat{W}^T + I_K \right) \right), \quad (2.34)
\]
subject to
\[
h = \text{Tr} \left( \hat{W} \hat{W}^T \right) - E \leq 0, \quad (2.35)
\]
where \( \text{Tr} \cdot \) denotes matrix trace and
\[
\hat{W} = \hat{W} A^{1/2} = \Sigma^{1/2} U^T W A^{1/2} = [\hat{w}_1, \cdots, \hat{w}_{K_1}], \quad (2.36)
\]
\[
A = \text{diag}(\alpha_1, \cdots, \alpha_{K_1}), \quad (2.37)
\]
\[
W = [w_1, \cdots, w_{K_1}], \quad (2.38)
\]
\[
\tilde{W} = [\tilde{w}_1, \cdots, \tilde{w}_{K_1}], \quad (2.39)
\]
\[
E = \rho \sigma^2. \quad (2.40)
\]
Then we can get an optimal solution as following:
\[
W = a U_0 \Sigma_0^{-1/2} V_0^T, \quad (2.41)
\]
\[
A = K_1^{-1} I_{K_1}, \quad (2.42)
\]
\[
a = \sqrt{EK_1 K_0^{-1}}. \quad (2.43)
\]
where $K_0$ is a positive number, $0 < K_0 \leq K$, and represents the size of the reduced dimension,
\begin{align}
\Sigma_0 &= \text{diag} \left( \sigma_1^2, \ldots, \sigma_{K_0}^2 \right), \\
U_0 &= U \left( :, 1:K_0 \right) \in \mathbb{R}^{K \times K_0}, \\
V_0 &= V \left( :, 1:K_0 \right) \in \mathbb{R}^{K_1 \times K_0},
\end{align}
and $V = [v_1, \ldots, v_{K_1}]$ is an $K_1 \times K_1$ unitary orthogonal matrix.

In this case, the optimal parameters $w_n \ (n = 1, \ldots, N)$ cluster into $K_1$ classes (see Eq. 2.11) and obey an uniform discrete distribution (see also Eq. 2.57 in Section 2.2.2). Here notice that $K_0$ is an unknown parameter which will be determined below.

In the case of $K = K_0 = K_1$, the solution of $W$, i.e. Eq. (2.41), is whitening-like filters. When $V = I_K$, the optimal matrix $W$ is the principal component analysis (PCA) whitening filters. However, when $V = U$, the optimal matrix $W$ is symmetrical and is a zero component analysis (ZCA) whitening filter. If $K < K_1$, this case leads to an overcomplete solution, whereas when $K_0 \leq K_1 < K$, the undercomplete solution arises. Since $K_0 \leq K_1$ and $K_0 \leq K$, $Q_C^G$ achieves its minimum when $K_0 = K$. However, in practice other factors may limit its reaching this maximum. For example, consider the average of squared weights,
\begin{equation}
\varsigma = \sum_{k=1}^{K_1} \alpha_k \|w_k\|^2 = \text{Tr} \left( W A W^T \right) = \frac{E}{K_0} \sum_{k=1}^{K_0} \sigma_k^2,
\end{equation}
where $\|\cdot\|$ denotes the Frobenius norm. The value of $\varsigma$ is extremely large when any $\sigma_k$ becomes vanishingly small. For real neurons these weights of connection are not allowed to be too large. Hence we impose a limitation on the weights, e.g. $\varsigma \leq E_1$, where $E_1$ is a positive constant. This yields another constraint on the objective function,
\begin{equation}
\tilde{h} = \frac{E}{K_0} \sum_{k=1}^{K_0} \sigma_k^{-2} - E_1 \leq 0.
\end{equation}

In this case, we can choose $K_0$ by the following (see Appendix for details):
\begin{equation}
K_0 = \arg \min_{K_0} \left( \begin{array}{c} \sum_{k=1}^{K_0} \sigma_k^2 \geq \epsilon \\
\sum_{k=1}^{K_0} \sigma_k^{-2} \geq \epsilon \end{array} \right),
\end{equation}
where $\epsilon$ is a positive constant, $0 < \epsilon \leq 1$.

Moreover, another advantage of a low-rank matrix $W$ is that it can significantly reduce overfitting for learning neural population parameters. In practice, the constraint (consw) is equivalent to the regularization term called a weight-decay term in many other optimization problems [Hoeffding & Kennard 1970; Cortes & Vapnik 1995; Hinton 2010], which can reduce overfitting to the training data. To prevent neural networks from over fitting, Srivastava et al. (2014) presented a technique to randomly drop units from the neural network during training, which may in fact be regarded as an attempt to reduce the rank of the weight matrix.

In this stage, we get the optimal parameter $W$ (see 2.41), but the matrix $V_0$ is uncertain and we need to go further to determine its optimal value (see section 2.3).

2.2.2 The 2nd Stage

For this stage, our goal is to maximize the MI $I(Y; R)$ and get the optimal parameters $\tilde{\theta}_n \ (n = 1, \ldots, N)$. Here the input is $y = (y_1, \ldots, y_{K_1})^T$ and the output $r = (r_1, \ldots, r_N)^T$ is also clustered into $K_1$ classes. The responses of $N_k$ neurons in the $k$-th subpopulation obey a Poisson distribution with mean $\tilde{f}(e_k^T y; \tilde{\theta}_k)$, where $e_k$ is a unit vector with 1 in the $k$-th element and $y_k = e_k^T y$. By (2.24) and (2.26), we have that
\begin{align}
\langle y_k \rangle_{y_k} &= 0, \\
\sigma_{y_k}^2 &= \langle y_k^2 \rangle_{y_k} = \|\tilde{w}_k\|^2.
\end{align}
Then for large $N$ and by (2.2), (2.5) and (2.30) we can use the following approximation,

$$I(Y; R) \simeq I_F = \frac{1}{2} \left\langle \ln \left( \det \left( \frac{\mathbf{J}(\mathbf{y})}{2\pi e} \right) \right) \right\rangle_y + H(\mathbf{Y}),$$

(2.52)

where

$$\mathbf{J}(\mathbf{y}) = \text{diag} \left( N \alpha_1 |g_1'(y_1)|^2, \ldots, N \alpha_{K_1} |g_{K_1}'(y_{K_1})|^2 \right),$$

(2.53)

$$g_k'(y_k) = \frac{\partial g_k(y_k)}{\partial y_k}, \quad k = 1, \ldots, K_1,$$

(2.54)

$$g_k(y_k) = 2\sqrt{f(y_k; \tilde{\theta}_k)} = 1, \quad k = 1, \ldots, K_1.$$  

(2.55)

It is easy to get that

$$I_F = \frac{1}{2} \sum_{k=1}^{K_1} \left\langle \ln \left( \frac{N \alpha_k |g_k(y_k)|^2}{2\pi e} \right) \right\rangle_y + H(\mathbf{Y})$$

$$\leq \frac{1}{2} \sum_{k=1}^{K_1} \left\langle \ln \left( \frac{|g_k'(y_k)|^2}{2\pi e} \right) \right\rangle_y - \frac{K_1}{2} \ln \left( \frac{K_1}{N} \right) + H(\mathbf{Y}),$$

(2.56)

where the equality holds if and only if

$$\alpha_k = \frac{1}{K_1}, \quad k = 1, \ldots, K_1,$$

(2.57)

which is consistent with Eq. (2.42).

On the other hand, it follows from the Jensen’s inequality that

$$I_F = \left\langle \ln \left( p(\mathbf{y})^{-1} \det \left( \frac{\mathbf{J}(\mathbf{y})}{2\pi e} \right)^{1/2} \right) \right\rangle_y$$

$$\leq \ln \int \det \left( \frac{\mathbf{J}(\mathbf{y})}{2\pi e} \right)^{1/2} dy,$$

(2.58)

with equality if and only if $p(\mathbf{y})^{-1} \det \left( \mathbf{J}(\mathbf{y}) \right)^{1/2}$ is a constant, which implies that

$$p(\mathbf{y}) = \frac{\det \left( \frac{\mathbf{J}(\mathbf{y})}{2\pi e} \right)^{1/2}}{\int \det \left( \frac{\mathbf{J}(\mathbf{y})}{2\pi e} \right)^{1/2} dy} = \frac{\prod_{k=1}^{K_1} |g_k'(y_k)|}{\int \prod_{k=1}^{K_1} |g_k'(y_k)| dy}.$$  

(2.59)

From (2.58) and (2.59), maximizing $I_F$ yields

$$p(\mathbf{y}_k) = \frac{|g_k'(y_k)|}{\int |g_k'(y_k)| dy_k}, \quad k = 1, \ldots, K_1.$$  

(2.60)

We assume that (2.60) holds, at least approximately. Hence we can let the peak of $g_k'(y_k)$ be at $y_k = (\mathbf{y}_k)_\nu = 0$ and $\langle y_k^2 \rangle_{y_k} = \sigma_{y_k}^2 = \|\mathbf{w}_k\|^2$. Then combining (2.54), (2.58) and (2.60) we can get the optimal parameters $\tilde{\theta}_k$ for the nonlinear functions $f(y_k; \tilde{\theta}_k)$, $k = 1, \ldots, K_1$.

### 2.3 The Final Objective Function

Section 2.2 gives the initial optimal solutions by maximizing $I(\mathbf{X}; \mathbf{Y})$ and $I(\mathbf{Y}; R)$. In this section, we will discuss how to find the final optimal $V_0$ and other parameters by maximizing $I(\mathbf{X}; R)$ from the initial optimal solutions.
First, we have
\[ y = \hat{W}^T \hat{x} = a \hat{y}, \]  
(2.61)
where \( a \) is given in (2.43) and
\[ \hat{y} = (\hat{y}_1, \cdots, \hat{y}_{K_1})^T = C^T \hat{x} = \tilde{C}^T \tilde{x}, \]  
(2.62)
\[ \hat{x} = \Sigma_0^{-1/2} U_0^T x, \]  
(2.63)
\[ C = V_0^T \in \mathbb{R}^{K_0 \times K_1}, \]  
(2.64)
\[ \tilde{x} = U_0 \Sigma_0^{-1/2} U_0^T x = U_0 \hat{x}, \]  
(2.65)
\[ \tilde{C} = U_0 C = [\tilde{c}_1, \cdots, \tilde{c}_{K_1}]. \]  
(2.66)

Then, from the above, we get
\[ I(X; R) = I(\hat{X}; R) \simeq \hat{I}_G = \frac{1}{2} \left\langle \ln \left( \frac{\det(G(\hat{x}))}{2\pi e} \right) \right\rangle_{\hat{x}} + H(\hat{X}), \]  
(2.67)
\[ G(\hat{x}) = N \hat{W} \hat{\Phi} \hat{W}^T + I_{K_0}, \]  
(2.68)
\[ \hat{W} = \Sigma^{1/2} U^T \hat{W} A^{1/2} = \sqrt{K_1^{-1} I_{K_0}} C = \sqrt{K_0^{-1} I_{K_0}} C, \]  
(2.69)
where \( I_{K_0} \) is a \( K \times K \) diagonal matrix with value 1 on the diagonal and
\[ \hat{\Phi} = \Phi^2, \]  
(2.70)
\[ \Phi = \text{diag} (\phi(\hat{y}_1), \cdots, \phi(\hat{y}_{K_1})), \]  
(2.71)
\[ \phi(\hat{y}_k) = a^{-1} \left| \frac{\partial g(\hat{y}_k)}{\partial \hat{y}_k} \right|, \]  
(2.72)
\[ \hat{y}_k = a^{-1} y_k = c_k^T \hat{x}, \quad k = 1, \cdots, K_1. \]  
(2.73)

Then we have
\[ \det (G(\hat{x})) = \det \left( N K_0^{-1} C \hat{\Phi} C^T + I_{K_0} \right), \]  
(2.74)
for large \( N \) and \( K_0/N \rightarrow 0, \)
\[ \det (G(\hat{x})) \approx \det (J(\hat{x})) = \det \left( N K_0^{-1} C \hat{\Phi} C^T \right), \]  
(2.75)
and
\[ \hat{I}_G \approx \hat{I}_F = -Q - \frac{K_0}{2} \ln (2\pi e) - \frac{K_0}{2} \ln (\varepsilon) + H(\hat{X}), \]  
(2.76)
\[ Q = \frac{1}{2} \left\langle \ln \left( \det \left( C \hat{\Phi} C^T \right) \right) \right\rangle_{\hat{x}}, \]  
(2.77)
\[ \varepsilon = \frac{K_0}{N}. \]  
(2.78)

Hence we can state the optimization problem as:
\[ \text{minimize} \quad Q[C] = - \frac{1}{2} \left\langle \ln \left( \det \left( C \hat{\Phi} C^T \right) \right) \right\rangle_{\hat{x}}, \]  
(2.79)
subject to \( CC^T = I_{K_0}. \)  
(2.80)

The gradient from (2.79) reads as follows:
\[ \frac{dQ}{dC} = - \left\langle \left( C \hat{\Phi} C^T \right)^{-1} C \hat{\Phi} + \hat{x} \omega^T \right\rangle_{\hat{x}}, \]  
(2.81)
where \( C = [c_1, \cdots, c_{K_1}], \) \( \omega = (\omega_1, \cdots, \omega_{K_1})^T, \)
\[ \omega_k = \phi(\hat{y}_k) \phi'(\hat{y}_k) c_k^T \left( C \hat{\Phi} C^T \right)^{-1} c_k, \quad k = 1, \cdots, K_1. \]  
(2.82)

In the following we will discuss how to get the optimal solution of \( C \) for two specific cases.
2.3.1 Algorithm 1: $K_0 = K_1$

Now $CC^T = C^T C = I_{K_1}$, then by Eq. (2.79) we have

$$Q_1[C] = - \frac{1}{2} \sum_{k=1}^{K_1} \ln \left( \phi(\hat{y}_k) \right) \hat{x},$$

(2.83)

d$Q_1[C]$$
\frac{dQ_1[C]}{dC} = - \langle \hat{x} \omega^T \rangle_x,$$

(2.84)

$$\omega_k = \frac{g''(\hat{y}_k)}{g'(\hat{y})}, \ k = 1, \cdots, K_1.$$

(2.85)

Under the orthogonality constraints (Eq. 2.80), we can use the following update rule for learning $C$ (Edelman et al., 1998; Amari, 1999),

$$C^{t+1} = C^t + \mu_t \frac{dC^t}{dt},$$

(2.86)

d$C^t$$
\frac{dC^t}{dt} = - \frac{dQ_1[C^t]}{dC^t} + C^t \left( \frac{dQ_1[C^t]}{dC^t} \right)^T C^t,$$

(2.87)

where the learning rate parameter $\mu_t$ changes with the iteration count $t$, $t = 1, \cdots, t_{\text{max}}$. Here we can use the empirical average to approximate the integral in (2.84) (ref. Eq. 2.15). We can also apply stochastic gradient descent (SGD) method for online updating $C^{t+1}$ in (2.86).

The orthogonality constraint (Eq. 2.80) can accelerate the convergence rate. In practice, the orthogonal constraint (2.80) for objective function (2.79) is not necessary in this case. We can completely abandon the constraint condition,

$$\text{minimize } Q_2[C] = - \frac{1}{2} \sum_{k=1}^{K_1} \ln \left( \phi(\hat{y}_k) \right) \hat{x} - \frac{1}{2} \ln \left( \det \left( C^T C \right) \right),$$

(2.88)

where we assume rank $(C) = K_1 = K_0$. If we let

$$\frac{dC}{dt} = - C C^T \frac{dQ_2[C]}{dC},$$

(2.89)

then

$$\text{Tr} \left( \frac{dQ_2[C]}{dC} \frac{dC^T}{dt} \right) = - \text{Tr} \left( C^T \frac{dQ_2[C]}{dC} \frac{dQ_2[C]}{dC^T} \right) \leq 0.$$ (2.90)

Therefore we can use the similar update rule (see Eq. 2.86) for learning $C$. In fact, it can also be extended to the case $K_0 > K_1$ by using the same objective function (2.88).

For the learning rate parameter $\mu_t$ (see 2.86), we apply an adaptive method for updating it. First, calculate

$$\mu_t = \frac{\nu_t}{\kappa_t}, \ t = 1, \cdots, t_{\text{max}},$$

(2.91)

$$\kappa_t = \frac{1}{K_1} \sum_{k=1}^{K_1} \| \nabla C^t(:,k) \|,$$

(2.92)

and $C^{t+1}$ by (2.86) and (2.87), then calculate the value $Q_1[C^{t+1}$]. If $Q_1[C^{t+1}] < Q_1[C^t]$, then let $v_{t+1} \leftarrow v_t$, continue for next iteration; otherwise, let $v_t \leftarrow v_t / \tau$, $\mu_t \leftarrow v_t / \kappa_t$ and recalculate $C^{t+1}$ and $Q_1[C^{t+1}]$. Here $0 < v_1 < 1$ and $0 < \tau < 1$ are set as constants. After getting $C^{t+1}$ for each update, we employ a Gram–Schmidt orthonormalization process for matrix $C^{t+1}$, where the orthonormalization process can accelerate the convergence. However, we can discard the Gram–Schmidt orthonormalization process after iterative epochs $t_0 (> 1)$ for more accurate optimization solution $C$. In this case, the objective function is given by the Eq. (2.88). We can also further optimize the parameter $b$ by gradient descent.

When $K_0 = K_1$, the objective function (2.88) $Q_2[C]$ without constraint is the same as the objective function of infomax ICA (IICA) (Bell & Sejnowski, 1995; 1997), and we can get the same optimal
solution $C$. Hence, in this sense, the IICA can be regarded as a special case of our method. However, our method has a wider range of application and can handle a number of generic situations. Our model is derived by neural populations with a huge number of neurons and not only for additive noise neuron model. Moreover, our method can get a faster convergence rate for training than IICA (see Section [3]).

2.3.2 Algorithm 2: $K_0 \leq K_1$

In this case, it is computationally expensive to update $C$ by using the gradient of $Q$ (see Eq. 2.81), since it needs to compute the inverse matrix for every $\hat{x}$. Consider the following inequalities (see Appendix for proof).

Proposition 2.2. The following inequations hold,

\[
\frac{1}{2} \langle \ln \left( \det \left( C \hat{\Phi} C^T \right) \right) \rangle_{\hat{x}} \leq \frac{1}{2} \ln \left( \det \left( C \left( C \hat{\Phi} C^T \right) \right) \right),
\]

\[
\langle \ln \left( \det \left( C \Phi C^T \right) \right) \rangle_{\hat{x}} \leq \ln \left( \det \left( C \left( C \hat{\Phi} C^T \right) \right) \right),
\]

\[
\langle \ln \left( \det \left( C \Phi C^T \right) \right) \rangle_{\hat{x}} \leq \frac{1}{2} \ln \left( \det \left( C \left( C \hat{\Phi} C^T \right) \right) \right),
\]

\[
\langle \ln \left( \det \left( C \Phi C^T \right) \right) \rangle_{\hat{x}} \leq \frac{1}{2} \ln \left( \det \left( C \left( C \hat{\Phi} C^T \right) \right) \right),
\]

where $C \in \mathbb{R}^{K_0 \times K_1}$, $K_0 \leq K_1$, $CC^T = I_{K_0}$.

We use the following objective function $\hat{Q}[C]$ as a substitute for $Q[C]$ (see Appendix for detailed discussion) and write the optimization problem as:

\[
\text{minimize } \hat{Q}[C] = -\frac{1}{2} \ln \left( \det \left( C \left( C \hat{\Phi} C^T \right) \right) \right),
\]

subject to $CC^T = I_{K_0}$.

The update rule (2.80) may also apply here and a modified algorithm similar to algorithm 1 may be used for parameter learning.

3 Experimental Results

We performed our all experiments using Matlab 2016a on a machine with 12 Intel CPU cores (at 2.4 GHz) and have applied our methods to natural images from Olshausen’s image dataset (Olshausen & Field, 1996) and images of handwritten digits from MNIST dataset (LeCun et al., 1998). The gray level of each raw image was normalized to the range of 0 to 1. The $M$ image patches with size $w \times w = K$ for training are randomly sampled from the natural images. We use the Poisson neuron model with the sigmoid like tuning function $f(y; \theta) = \frac{1}{1 + \exp(-\beta y - b)}$, then $g(y) = 2\sqrt{f(y; \bar{\theta})} = \frac{1}{1 + \exp(-\beta y - b)}$, where $\bar{\theta} = (\beta, b)^T$. We can solve their initial value (see section 2.2.2), $b_0 = 0$, $\beta_0 = 1.81 \sqrt{K_1 K_0^{-1}}$. For our experiments, we set $\beta = 0.5\beta_0$ for $t = 1, \cdots, t_0$ and $\beta = \beta_0$ for $t = t_0 + 1, \cdots, t_{\max}$.

Firstly, we tested the case of $K = K_0 = K_1 = 144$ and randomly sampled $M = 10^5$ image patches with size $12 \times 12$ from the Olshausen’s natural images, assuming that $N = 10^6$ neurons were divided into $K_1 = 144$ classes and $\epsilon = 1$ (see Eq. 2.49). The input patches were preprocessed by a ZCA whitening filter (see Eq. 2.65). To test our algorithms, we chose the batch size to be equal to the number of training samples $M$, although we could also choose a smaller batch size. We updated the matrix $C$ from a random start, and set parameters $t_{\max} = 300$, $t_0 = 50$, $v_1 = 0.4$, and $\tau = 0.8$ for all experiments.

In this case, the optimal solution $C$ looked similar to the optimal solution of IICA (Bell & Sejnowski, 1997). We also compared the fast ICA (FICA) (Hyvärinen, 1999), a faster algorithm than IICA.
We also tested the restricted Boltzmann machine (RBM) (Hinton, 2002; Hinton et al., 2006) for a unsupervised learning of representations, and found that it could not easily learn Gabor-like filters as trained by contrastive divergence with Olshausen’s image dataset. However, an improved method by adding a sparsity constraint on the output units, e.g., sparse RBM (SRBM) (Lee et al., 2008) or sparse autoencoder (Hinton, 2010; Ng, 2011), can attain Gabor-like filters from this dataset. Similar results (i.e. Gabor-like filters) were also reproduced by the denoising autoencoders (Vincent et al., 2010), which method requires a careful choice of parameters, such as noise level, learning rate, and batch size.

In order to compare our methods, i.e. algorithm 1 (Alg.1) and algorithm 2 (Alg.2), with other methods, i.e. IICA, FICA and SRBM, we implemented these algorithms by using the same initial weights and the same training data set (i.e. 10^5 image patches preprocessed by a ZCA whitening filter). To get a good result by IICA, we must carefully select the parameters; we set the batch size as 50, the initial learning rate as 0.01, and final learning rate as 0.0001, with an exponential decay with the epoch of iterations. IICA tends to have a faster convergence rate for a bigger batch size but it may become harder to escape local minima. For FICA, we chose the nonlinearity function \( f(u) = \log \cosh(u) \) as contrast function, and for SRBM, we set the sparseness control constant \( p \) as 0.01 and 0.03. The number of epochs for iterations was set to 300 for all algorithms. Figure 2 shows the filters learnt by our methods and other methods. Each filter in Figure 2(a) corresponds to a column vector of matrix \( \tilde{\Phi} \) (see Eq. 2.66), where each vector for display is normalized by \( \tilde{c}_k \leftarrow \tilde{c}_k / \max(|\tilde{c}_{1,k}|, \ldots, |\tilde{c}_{K_k,k}|) \), \( k = 1, \ldots, K_1 \). The results in Figures 2(a), 2(b) and 2(c) look very similar to one another, and slightly different from the results in Figure 2(d) and 2(e). There are no Gabor-like filters in Figure 2(f), which corresponds to SRBM with \( p = 0.03 \).

To quantify the efficiency of learning representations by the above algorithms, we calculate the coefficient entropy (CFE) for estimating coding cost (Lewicki & Olshausen, 1999; Lewicki & Sejnowski, 2000) as follows:

\[
\hat{y}_k = \zeta \tilde{\Phi}^T \tilde{x}, \quad k = 1, \ldots, K_1, \tag{3.100}
\]

\[
\zeta = \frac{K_1}{\sum_{k=1}^{K_1} \| \tilde{\Phi}_k \|}, \tag{3.101}
\]

where \( \tilde{x} \) is defined by Eq. (2.65), and \( \tilde{\Phi}_k \) is the corresponding optimal filter. To estimate the probability density of coefficients \( q_k(\hat{y}_k) \) \((k = 1, \ldots, K_1)\) from the \( M \) training samples, we apply the kernel density estimation for \( q_k(\hat{y}_k) \) and use a normal kernel with an adaptive optimal window width. Then we define the CFE \( h \) as

\[
h = \frac{1}{K_1} \sum_{k=1}^{K_1} H_k(\hat{Y}_k), \tag{3.102}
\]

\[
H_k(\hat{Y}_k) = -\Delta \sum_n q_k(n\Delta) \log_2 q_k(n\Delta), \tag{3.103}
\]

where \( q_k(\hat{y}_k) \) is quantized as discrete \( q_k(n\Delta) \) and \( \Delta \) is the step size.

Methods such as IICA and SRBM as well as our methods are feedforward structures in which information is transferred directly through a nonlinear function, e.g., the sigmoid function. Hence, we can use the amount of transmitted information to measure the results learnt by these methods. We consider a stochastic system with nonlinear transfer functions, which is a neural population with \( N \) neurons. We chose the sigmoid function as the transfer function and Gaussian noise with standard deviation set to 1 as the system noise. In this case, from (2.2), (A.2) and (A.5), we know that the proximate \( \text{MI} \ I_G \) is equivalent to Poisson neuron model. It follows from (2.67)–(2.78) that

\[
I(\hat{X}; R) = I(\tilde{X}; \tilde{R}) = H(\tilde{X}) - H(\tilde{X}|\tilde{R}) \simeq \tilde{I}_G = H(\tilde{X}) - h_1, \tag{3.104}
\]

\[
H(\tilde{X}|\tilde{R}) \simeq h_1 = -\frac{1}{2} \left\langle \ln \left( \det \left( \frac{1}{2\pi e} \left( NK_0^{-1} \Phi \Phi^T + I_{K_0} \right) \right) \right) \right\rangle_\tilde{x}, \tag{3.105}
\]

where we set \( N = 10^6 \). A good representation should make the MI \( I(\hat{X}; R) \) as big as possible. Equivalently, for the same inputs, a good representation should make the conditional entropy (CDE) \( H(\tilde{X}|\tilde{R}) \) (or \( h_1 \)) as small as possible.

Figure 3 shows how CFE and CDE varied with training time. We calculated CFE and CDE by sampling once every 10 epochs from a total of 300 epochs. These results show that our algorithms
Figure 2: Comparison of filters obtained from $10^5$ natural image patches with size $12 \times 12$ by our methods (Alg.1 and Alg.2) and other methods. The number of output filters is $K_1 = 144$. (a) and (b): Alg.1 and Alg.2, respectively. (c): IICA. (d): FICA. (e): SRBM ($p = 0.01$). (f): SRBM ($p = 0.03$).

Figure 3: Comparison of quantization effects and convergence rate by coefficient entropy (ref. 3.102) and conditional entropy (ref. 3.105) corresponding to training results (filters) shown in Figure 2. The coefficient entropy (panel a) or conditional entropy (panel b and c) are shown a function of training time on a logarithmic scale. All experiments run on the same machine using Matlab. Here we sampled once every 10 epochs out of a total of 300 epochs. Notice that we set epoch number $t_0 = 50$ for Alg.1 and Alg.2 and the start time was set to 1 second. See main text for more details.

had a fast convergence rate towards stable solutions while having CFE and CDE values similar to the algorithm of IICA, which converged much more slowly. Here the values of CFE and CDE should be as small as possible for a good representation learnt from the same data set. Notice that here we set epoch number $t_0 = 50$ in our algorithms (see Alg.1 and Alg.2), and the start time was set to 1 second. This explains the step seen in Figure 3(b) for Alg.1 and Alg.2 since the parameter $\beta$ was updated when epoch number $t = t_0$. FICA had a convergence rate close to our algorithms but had a big CFE, which is reflected by the quality of the filter results in Figure 2. The convergence rate and CFE for SRBM were close to IICA, but SRBM had a much bigger CDE than IICA, which implies
that the information had a greater loss when passing through the system optimized by SRBM than by IICA or our methods.

From Figure 3(c) we see that the CDE (or MI I (X; R), see Eq. 3.104 and 3.105) decreases (or increases) with the increase of the value of the sparseness control constant p. Note that a smaller p means sparser outputs. Hence, in this sense, increasing sparsity may result in sacrificing some information. On the other hand, a weak sparsity constraint may lead to failure of learning a significant result (i.e., Gabor-like filters, see Figure 2(f)), and increasing sparsity has an advantage in eliminating the impact of noise in many practical cases.

Similar situation also occurs in sparse coding (Olshausen & Field, 1997), which provides a class of algorithms for learning overcomplete dictionary representations of the input signals. However, its training is time consuming due to its expensive computational cost, although many new training algorithms have emerged (Aharon et al., 2006; Elad & Aharon, 2006; Lee et al., 2006; Mairal et al., 2009; 2010). We compared our algorithm with an up-to-date sparse coding algorithm, mini-batch dictionary learning (MBDL) as given in (Mairal et al., 2009; 2010) and integrated in Python library, i.e., scikit-learn. The input data was the same as the above, i.e., 10^5 nature image patches preprocessed by a ZCA whitening filter.

We denotes the optimal dictionary learnt by MBDL as \( \hat{\mathbf{B}} \in \mathbb{R}^{K \times K_1} \) for which each column represents a basis vector, then we have that

\[
\mathbf{x} \approx \mathbf{U} \Sigma^{1/2} \mathbf{U}^T \hat{\mathbf{B}} \mathbf{y} = \hat{\mathbf{B}} \mathbf{y},
\]

\[
\hat{\mathbf{B}} = \mathbf{U} \Sigma^{1/2} \mathbf{U}^T \hat{\mathbf{B}},
\]

where \( \mathbf{y} = (y_1, \ldots, y_{K_1})^T \) is the coefficient vector.

Similarly, we can obtain a dictionary from the filter matrix C. Suppose \( \text{rank} (\mathbf{C}) = K_0 \leq K_1 \), then it follows from (2.61) that

\[
\hat{\mathbf{x}} = (a \mathbf{C} \mathbf{C}^T)^{-1} \mathbf{C} \mathbf{y}.
\]

And by (2.63) and (3.108), we get

\[
\mathbf{x} \approx \mathbf{B} \mathbf{y} = a \mathbf{B} \mathbf{C}^T \Sigma_0^{-1/2} \mathbf{U}_0^T \mathbf{x},
\]

\[
\mathbf{B} = a^{-1} \mathbf{U}_0 \Sigma_0^{1/2} \left( \mathbf{C} \mathbf{C}^T \right)^{-1} \mathbf{C} = [\mathbf{b}_1, \ldots, \mathbf{b}_{K_1}],
\]

where \( \mathbf{y} = \mathbf{W}^T \mathbf{x} = a \mathbf{C}^T \Sigma_0^{-1/2} \mathbf{U}_0^T \mathbf{x} \), the vectors \( \mathbf{b}_1, \ldots, \mathbf{b}_{K_1} \) can be regarded as the basis vectors and the strict equality holds when \( K_0 = K_1 = K \). Recall that \( \mathbf{X} = [x_1, \ldots, x_M] = \mathbf{USV}^T \) (see Eq. 2.6) and \( \mathbf{Y} = [y_1, \ldots, y_M] = \mathbf{W}^T \mathbf{X} = a \sqrt{M - \mathbf{TC}^T \mathbf{V}_{0}^T} \), then we can get \( \hat{\mathbf{X}} = \mathbf{B} \mathbf{Y} = \sqrt{M - \mathbf{U}_0 \Sigma_0^{1/2} \mathbf{V}_{0}^T} \approx \mathbf{X} \). Hence, the Eq. (3.109) holds.

The basis vectors shown in Figure 4(a)-4(e) correspond to filters in Figure 2(a)-2(e). And Figure 4(f) illustrates the optimal dictionary \( \hat{\mathbf{B}} \) learnt by MBDL, where we set the regularization parameter as \( \lambda = 1/2 \sqrt{K} \), the batch size as 50 and the total number of iterations to perform as 20000, which took about 3 hours for training. From Figure 4 we see that these basis vectors obtained by the above algorithms have local Gabor-like shapes except for those by SRBM. If \( \text{rank}(\hat{\mathbf{B}}) = K = K_1 \), then the matrix \( \hat{\mathbf{B}}^{-T} \) can be regarded as a filter matrix like matrix \( \hat{\mathbf{C}} \) (see Eq. 2.66). However, from the column vector of matrix \( \hat{\mathbf{B}}^{-T} \) we cannot find any local Gabor-like filter that resembles the filters shown in Figure 2. Our algorithm has less computational cost and a much faster convergence rate than the sparse coding algorithm. Moreover, the sparse coding method involves a dynamic generative model that requires relaxation and is therefore unsuitable for fast inference, whereas the feedforward framework of our model is easy for inference because it only requires evaluating the nonlinear tuning functions.

We have trained our model on the same nature image patches with a highly overcomplete setup by optimizing the objective (2.98) by Alg.2 and got Gabor-like filters. The results of 400 typical filters chosen from 1024 output filters are displayed in Figure 5(a) and corresponding base (see Eq. 3.110) are shown in Figure 5(b). In this case, we set parameters \( K_1 = 1024, t_{\max} = 100, t_0 = 50, \nu_1 = 0.4, \tau = 0.8, \epsilon = 0.98 \) (see 2.49), and got rank(\( \mathbf{B} \)) = \( K_0 = 82 \). Compared to the ICA-like
Figure 4: Comparison of basis vectors obtained by our method and other methods. Panel (a)–(e) correspond to panel (a)–(e) in Figure 2, where the basis vectors are given by (3.110). The basis vectors in panel (f) are learnt by MBDL and given by (3.107).

results in Figure 2(a)–Figure 2(c), the average size of Gabor-like filters in Figure 5(a) is bigger, indicating that the small noise-like local structures in images have been filtered out.

We have also trained our model on 60000 images of handwritten digits from MNIST dataset (LeCun et al., 1998) and the resultant 400 typical optimal filters and bases are shown in Figure 5(c) and Figure 5(d), respectively. All parameters were the same as Figure 5(a) and Figure 5(b), $K_1 = 1024$, $t_{max} = 100$, $t_0 = 50$, $v_1 = 0.4$, $\tau = 0.8$ and $\epsilon = 0.98$, from which we got rank $(B) = K_0 = 183$. From these figures we can see that the salient features of the input images are reflected in these filters and bases. We could also get the similar overcomplete filters and bases by SRBM and MBDL. However, the results depended sensitively on the choice of parameters and the training took a long time.

Figure 6 shows that CFE as a function of training time for Alg.2, where Figure 6(a) corresponds to Figure 5(a)–5(b) for learning nature image patches and Figure 6(b) corresponds to Figure 5(c)–5(d) for learning MNIST dataset. We set parameters $t_{max} = 100$, $t_0 = 50$ and $\tau = 0.8$ for all experiments and varied parameter $v_1$ for each experiment, with $v_1 = 0.2$, 0.4, 0.6 or 0.8. These results indicate a fast convergence rate for training on different datasets. Generally, the convergence is not sensitive to the change of parameter $v_1$.

We have also performed additional tests on other image datasets and got similar results, confirming the speed and robustness of our learning method. Compared with other methods, e.g., IICA, FICA, MBDL, SRBM or sparse autoencoders etc., our method appeared to be more efficient and robust for unsupervised learning of representations. We also found that the complete or overcomplete filters and bases learnt by our methods have the local Gabor-like shapes while the results by SRBM or MBDL do not have this property.

Similar to the sparse coding method applied to image denoising (Elad & Aharon, 2006), our method (see Eq. 3.110) can also be applied to image denoising. Figure 7 shows an example of image denoising. The filters or bases were learnt by using image patches with size $7 \times 7$ sampled from the left half of the image, and subsequently used to reconstruct the right half of the image which was
Figure 5: Filters and bases obtained from Olshausen’s image dataset and MNIST dataset by Algorithm 2. (a) and (b): 400 typical filters and the corresponding bases obtained from Olshausen’s image dataset, where $K_0 = 82$ and $K_1 = 1024$. (c) and (d): 400 typical filters and the corresponding bases obtained from the MNIST dataset, where $K_0 = 183$ and $K_1 = 1024$.

distorted by Gaussian noise. A common practice for evaluating the results of image denoising is by looking at the difference between the reconstruction and the original image. If the reconstruction is perfect the difference will look like Gaussian noise. In Figure 7(c) and 7(d) a dictionary (100 bases) was learnt by MBDL and orthogonal matching pursuit was used to estimate the sparse solution $\hat{x}$.

For our method (showed in Figure 7(b)), we first get the optimal filters parameter $W$, where $W$ is a low rank matrix ($K_0 < K$), then from the distorted image patches $x_m$ ($m = 1, \cdots, M$), we can get filter outputs $y_m = W^T x_m$ and reconstruction $\hat{x}_m = By_m$, where we set parameter $\epsilon = 0.975$ and $K_0 = K_1 = 14$. As can be seen from Figure 7, our method worked better than dictionary learning, although we only used 14 bases comparing with 100 bases used by dictionary learning. Moreover, our method is more efficient. In fact, we can get better optimal bases $B$ by a generative model using our infomax approach, which we will discuss in another paper.

Python source code is available at http://scikit-learn.org/stable/downloads/plot_image_denoising.py
Figure 6: CFE as a function of training time for Alg.2, with $v_1 = 0.2, 0.4, 0.6$ or $0.8$. In all experiments parameters were set to $t_{\text{max}} = 100$, $t_0 = 50$ and $\tau = 0.8$. (a): corresponding to Figure 5(a) or Figure 5(b) (b): corresponding to Figure 5(c) or Figure 5(d).

Distorted image

Algorithm 1

Figure 7: Image denoising. (a): the right half of the original image is distorted by Gaussian noise and the norm of the difference between the distorted image and the original image is 23.48. (b): image denoising by our method (Algorithm 1), with 14 bases used. (c) and (d): image denoising using dictionary learning, with 100 bases used.

4 CONCLUSIONS

In this paper, we have presented a framework for unsupervised learning of representations via information maximization for neural populations. Information theory is a powerful tool for machine learning and it also provides a benchmark of optimization principle for neural information processing in nervous systems. In our framework, first, we give an asymptotic approximation to MI for a large-scale neural population. Then, to solve the infomax objective, we demonstrate that hierarchical infomax can provide a good approximation to the global optimal solution. Analytical solutions of the hierarchical infomax are further improved by a fast convergence algorithm based on gradient de-
scent. Using this method, we can optimize a highly nonlinear function via hierarchical optimization using infomax principle.

Moreover, from the viewpoint of information theory, the unsupervised pre-training for deep learning (Hinton & Salakhutdinov, 2006; Bengio et al., 2007) may be reinterpreted as a process of hierarchical infomax given by our framework, which might help explain why unsupervised pre-training helps deep learning (Erhan et al., 2010). In our framework, a pre-whitening step can emerge naturally by the hierarchical infomax, which might also explain why a pre-whitening step is useful for training in many learning algorithms (Coates et al., 2011; Bengio, 2012).

Our model naturally incorporates a considerable degree of biological realism, allowing a large-scale neural population with Poisson neuron model, and taking into account of several actual biological constraints, such as membrane noise, limited energy, and bounded connection weights. We employ a technique to attain a low-rank weight matrix for optimization, so as to avoid the influence of noise and reduce overfitting during training. In our model, many parameters have been optimized, including the population density of parameters, filter weight vectors, and parameters for nonlinear tuning functions, which could not be done by many other methods.

Our experimental results suggest that our method for unsupervised learning of representations has obvious advantages in its training speed and robustness over the main existing methods. Our model has a nonlinear feedforward structure and is convenient for fast learning and inference, which provides a simple and flexible framework for unsupervised learning of presentations and should be easily extended to training deep structure networks. In future work, it would interesting to use our method to train deep structure networks with unsupervised or supervised learning.

ACKNOWLEDGMENTS

We thank Prof. Honglak Lee for sharing Matlab code for algorithm comparison, Prof. Shan Tan for discussions and comments and Kai Liu for helping draw Figure 1. Supported by grant NIH-NIDCD R01 DC013698.

REFERENCES

Aharon, M., Elad, M., & Bruckstein, A. (2006). K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation. *Signal Processing, IEEE Transactions on*, 54(11), 4311–4322.

Amari, S. (1999). Natural gradient learning for over- and under-complete bases in ica. *Neural Comput.*, 11(8), 1875–1883.

Atick, J. J. (1992). Could information theory provide an ecological theory of sensory processing? *Network: Comp. Neural.*, 3(2), 213–251.

Barlow, H. B. (1961). Possible principles underlying the transformation of sensory messages. *Sensory Communication*, (pp. 217–234).

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

Bell, A. J. & Sejnowski, T. J. (1997). The “independent components” of natural scenes are edge filters. *Vision Res.*, 37(23), 3327–3338.

Bengio, Y. (2012). Deep learning of representations for unsupervised and transfer learning. *Unsupervised and Transfer Learning Challenges in Machine Learning*, 7, 19.

Bengio, Y., Courville, A., & Vincent, P. (2013). Representation learning: A review and new perspectives. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 35(8), 1798–1828.

Bengio, Y., Lamblin, P., Popovici, D., Larochelle, H., et al. (2007). Greedy layer-wise training of deep networks. *Advances in neural information processing systems*, 19, 153.
Borst, A. & Theunissen, F. E. (1999). Information theory and neural coding. *Nature neuroscience*, 2(11), 947–957.

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

Carlo, C. N. & Stevens, C. F. (2013). Structural uniformity of neocortex, revisited. *Proceedings of the National Academy of Sciences*, 110(4), 1488–1493.

Clarke, B. S. & Barron, A. R. (1990). Information-theoretic asymptotics of Bayes methods. *IEEE Trans. Inform. Theory*, 36(3), 453–471.

Coates, A., Ng, A. Y., & Lee, H. (2011). An analysis of single-layer networks in unsupervised feature learning. In *International conference on artificial intelligence and statistics* (pp. 215–223).

Cortes, C. & Vapnik, V. (1995). Support-vector networks. *Machine learning*, 20(3), 273–297.

Cover, T. M. & Thomas, J. A. (2006). *Elements of Information, 2nd Edition*. New York: Wiley-Interscience.

Edelman, A., Arias, T. A., & Smith, S. T. (1998). The geometry of algorithms with orthogonality constraints. *SIAM J. Matrix Anal. Appl.*, 20(2), 303–353.

Elad, M. & Aharon, M. (2006). Image denoising via sparse and redundant representations over learned dictionaries. *Image Processing, IEEE Transactions on*, 15(12), 3736–3745.

Erhan, D., Bengio, Y., Courville, A., Manzagol, P.-A., Vincent, P., & Bengio, S. (2010). Why does unsupervised pre-training help deep learning? *The Journal of Machine Learning Research*, 11, 625–660.

Glorot, X., Bordes, A., & Bengio, Y. (2011). Deep sparse rectifier neural networks. In *International Conference on Artificial Intelligence and Statistics* (pp. 315–323).

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., & Bengio, Y. (2014). Generative adversarial nets. In *Advances in Neural Information Processing Systems* (pp. 2672–2680).

Hinton, G. (2010). A practical guide to training restricted boltzmann machines. *Momentum*, 9(1), 926.

Hinton, G., Osindero, S., & Teh, Y.-W. (2006). A fast learning algorithm for deep belief nets. *Neural computation*, 18(7), 1527–1554.

Hinton, G. E. (2002). Training products of experts by minimizing contrastive divergence. *Neural computation*, 14(8), 1771–1800.

Hinton, G. E. & Salakhutdinov, R. R. (2006). Reducing the dimensionality of data with neural networks. *Science*, 313(5786), 504–507.

Hoerl, A. E. & Kennard, R. W. (1970). Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(1), 55–67.

Huang, W. & Zhang, K. (2016). Information-theoretic bounds and approximations in neural population coding. *Neural Comput, submitted*, URL https://arxiv.org/abs/1611.01414.

Hubel, D. H. & Wiesel, T. N. (1962). Receptive fields, binocular interaction and functional architecture in the cat’s visual cortex. *The Journal of physiology*, 160(1), 106–154.

Hyvärinen, A. (1999). Fast and robust fixed-point algorithms for independent component analysis. *Neural Networks, IEEE Transactions on*, 10(3), 626–634.

Hyvärinen, A. (2005). Estimation of non-normalized statistical models using score matching. *The Journal of Machine Learning Research*, 6, 695–709.
Konstantinides, K. & Yao, K. (1988). Statistical analysis of effective singular values in matrix rank determination. *Acoustics, Speech and Signal Processing, IEEE Transactions on*, 36(5), 757–763.

Kreutz-Delgado, K., Murray, J. F., Rao, B. D., Engan, K., Lee, T. S., & Sejnowski, T. J. (2003). Dictionary learning algorithms for sparse representation. *Neural computation*, 15(2), 349–396.

Le, Q. V., Karpenko, A., Ngiam, J., & Ng, A. Y. (2011). Ica with reconstruction cost for efficient overcomplete feature learning. In *Advances in Neural Information Processing Systems* (pp. 1017–1025).

LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11), 2278–2324.

Lee, H., Battle, A., Raina, R., & Ng, A. Y. (2006). Efficient sparse coding algorithms. In *Advances in neural information processing systems* (pp. 801–808).

Lee, H., Ekanadham, C., & Ng, A. Y. (2008). Sparse deep belief net model for visual area v2. In *Advances in neural information processing systems* (pp. 873–880).

Lee, T.-W., Girolami, M., Bell, A. J., & Sejnowski, T. J. (2000). A unifying information-theoretic framework for independent component analysis. *Computers & Mathematics with Applications*, 39(11), 1–21.

Lewicki, M. S. & Olshausen, B. A. (1999). Probabilistic framework for the adaptation and comparison of image codes. *JOSA A*, 16(7), 1587–1601.

Lewicki, M. S. & Sejnowski, T. J. (2000). Learning overcomplete representations. *Neural computation*, 12(2), 337–365.

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

MacKay, D. J. C. (2003). *Information Theory, Inference and Learning Algorithms*. Cambridge: Cambridge University Press.

Mairal, J., Bach, F., Ponce, J., & Sapiro, G. (2009). Online dictionary learning for sparse coding. In *Proceedings of the 26th annual international conference on machine learning* (pp. 689–696). ACM.

Mairal, J., Bach, F., Ponce, J., & Sapiro, G. (2010). Online learning for matrix factorization and sparse coding. *The Journal of Machine Learning Research*, 11, 19–60.

Montufar, G. F., Pascanu, R., Cho, K., & Bengio, Y. (2014). On the number of linear regions of deep neural networks. In *Advances in Neural Information Processing Systems* (pp. 2924–2932).

Nair, V. & Hinton, G. E. (2010). Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)* (pp. 807–814).

Ng, A. (2011). Sparse autoencoder. *CS294A Lecture notes*, 72, 1–19.

Olshausen, B. A. & Field, D. J. (1996). Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, 381(6583), 607–609.

Olshausen, B. A. & Field, D. J. (1997). Sparse coding with an overcomplete basis set: A strategy employed by v1? *Vision Res.*, 37(23), 3311–3325.

Pouget, A., Dayan, P., & Zemel, R. (2000). Information processing with population codes. *Nature Reviews Neuroscience*, 1(2), 125–132.

Ranzato, M., Poultney, C., Chopra, S., & LeCun, Y. (2007). Efficient learning of sparse representations with an energy-based model. In *Advances in neural information processing systems* (pp. 1137–1144).

Rao, C. R. (1945). Information and accuracy attainable in the estimation of statistical parameters. *Bulletin of the Calcutta Mathematical Society*, 37(3), 81–91.
A.1 Formulas for Poisson Noise and Gaussian Noise

For Poisson neuron model, from Eq. (2.2) we get

\[
p(r_n|x; \theta_n) = \frac{f(x; \theta_n)^{r_n}}{r_n!} \exp(-f(x; \theta_n)), \quad (A.1)
\]

\[
S(x; \theta_n) = \frac{1}{f(x; \theta_n)} \frac{\partial f(x; \theta_n)}{\partial x} \frac{\partial f(x; \theta_n)}{\partial x^T} = \frac{\partial g(x; \theta_n)}{\partial x} \frac{\partial g(x; \theta_n)}{\partial x^T}, \quad (A.2)
\]

where \( f(x; \theta_n) \geq 0 \) is the tuning curve (mean response) of the \( n \)-th neuron and \( g(x; \theta_n) = 2\sqrt{f(x; \theta_n)} \). (A.3)

Similarly, for Gaussian noise, we have

\[
p(r_n|x; \theta_n) = \frac{1}{\sigma \sqrt{2\pi}} \exp \left( -\frac{(r_n - f(x; \theta_n))^2}{2\sigma^2} \right), \quad (A.4)
\]

\[
S(x; \theta_n) = \frac{1}{\sigma^2} \frac{\partial f(x; \theta_n)}{\partial x} \frac{\partial f(x; \theta_n)}{\partial x^T}, \quad (A.5)
\]

where \( \sigma > 0 \) denotes the standard deviation of noise.

A.2 On the Choice of Parameter \( K_0 \)

From (2.35) and (2.48) we get the optimal \( K_0 = \arg \max_{K_0} \left( E \tilde{K}_0^{-1} \sum_{k=1}^{\tilde{K}_0} \sigma_k^{-2} \right) \). Generally, by this constraint, small values of \( \sigma_k^2 \) will result in \( K_0 < K \) and a low-rank matrix \( W \) (Eq. 2.41).

On the other hand, the low-rank matrix \( W \) can filter out the noise of stimulus \( x \). Consider the transformation \( Y = W^T X \), where \( X = [x_1, \cdots, x_M] \) and \( Y = [y_1, \cdots, y_M] \) for \( M \) samples. It follows from SVD of \( X \) that

\[
X = US\tilde{V}^T, \quad (A.6)
\]

where \( U \) is given in (2.23), \( \tilde{V} \) is a \( M \times M \) unitary orthogonal matrix and \( S \) is a \( K \times M \) diagonal matrix with non-negative real numbers on the diagonal, \( S_{k,k} = \sqrt{M-1}\sigma_k \) (\( k = 1, \cdots, K \), \( K \leq M \)), \( SS^T = (M-1)\Sigma \). Let

\[
\bar{X} = \sqrt{M-1}U_0 \Sigma_0^{1/2} \tilde{V}^T_0 \approx X, \quad (A.7)
\]
where $\Sigma_0$ and $U_0$ are given in (2.44) and (2.45), respectively, $\tilde{V}_0 = \tilde{V}(:,1:K_0) \in \mathbb{R}^{M \times K_0}$. Then
\[
Y = W^T X = \alpha V_0 \Sigma_0^{-1/2} U_0^T \Sigma V^T = W^T \tilde{X} = \alpha \sqrt{M-1} V_0 \tilde{V}_0^T. \quad (A.8)
\]

Here $\tilde{X}$ can be regarded as a denoised version of $X$. The determination of the effective rank $K_0 \leq K$ of the matrix $\tilde{X}$ by using singular values is based on various criteria \cite{KonstantinidesYao1988}. Here we let $K_0$ satisfy
\[
K_0 = \arg \min_{K_0} \left( \frac{\sum_{k=1}^{K_0} \sigma_k^2}{\sum_{k=1}^{K} \sigma_k^2} \geq \epsilon \right), \quad (A.9)
\]
where $\epsilon$ is a positive constant, $0 < \epsilon \leq 1$.

### A.3 Proof of Proposition 2.1

**Proof.** It follows from the data-processing inequality \cite{CoverThomas2006} that
\[
I(X;R) \leq I(\tilde{Y};R), \quad (A.10)
\]
\[
I(X;R) \leq I(X;\tilde{Y}) \leq I(X;\tilde{Y}) \quad (A.11)
\]

On the other hand, when $N_k$ is large, from Eq. (2.19) we know that the distribution of $\tilde{Z}_k$, $N(0,N_k^{-1}\sigma^2)$, approaches a Dirac delta function $\delta(\tilde{z}_k)$, then by (2.16) and (2.18) we get $p(r|\tilde{y}) \simeq p(r|y) = p(r|x)$ and
\[
I(X;R) = I(Y;R) - \left( \ln \frac{p(r|y)}{p(r|x)} \right)_{r,x} = I(Y;R), \quad (A.12)
\]
\[
I(Y;R) = I(\tilde{Y};R) - \left( \ln \frac{p(r|y)}{p(r|\tilde{y})} \right)_{r,y,\tilde{y}} \simeq I(\tilde{Y};R). \quad (A.13)
\]

It follows from (A.10), (A.12) and (A.13) that (2.20) and (2.22) hold. Notice that
\[
p(\tilde{y}_k|x) = p(\tilde{y}_{k_1}, \ldots, \tilde{y}_{k_{N_k}}|x) = N(w_k^T x, N_k^{-1}\sigma^2), k = 1, \ldots, K_1. \quad (A.14)
\]

Then we have
\[
p(\tilde{y}|x) = p(\tilde{y}|x), \quad (A.15)
\]
\[
I(X;\tilde{Y}) = I(X;\tilde{Y}). \quad (A.16)
\]

Combining (A.11) and (A.16) we immediately get (2.21). This completes the proof of Proposition 2.1.

### A.4 Proof of Proposition 2.2

**Proof.** Functions $\ln \det \left( \left\langle C \left( \mathbf{\Phi} \right) \tilde{X} \right\rangle C^T \right)$ and $\ln \det \left( \left\langle C \left( \mathbf{\Phi} \right) \tilde{X} \right\rangle C^T \right)$ are concave functions about $p(\tilde{x})$ (see the proof of Proposition 5.2. in \cite{HuangZhang2016}), which fact establishes inequalities (2.93) and (2.94).

Next we will prove the inequality (2.97). By SVD, we have
\[
C\Phi = \hat{U}\hat{D}\tilde{V}^T. \quad (A.17)
\]

where $\hat{U}$ is a $K_0 \times K_0$ unitary orthogonal matrix, $\tilde{V} = [\tilde{v}_1, \tilde{v}_2, \ldots, \tilde{v}_{K_1}]$ is an $K_1 \times K_1$ unitary orthogonal matrix, and $\hat{D}$ is an $K_0 \times K_1$ rectangular diagonal matrix with $K_0$ positive real numbers.
on the diagonal. Then by the matrix Hadamard’s inequality and Cauchy–Schwarz inequality we have that
\[
\begin{align*}
\det \left( \mathbf{C} \Phi \mathbf{C}^T \mathbf{C} \Phi \mathbf{C}^T \right)^{-1} \\
&= \det \left( \mathbf{D} \mathbf{V}^T \mathbf{C}^T \mathbf{C} \mathbf{V} \mathbf{D} \mathbf{V}^T \mathbf{C} \mathbf{V} \mathbf{D} \mathbf{D}^T \right)^{-1} \\
&= \det \left( \mathbf{V}_1^T \mathbf{C} \mathbf{V}_1 \right) \\
&= \det \left( \mathbf{C} \mathbf{V}_1 \right)^2 \\
&\leq \prod_{k=1}^{K_0} \left( \mathbf{C} \mathbf{V}_1 \right)^2_{k,k} \\
&\leq \prod_{k=1}^{K_0} \left( \mathbf{C} \mathbf{C}^T \right)^2_{k,k} \left( \mathbf{V}_1^T \mathbf{V}_1 \right)^2_{k,k} \\
&= 1,
\end{align*}
\] (A.18)

where \( \mathbf{V}_1 = [\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_{K_0}] \in \mathbb{R}^{K_1 \times K_0} \). The last equality holds because of \( \mathbf{C} \mathbf{C}^T = \mathbf{I}_{K_0} \) and \( \mathbf{V}_1^T \mathbf{V}_1 = \mathbf{I}_{K_0} \). Hence inequality (2.97) establishes and the equality holds if and only if \( K_0 = K_1 \) or \( \mathbf{C} \mathbf{V}_1 = \mathbf{I}_{K_0} \).

Similarly, we get inequality (2.95),
\[
\ln \left( \det \left( \mathbf{C} \langle \Phi \rangle \mathbf{C}^T \right) \right) \leq \frac{1}{2} \ln \left( \det \left( \mathbf{C} \langle \Phi \rangle^2 \mathbf{C}^T \right) \right),
\] (A.19)

By Jensen’s inequality, we have that
\[
\langle \phi (\hat{y}_k) \rangle_{\hat{x}}^2 \leq \langle \phi (\hat{y}_k) \rangle_{\hat{x}}^2, \forall k = 1, \ldots, K_1.
\] (A.20)

Then it follows from (A.20) that inequality (2.96),
\[
\frac{1}{2} \ln \left( \det \left( \mathbf{C} \langle \Phi \rangle^2 \mathbf{C}^T \right) \right) \leq \frac{1}{2} \ln \left( \det \left( \mathbf{C} \langle \Phi \rangle \mathbf{C}^T \right) \right).
\] (A.21)

This completes the proof of Proposition 2.2.

A.5 Discussion on Algorithm 2

By Proposition 2.2 if \( K_0 = K_1 \) then we get
\[
\begin{align*}
\frac{1}{2} \langle \ln \left( \det \left( \mathbf{C} \langle \Phi \rangle \right) \right) \rangle_{\hat{x}} &\leq \frac{1}{2} \ln \left( \det \left( \langle \Phi \rangle \right) \right), \\
\langle \ln \left( \det \left( \mathbf{C} \langle \Phi \rangle \right) \right) \rangle_{\hat{x}} &\leq \ln \left( \det \left( \langle \Phi \rangle \right) \right) \\
&= \frac{1}{2} \ln \left( \det \left( \langle \Phi \rangle \right) \right) \\
&\leq \frac{1}{2} \ln \left( \det \left( \langle \Phi \rangle \right) \right), \\
\ln \left( \det \left( \mathbf{C} \langle \Phi \rangle \right) \right) &\leq \frac{1}{2} \ln \left( \det \left( \mathbf{C} \langle \Phi \rangle \right) \right).
\end{align*}
\] (A.22)

On the other hand, it follows from (2.77) and Proposition 2.2 that
\[
\begin{align*}
\langle \ln \left( \det \left( \mathbf{C} \mathbf{C}^T \right) \right) \rangle_{\hat{x}} &\leq -Q \leq \frac{1}{2} \ln \left( \det \left( \mathbf{C} \langle \Phi \rangle \mathbf{C}^T \right) \right), \\
\langle \ln \left( \det \left( \mathbf{C} \mathbf{C}^T \right) \right) \rangle_{\hat{x}} &\leq -\hat{Q} \leq \frac{1}{2} \ln \left( \det \left( \mathbf{C} \langle \Phi \rangle \mathbf{C}^T \right) \right).
\end{align*}
\] (A.27) (A.28)
Hence we can see that \( \hat{Q} \) is close to \( Q \) (see \( 2.77 \)). Moreover, it follows from the Cauchy–Schwarz inequality that

\[
\langle (\Phi)_{k,k} \rangle_{\hat{x}} = \langle \phi(\hat{y}_k) \rangle_{\hat{y}_k} \leq \left( \int \phi(\hat{y}_k)^2 d\hat{y}_k \int p(\hat{y}_k)^2 d\hat{y}_k \right)^{1/2},
\]

where \( k = 1, \ldots, K_1 \), the equality holds if and only if the following holds,

\[
p(\hat{y}_k) = \frac{\phi(\hat{y}_k)}{\int \phi(\hat{y}_k) d\hat{y}_k}, \quad k = 1, \ldots, K_1,
\]

which is the similar to Eq. \( 2.60 \).

Since \( I(X;R) = I(Y;R) \) (see Proposition \( 2.1 \)), by maximizing \( I(X;R) \) we hope the equality in inequality \( 2.58 \) and equality \( 2.60 \) hold at least approximately. On the other hand, let

\[
C^{opt} = \arg \min_C Q[C] = \arg \max_C \left( \left\langle \ln \left( \det \left( C \hat{\Phi} \hat{C}^T \right) \right) \right\rangle_{\hat{x}} \right), \quad \hat{C}^{opt} = \arg \min_{\hat{C}} \hat{Q}[\hat{C}] = \arg \max_{\hat{C}} \left( \left\langle \ln \left( \det \left( C \langle \hat{\Phi} \rangle_{\hat{x}}^2 \hat{C}^T \right) \right) \right\rangle_{\hat{x}} \right),
\]

\( C^{opt} \) and \( \hat{C}^{opt} \) make \( 2.60 \) and \( A.30 \) to hold up, which implies that they are the same optimal solution: \( C^{opt} = \hat{C}^{opt} \).