Analyzing Feature Extraction by Contrastive Divergence Learning in RBMs

Ryo Karakida[1] **Masato Okada**[1,2] **Shun-ichi Amari**[2]
[1]The University of Tokyo, Chiba, Japan, [2]RIKEN BSI, Saitama, Japan
{karakida@mns.|okada@}k.u-tokyo.ac.jp, amari@brain.riken.jp

Abstract

The restricted Boltzmann machine (RBM) is an essential constituent of deep learning but it is hard to train by using maximum likelihood (ML) learning, which minimizes the Kullback-Leibler (KL) divergence. Instead, contrastive divergence (CD) learning is widely used in practice. To clarify the performance of CD learning, one can compare solutions extracted by ML learning with those extracted by CD_1, and in general, CD_n learning. In this paper, we analytically derive equilibrium points of ML and CD_n learning rules in RBMs. As a result, We find that stable equilibrium points of CD_n learning coincide with those of ML learning in a Gaussian-Gaussian RBM. We also find that the ML and CD_n stable points in the Gaussian-Gaussian RBM correspond to the extraction of principal components whose eigenvalues are larger than a certain value. Moreover, we reveal that ML and CD_n learning in a Gaussian-Bernoulli RBM can extract independent components as one of their stable points. Our analysis demonstrates that the same stable solutions as extracted by ML learning are extracted simply by performing CD_1 learning. Expanding on this study should be a way to elucidate the specific features extracted by CD learning in other types of RBMs or in deep networks consisting of RBMs.

1 Introduction

The restricted Boltzmann machine (RBM) is a bipartite graphical model whose visible and hidden units are conditionally independent of each other [1, 2]. Contrastive divergence (CD) learning, an approximate algorithm of maximum likelihood (ML) learning, makes efficient use of the conditional independence of RBMs [2]. If ML learning is used to train an RBM, it requires many iterations of Gibbs sampling at each update step and takes too much computational time. In contrast, CD learning requires only a few iterations of Gibbs sampling and completes the training in a short time. CD_n learning uses n steps of Gibbs sampling. As evidenced empirically, an RBM trained by CD_1 learning has solutions that are close to those of an RBM trained by ML learning [3]. CD learning has become a standard method for deep learning, and stacked RBMs pretrained by CD learning have performed well in practical applications [4, 5, 6].

It is important to know what differences exist between ML and CD learning. The previous theoretical studies have demonstrated the properties of convergence and equilibrium solutions as regards CD learning. For instance, there are certain cases where CD learning does not converge [7]. Even if the learning procedure converges to equilibrium points, these points do not necessarily maximize the likelihood function [3]. In particular, Williams and Agakov gained theoretical insights into the RBM with continuous units and demonstrated that the gradients of CD learning are biased in comparison with those of ML learning [8]. Although it is unknown how biased the solutions obtained by the CD learning are in general, there are nonetheless special conditions under which ML and CD learning give the same solution [9, 10, 11].

In contrast with the previous studies, a question left unanswered is what features are commonly or differently extracted by ML and CD learning? For using CD learning in practice, it is important to clarify the specific features extracted by ML, CD_n, and, in particular, the widely used CD_1. Here, the way to identify the features extracted by a learning rule is equilibrium analysis [12]. Analysis of the equilibrium points and their stability have clarified features that can be extracted by a number of learning rules, including principal or minor components extracted in neural networks [13, 14, 15], principal components extracted by ML learning in the probabilistic PCA model [16], and independent components extracted by ICA learning rules [17, 18]. If it is possible to carry out similar equilibrium analysis on CD learning, we can identify the features extracted by CD learning.

In this study, we used the equilibrium analysis to identify the features extracted by ML and CD_n learning in RBMs. First, we derived an analytical form of equilibrium points extracted by ML learning in a Gaussian-Gaussian RBM whose visible and hidden units are continuous [8]. The analytical form demonstrated that ML learning extracts principal components whose eigenvalues are larger than a certain value. In addition, we analyzed their stability by a perturbation method and revealed that stable points correspond to extract as many components as the model allows. Next, we derived analytical form of equilibrium points extracted by CD_n learning and found that it coincides with that of ML learning. In addition, their stability also coincides with that of ML learning. We thus concluded that CD_n learning extracts the same principal components as ML learning. Moreover, we also apply the same analytical methods to a Gaussian-Bernoulli RBM whose hidden units are binary [4, 19]. We revealed that both ML and CD_n learning in the Gaussian-Bernoulli RBM extract independent components as one of stable points under certain conditions.

The results for CD_n learning are independent of n. Because CD_1 can extract the same features as ML, CD_1 seems to be efficient to train the RBMs. Expanding our analysis would help to elucidate features that can be extracted in RBMs with binary visible units or stacked RBMs.

2 Model

2.1 Gaussian-Gaussian RBM

The probability distribution of a Gaussian-Gaussian RBM is defined as follows [8, 20]:

$$q(\mathbf{h},\mathbf{v}) = \exp\left(-\sum_{i=1}^{M}\frac{{h_i}^2}{2s_i^2} - \sum_{j=1}^{N}\frac{{v_j}^2}{2\sigma_j^2} + \sum_{i,j}W_{ij}\frac{h_i}{s_i}\frac{v_j}{\sigma_j} + \mathbf{b}^T\mathbf{v} + \mathbf{c}^T\mathbf{h} - \psi\right). \tag{1}$$

Both of hidden units h_i and visible units v_j are continuous and obey a Gaussian distribution characterized by a variance s_i^2 $(i = 1, ..., M)$ and σ_j^2 $(j = 1, ..., N)$. Although here we consider only the case of $\mathbf{b} = \mathbf{c} = 0$, we can analyze the general case in the same way. Furthermore, W denotes an $M \times N$ weight matrix and ψ is a normalization factor.

The joint distribution $q(\mathbf{h},\mathbf{v})$ yields the following conditional distributions: $q(\mathbf{h}|\mathbf{v}) = (\sqrt{2\pi}^M \Pi_i s_i)^{-1}\exp\left(-||S^{-1}\mathbf{h} - W\Sigma^{-1}\mathbf{v}||^2/2\right)$, $q(\mathbf{v}|\mathbf{h}) = (\sqrt{2\pi}^N \Pi_i \sigma_i)^{-1}\exp(-||\Sigma^{-1}\mathbf{v} - W^T S^{-1}\mathbf{h}||^2/2)$. Let us describe the squared norm of a vector by $||\cdot||^2$. We define the variance matrix of the hidden units by an $M \times M$ diagonal matrix $S = \mathrm{diag}(s_1, s_2, ..., s_M)$ and that of the visible units by an $N \times N$ diagonal matrix $\Sigma = \mathrm{diag}(\sigma_1, \sigma_2, ..., \sigma_N)$.

ML learning rule. The learning rule of maximum likelihood (ML) is derived by minimizing the Kullback-Leibler (KL) divergence between the input distribution and the model distribution [2],

$$\tau\frac{dW}{dt} = <\mathbf{h}\mathbf{v}^T>_0 - <\mathbf{h}\mathbf{v}^T>_q, \tag{2}$$

where τ is a learning constant. The positive phase is defined by $<\mathbf{h}\mathbf{v}^T>_0 = \int \mathbf{h}\mathbf{v}^T p(\mathbf{h}|\mathbf{v})p_0(\mathbf{v})d\mathbf{h}d\mathbf{v}$, where $p_0(\mathbf{v})$ is the input data distribution. In contrast, the negative phase is defined by $<\mathbf{h}\mathbf{v}^T>_q = \int \mathbf{h}\mathbf{v}^T q(\mathbf{h},\mathbf{v})d\mathbf{h}d\mathbf{v}$, which is the expectation with respect to the model distribution $q(\mathbf{h},\mathbf{v})$. In the case of the Gaussian-Gaussian RBM, the ML learning rule (2) is calculated explicitly as

$$\tau\frac{dW}{dt} = SW\Sigma^{-1}C - SW(I_N - W^TW)^{-1}\Sigma. \tag{3}$$

Let us denote the data covariance matrix estimated by the empirical average of the input by $C = \int \mathbf{v}\mathbf{v}^T p_0(\mathbf{v}) d\mathbf{v}$. I_N is an $N \times N$ identity matrix.

CD$_n$ learning rule. CD$_n$ learning uses a distribution obtained after alternating Gibbs sampling between the visible and hidden layers n times. The CD$_n$ learning rule is defined as follows:

$$\tau \frac{dW}{dt} = < \mathbf{h}\mathbf{v}^T >_0 - < \mathbf{h}\mathbf{v}^T >_n . \tag{4}$$

The positive phase $< \mathbf{h}\mathbf{v}^T >_0$ is the same as in the ML learning rule. The negative phase is defined by an iterated integral of conditional probability distributions, that is, $< \mathbf{h}\mathbf{v}^T >_n = \int \mathbf{h}_n \mathbf{v}_n^T q(\mathbf{h}_n|\mathbf{v}_n) q(\mathbf{v}_n|\mathbf{h}_{n-1}) \cdots q(\mathbf{h}_0|\mathbf{v}_0) p_0(\mathbf{v}_0) d\mathbf{h}_n d\mathbf{v}_n \cdots d\mathbf{h}_0 d\mathbf{v}_0$. The CD$_n$ learning rule for a Gaussian-Gaussian RBM is calculated to give a non-linear differential equation with the $(4n+1)$-th power of W [8],

$$\tau \frac{dW}{dt} = SW\Sigma^{-1}C - SW \left\{ (W^T W)^n \Sigma^{-1} C \Sigma^{-1} (W^T W)^n + \sum_{k=0}^{2n-1} (W^T W)^k \right\} \Sigma. \tag{5}$$

We found that the CD learning rule (5) corresponds to the ML learning rule (3) as follows. The condition that the model distribution $q(\mathbf{v})$ is a Gaussian is equivalent to the eigenvalues ε_i of $W^T W$ satisfying $0 \leq \varepsilon_i < 1$. Under this condition, we can apply a Neumann series expansion to the inverse matrix of the ML learning, that is, $(I_N - W^T W)^{-1} = \sum_{k=0}^{\infty} (W^T W)^k$. The negative phase of the CD learning rule consists of a Neumann series terminated at the $(2n-1)$-th term and the $4n$-th power of W, $(W^T W)^n \Sigma^{-1} C \Sigma^{-1} (W^T W)^n$. In particular, the CD$_n$ learning rule converges to the ML learning rule when $n \to \infty$. This means that an infinite number of iterations of Gibbs sampling correspond to ML learning.

In spite of the non-linearity of the learning rule (3) and (5), in section 3, we will derive an analytical form of the equilibrium points and show their stability.

2.2 Gaussian-Bernoulli RBM

The Gaussian-Bernoulli RBM has binary hidden units $h_i = \{0, 1\}$ and Gaussian visible units v_i, whose probability distribution is defined as follows [4, 19]:

$$q(\mathbf{h}, \mathbf{v}) = \exp\left(-\sum_{j=1}^{N} \frac{{v_j}^2}{2\sigma_j^2} + \sum_{i,j} W_{ij} h_i \frac{v_j}{\sigma_j} + \mathbf{b}^T \mathbf{v} + \mathbf{c}^T \mathbf{h} - \psi \right). \tag{6}$$

The joint distribution (6) yields the following conditional distributions: $q(\mathbf{h}|\mathbf{v}) = \prod_i \exp(h_i \mathbf{w_i} \Sigma^{-1} \mathbf{v}) / \{1 + \exp(\mathbf{w_i} \Sigma^{-1} \mathbf{v})\}$, $q(\mathbf{v}|\mathbf{h}) = (\sqrt{2\pi}^N \prod_i \sigma_i)^{-1} \exp\left(-||\Sigma^{-1}\mathbf{v} - W^T \mathbf{h}||^2/2\right)$.

ML learning rule. Assuming $\mathbf{b} = \mathbf{c} = \mathbf{0}$ for simplicity, we obtain the ML learning rule,

$$\tau \frac{d\mathbf{w}_i}{dt} = < h_i \mathbf{v}^T >_0 - \sum_{x=1}^{M} k_{ix} \mathbf{w}_x \Sigma. \tag{7}$$

We obtain the positive term such that $< h_i \mathbf{v}^T >_0 = \int g(\mathbf{w}_i \Sigma^{-1} \mathbf{v}) \mathbf{v}^T p_0(\mathbf{v}) d\mathbf{v}$, where we denote a sigmoid function by $g(\cdot)$ and the i-th row of W by $\mathbf{w}_i$. The notation k_{ix} in the negative term represents the interaction between hidden units, i.e., $k_{ix} = \sum_{\mathbf{h}} h_i h_x \exp(||W^T \mathbf{h}||^2/2) / \sum_{\mathbf{h}} \exp(||W^T \mathbf{h}||^2/2)$.

CD$_n$ learning rule. Regarding the CD learning rule (4), the positive phase is the same as in the ML learning rule. The negative phase $< h_i \mathbf{v}^T >_n$ is difficult to formulate explicitly in terms of the weight matrix W:

$$\tau \frac{d\mathbf{w}_i}{dt} = < h_i \mathbf{v}^T >_0 - \int g(\mathbf{w}_i \Sigma^{-1} \mathbf{v}_n) \mathbf{v}_n^T \prod_{k=0}^{n-1} q(\mathbf{v}_{k+1}|\mathbf{h}_k) q(\mathbf{h}_k|\mathbf{v}_k) d\mathbf{v}_{k+1} d\mathbf{h}_k p(\mathbf{v}_0) d\mathbf{v}_0. \tag{8}$$

Although it is difficult to derive general solutions of learning rules (7) and (8), we will obtain one of equilibrium points under certain conditions.

3 Principal Component Extraction in Gaussian-Gaussian RBM

The previous section gave the ML learning rule (3) and CD$_n$ learning rule (5) for the Gaussian-Gaussian RBM. These learning rules are non-linear differential equations of the weight matrix W and are difficult to solve analytically. Here, let us assume that the variances of the visible and hidden units are homogeneous, i.e., $\Sigma = \sigma I_N$ and $S = sI_M$. In this case, we can demonstrate that it is possible to derive an analytical form of the equilibrium points and prove their stability. Our analysis proves that both ML and CD$_n$ learning extract principal components.

3.1 ML Solutions

By setting $dW/dt = 0$ in the ML learning rule (3), we obtain the equation of the equilibrium,

$$WC = \sigma^2 W(I_N - W^T W)^{-1}. \tag{9}$$

We can derive the following theorem about the equilibrium points W satisfying eq. (9). As preparation, let us denote a singular value decomposition by $W = UAR^T$, where U is an $M \times M$ orthogonal matrix, A is an $M \times N$ diagonal matrix, and R is an $N \times N$ orthogonal matrix.

Theorem 1.1. *Assume that the covariance matrix C has non-degenerate eigenvalues λ_i $(i = 1, ..., N)$ and that there exist k eigenvalues satisfying $\lambda_i > \sigma^2$ $(i = 1, ..., k)$. A necessary and sufficient condition for $W = UAR^T$ to satisfy eq. (9) is 1) U is an arbitrary $M \times M$ orthogonal matrix, 2) A is an $M \times N$ diagonal matrix $A = \mathrm{diag}\left(\sqrt{1 - \frac{\sigma^2}{\lambda_1}}, ..., \sqrt{1 - \frac{\sigma^2}{\lambda_m}}, 0, ..., 0\right)$, where m eigenvalues are chosen from $\lambda_i > \sigma^2$ $(0 \leq m \leq k)$, and 3) R^T is an $N \times N$ orthogonal matrix V that diagonalizes the covariance matrix such that $C = V^T \mathrm{diag}(\lambda_1, ..., \lambda_N)V$.*

Proof. Without loss of generality, we may assume A to be an $M \times N$ diagonal matrix $A = \mathrm{diag}\,(\alpha_1, ..., \alpha_m, 0, ..., 0)$, where we denote the rank of W as m and the singular value of W as α_i. Substituting the singular value decomposition $W = UAR^T$ into eq. (9), we get $A\left\{R^T CR(I_N - A^T A) - \sigma^2 I_N\right\} = O$. This equation needs $R^T CR$ to become $R^T CR = \begin{bmatrix} \mathrm{diag}\left(\frac{\sigma^2}{1-\alpha_1^2}, ..., \frac{\sigma^2}{1-\alpha_m^2}\right) & O \\ O & Q \end{bmatrix}$. Note that Q is an $(N-m) \times (N-m)$ symmetric matrix. Diagonalizing Q by using an orthogonal matrix P, $Q = P\mathrm{diag}(Q)P^T$, we obtain

$$C = \left(R \begin{bmatrix} I_m & O \\ O & P \end{bmatrix}\right) \begin{bmatrix} \mathrm{diag}\left(\frac{\sigma^2}{1-\alpha_1^2}, ..., \frac{\sigma^2}{1-\alpha_m^2}\right) & O \\ O & \mathrm{diag}(Q) \end{bmatrix} \left(R \begin{bmatrix} I_m & O \\ O & P \end{bmatrix}\right)^T. \tag{10}$$

Here, we diagonalize $C = V^T \mathrm{diag}(\lambda_1, ..., \lambda_n)V$ and assume that the eigenvalues of C are non-degenerate. It is necessary for R^T to become $R^T = \begin{bmatrix} I_m & O \\ O & P \end{bmatrix} V$. In addition, it is necessary for α_i to be $\alpha_i = \sqrt{1 - \sigma^2/\lambda_i}$ if $\lambda_i > \sigma^2$. From the above, W turns out to be

$$W = UAR^T = U \begin{bmatrix} \mathrm{diag}\left(\sqrt{1 - \frac{\sigma^2}{\lambda_1}}, ..., \sqrt{1 - \frac{\sigma^2}{\lambda_m}}\right) & O \\ O & O \end{bmatrix} \begin{bmatrix} I_m & O \\ O & P \end{bmatrix} V = UAV. \tag{11}$$

Moreover, the sufficient condition that the derived $W = UAV$ satisfies eq. (9) is verifiable. Therefore, $W = UAV$ is a necessary and sufficient condition for W to satisfy eq. (9). □

The analytical form $W = UAV$ shown in Theorem 1.1 claims that ML learning extracts principal component vectors corresponding to large eigenvalues $\lambda_i > \sigma^2$ $(i = 1, ..., m)$. The extracted vectors are scaled by $\sqrt{1 - \sigma^2/\lambda_i}$ and rotated by the orthogonal matrix U. It is remarkable that the principal components extracted by ML learning are characterized by the model variance of the visible units σ^2, not that of the hidden units s^2.

If one substitutes $W = UAV$ into the model distribution $q(\mathbf{h}, \mathbf{v})$, one can get the model distribution of the visible units $q(\mathbf{v})$ as follows:

$$q(\mathbf{v}) = \frac{1}{(\sqrt{2\pi}\sigma)^{N-m}\sqrt{\lambda_1 \cdots \lambda_m}} \exp\left(-(V\mathbf{v})^T \mathrm{diag}(\lambda_1, ..., \lambda_m, \sigma^2, ..., \sigma^2)^{-1}(V\mathbf{v})/2\right). \tag{12}$$

Let us define η_i as the eigenvalues of the model covariance $\int \mathbf{v}\mathbf{v}^T q(\mathbf{v})d\mathbf{v}$. The model distribution (12) leads to $\eta_i = \lambda_i (i = 1, ..., m), \quad \eta_i = \sigma^2 (i = m+1, ..., N)$. We found that the smaller eigenvalues λ_i of the input covariance C are replaced by the model variance σ^2.

Note that the equilibrium points W have rotational degrees of freedom caused by U and another degrees of freedom, m $(0 \leq m \leq k)$, which means the rank of W and how many principal components are extracted.

3.2 Stability of ML solutions

We derived the analytical form of the equilibrium points, $W = UAV$, for the ML learning rule. However, they include not only stable points but also unstable points that the learning never converges to. A theoretical perturbation calculation around the equilibrium points leads to a necessary and sufficient condition for stability.

Theorem 1.2. *In the case of $k \leq M$, the equilibrium points $W = UAV$ obtained in Theorem 1.1 are stable if and only if $m = k$.*

Proof. Let us denote the gradient of the ML learning as $F(W) \equiv WC - \sigma^2 W(I_N - W^T W)^{-1}$. In order to prove that an equilibrium is stable, we should show that the inner product between any perturbation $\Delta W^{(ij)}$ and ΔF is negative around the equilibrium W, where $\Delta F \equiv F(W + \Delta W^{(ij)})$. Let us define the perturbation as $\Delta W^{(ij)} = \varepsilon U E^{(ij)} V$, where ε is an infinitesimal change and entries of $M \times N$ matrix $E^{(ij)}$ are zero except for the i-th row and j-th column entry $E^{(ij)}_{ij} = 1$. The matrix $\Delta W^{(ij)}$ means perturbing the i-th row of W by the j-th eigenvector of the covariance matrix C on the coordinate rotated by U. Although we omitted the derivation of the inner product, it is expressed by the following trace:

$$\mathrm{Tr}(\Delta W^{(ij)T} \Delta F) \quad = \quad \left[\lambda_j - \frac{\sigma^2}{(1-\alpha_i^2)(1-\alpha_j^2)} - \frac{\sigma^2 \alpha_i^2}{(1-\alpha_i^2)^2} \delta_{ij} \right] \varepsilon^2. \tag{13}$$

The notation α_i denotes $\alpha_i = \sqrt{1 - \sigma^2/\lambda_i}$ $(i = 1, ..., m)$, 0 $(i = m+1, ..., N)$.

Let us denote the coefficient of the inner product by $\phi^{(ij)} \equiv \mathrm{Tr}(\Delta W^{(ij)T} \Delta F)/\varepsilon^2$. We should show that $\phi^{(ij)}$ is negative for any indexes i and j. In the case of $m < k$, if $m < i$ and $m < j \leq k$, the inner product becomes $\phi^{(ij)} = \lambda_j - \sigma^2 > 0$. This means that such an equilibrium point is unstable. Therefore, $m = k$ is necessary. In the case of $m = k$, (i) if $i, j \leq k$, the inner product is $\phi^{(ij)} = \lambda_j(1 - \lambda_i/\sigma^2) + \lambda_i(1 - \lambda_i/\sigma^2)\delta_{ij} < 0$. This case is stable. (ii) if $i \leq k < j$, $\phi^{(ij)} = \lambda_j - \lambda_i < 0$. This case is stable. (iii) if $j \leq k < i$, $\phi^{(ij)} = 0$. This case is neutral, and this is caused by the rotational degree of freedom due to U. (iv) if $k < i, j$, $\phi^{(ij)} = \lambda_j - \sigma^2 < 0$. This case is stable. Therefore, the stable points are those $W = UAV$ satisfying $m = k$. The other points with $m < k$ are unstable. □

As indicated by Theorem 1.2, the ML learning converges to the stable equilibrium points $W = UAV$ with rank $m = k$ and extracts all of the eigenvalues larger than σ^2.

In the case of $M < k$, where the number of hidden units is smaller, we can also prove stability in the same manner as above. In this case, an equilibrium is stable if and only if $m = M$. In addition, such stable equilibrium with $m = M$ extracts the largest M principal components, which correspond to the largest M eigenvalues among λ_i $(i = 1, ..., k)$. This means that the Gaussian-Gaussian RBM can extract only the largest principal components by reducing the number of hidden units.

3.3 CD$_n$ solutions

In the same way as shown for ML learning, we can also derive an analytical form for the equilibrium points of CD$_n$ learning. We get the following equation in the case of setting $dW/dt = 0$ in the CD$_n$ learning rule (5),

$$WC = W \left\{ (W^T W)^n C (W^T W)^n + \sigma^2 \sum_{k=0}^{2n-1} (W^T W)^k \right\}. \tag{14}$$

At this point, we can derive the following theorem about the equilibrium points satisfying eq. (14).

Theorem 2.1. *Assume that the covariance matrix C has non-degenerate eigenvalues* λ_i $(i = 1, ..., N)$ *and that there exist* k *eigenvalues satisfying* $\lambda_i > \sigma^2$ $(i = 1, ..., k)$. *A necessary and sufficient condition for* $W = UAR^T$ *to satisfy eq. (14) is 1)* U *is an arbitrary* $M \times M$ *orthogonal matrix, 2)* A *is an* $M \times N$ *diagonal matrix* $A = \mathrm{diag}\left(\sqrt{1 - \frac{\sigma^2}{\lambda_1}}, ..., \sqrt{1 - \frac{\sigma^2}{\lambda_m}}, 0, ..., 0\right)$, *where* m *eigenvalues are chosen from* $\lambda_i > \sigma^2$ $(0 \leq m \leq k)$, *and 3)* R^T *is an* $N \times N$ *orthogonal matrix* V *that diagonalizes the covariance matrix such that* $C = V^T \mathrm{diag}(\lambda_1, ..., \lambda_N)V$.

Proof. We omit the proof of Theorem 2.1 because we can prove it in a similar way to Theorem 1.1. □

Theorems 1.1 and 2.1 clarify that the equilibrium points of ML and CD_n learning have the same analytical form: $W = UAV$. Namely, the whole set of equilibrium points including stable and unstable points in CD_n learning coincides with that of ML learning.

3.4 Stability of CD_n solutions

We can also derive a necessary and sufficient condition for the stability of equilibrium points in CD_n learning as follows.

Theorem 2.2. *In the case of* $k \leq M$, *the equilibrium points* $W = UAV$ *obtained in Theorem 2.1 are stable if and only if* $m = k$.

Proof. We can prove the theorem by using the same process as shown in Theorem 1.2. Calculating the inner product $\phi^{(ij)} = \mathrm{Tr}(\Delta W^{(ij)T} \Delta F)/\varepsilon^2$, we obtain

$$(\lambda_j - \frac{\sigma^2}{1 - \alpha_j^2})(1 - \alpha_j^{4n}) - \alpha_i^2[(\alpha_j^{2n}\lambda_j + \alpha_i^{2n}\lambda_i)D_n + \sigma^2 \sum_{k=1}^{2n-1} D_k + (2\lambda_i \alpha_i^{2n} D_n + \sigma^2 \sum_{k=1}^{2n-1} D_k)\delta_{ij}], \tag{15}$$

where $D_k = \sum_{l=0}^{k-1} \alpha_i^{2(k-1-l)} \alpha_j^{2l}$. This inner product is negative for any i and j if and only if $m = k$. □

In the case of $M < k$, we can prove that an equilibrium is stable if and only if $m = M$ and that the stable equilibrium with $m = M$ has the largest M principal components.

Interestingly, as we can see from Theorems 1.2 and 2.2, the analytical form of the stable points in CD_n learning coincide with those of ML learning. This means that the stable points of CD_n learning maximize the likelihood in the Gaussian-Gaussian RBM. In terms of the model distribution of the visible units (12), $q(\mathbf{v})$ of ML learning coincides with that of CD_n at the stable points.

Note that the stable analytical form of CD_n learning is independent of n, the number of Gibbs sampling. Therefore, it is possible to extract the same principal components as in ML learning simply by performing CD_1 learning.

3.5 Related work

The following two models of principal component analysis (PCA) have similar equilibrium points of learning rules to Gaussian-Gaussian RBM.

Probabilistic PCA (PPCA) is a latent variable model of PCA, which gives the conditional distribution of the visible variables when the hidden variables are known [16]. The ML learning rule of PPCA is $dW/dt = W(\sigma^2 I_N + W^T W)^{-1} C(\sigma^2 I_N + W^T W)^{-1} - W(\sigma^2 I_N + W^T W)^{-1}$. Tipping and Bishop proved that the equilibrium points of PPCA are $W = U\mathrm{diag}\left(\sqrt{\lambda_1 - \sigma^2}, ..., \sqrt{\lambda_m - \sigma^2}, 0, ..., 0\right)V$, and they also proved the stability of the equilibrium points [16]. ML learning on PPCA is similar to that of Gaussian-Gaussian RBM with regard to the extraction of principal components whose eigenvalues are larger than the model variance σ^2. However, the singular values extracted using PPCA are different from those of Gaussian-Gaussian RBM.

Oja's subspace algorithm is a method to carry out PCA in a two-layer neural network. Its learning rule is $dW/dt = WC - WCW^T W$ [13, 21, 22]. The subspace algorithm extracts principal components as stable equilibrium points [23]. In this respect, the subspace algorithm is similar to the Gaussian-Gaussian RBM but different in that the singular values of W extracted by the subspace algorithm are equal to 1.

4 Independent Component Extraction in Gaussian-Bernoulli RBM

The equilibrium analysis shown in the previous section can be extended for the Gaussian-Bernoulli RBM. Here, let us assume the number of units $N = M$ and homogeneous variances $\Sigma = \sigma I_N$. In this case, we found the following sufficient condition for stable equilibrium points of ML and CD$_n$ learning in the Gaussian-Bernoulli RBM.

Theorem 3. *Assume that 1) the input distribution $p_0(\mathbf{v})$ is generated from independent source signals such that $\mathbf{v} = B\mathbf{s}$ and $p(\mathbf{s}) = p(s_1)p(s_2)\cdots p(s_N)$. The mixing matrix B is an $N \times N$ orthogonal matrix. 2) The source signals are non-negative $s_i \geq 0$ whose means $\mu_i = \int s_i p(s_i) ds_i$ satisfy $\mu_i \gg \sigma$. Under these assumptions, a sufficient condition for stable equilibrium points of both ML and CD$_n$ learning is $W = DB^T$, where $D = \sigma^{-1}\mathrm{diag}(\mu_1, \mu_2, ..., \mu_N)$.*

Proof. We can obtain $dW/dt = 0$ under the assumption 1) and 2), if we substitutes the above $W = DB^T$ into the ML learning rule (7) or the CD$_n$ learning rule (8). Therefore, $W = DB^T$ is a sufficient condition for equilibrium points.
In addition, we can prove the stability of $W = DB^T$ in a similar process as shown in Theorem 1.2. Because we obtain the negative inner product $\phi^{(ij)} = -1 < 0$ for any i and j in both ML and CD$_n$ learning, $W = DB^T$ is a stable point. □

The stable point $W = DB^T$ corresponds to extraction of independent components. It is equivalent to the unmixing matrix which is obtained by independent component analysis (ICA) [17, 18]. Namely, $W = DB^T$ separates the input data $\mathbf{v}$ into the independent source signals $\mathbf{s}$ such as $W\mathbf{v} = DB^T B\mathbf{s} = D\mathbf{s}$. In addition, each hidden unit h_i becomes the detector of each independent source s_i as is confirmed by a conditional distribution $q(h_i = 1|\mathbf{v}) = \mathrm{sigmoid}(\sigma^{-2}\mu_i s_i)$.

5 Experiments

5.1 Gaussian-Gaussian RBM

Here, we show that the results of the simulation on the Gaussian-Gaussian RBM agree with the theorems shown in the previous section very well.

Figure 1A shows that the Gaussian-Gaussian RBM trained with ML learning extracted principal components corresponding to $\lambda_i > \sigma^2$. We set the number of units to $N = M = 10$ and the model variance to $\sigma^2 = s^2 = 1$. We used a 10-dimensional Gaussian distribution as the input distribution $p_0(\mathbf{v})$ with zero mean and covariance matrix $\mathrm{diag}(0.2, 0.4, ..., 2)$. The black points in Figure 1A presents the mean and standard deviation of the extracted eigenvalues η_i in the simulation. Note that we estimated η_i from 20 batch inputs, where each batch input consists of $[\mathbf{v}^{(1)}, \mathbf{v}^{(2)}, ..., \mathbf{v}^{(1000)}]$ generated from $p_0(\mathbf{v})$. The red points in the figure show the theoretical values of η_i, and they

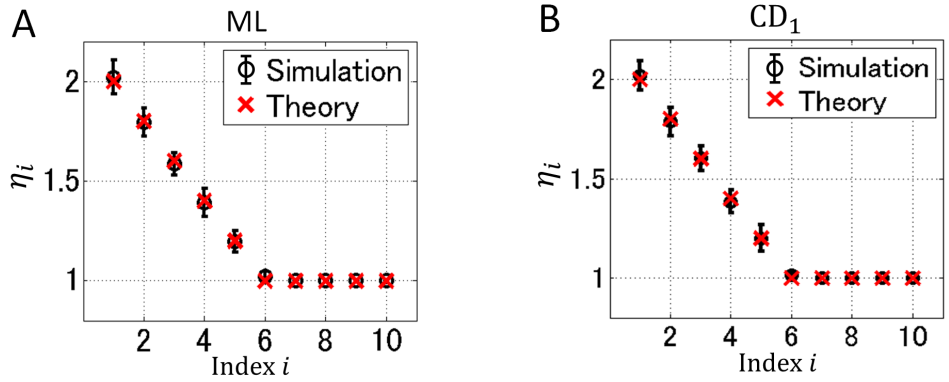

Figure 1: **Extraction of principal components in Gaussian-Gaussian RBM.** (A) Let us define eigenvalues extracted by ML learning as η_i. The black points are the η_i in the simulation ($N = M = 10$, input eigenvalues $\lambda_i = \{0.2, 0.4, ..., 2\}$, model variance $\sigma^2 = s^2 = 1$). The red points are the η_i expected from theory. (B) The eigenvalues η_i extracted by CD$_1$ learning using the same model and input data as used in the case of ML learning.

match the values in the simulation. As in section 3.1, ML learning extracted only larger eigenvalues $\eta_i = \lambda_i > \sigma^2$. In addition, the smaller eigenvalues corresponding to $\lambda_i < \sigma^2$ were replaced with the model variance such that $\eta_i = \sigma^2$. Remarkably, all of the larger principal components satisfying $\lambda_i > \sigma^2$ were extracted because they corresponded to the stable equilibrium points in Theorem 1.2.

Figure 1B presents the results of CD_1 learning on the same model and input data as used on ML learning. CD_1 learning in the simulation extracted the principal components as expected from Theorems 2.1 and 2.2. It is remarkable that the CD_1 learning results shown in figure 1B extracted the same principal components as ML learning did in figure 1A. The negative phase of ML learning takes a long time to run many iterations of Gibbs sampling. In contrast, CD_1 learning only needs one step of Gibbs sampling and saves computational time. In the simulation, CD_1 learning efficiently obtained the same principal components and model distribution $q(\mathbf{v})$ as ML learning did.

5.2 Gaussian-Bernoulli RBM

We show the simulation results that the Gaussian-Bernoulli RBM extracts independent components as is claimed in Theorem 3.

Figure 2 presents the results of CD_1 learning in the Gaussian-Bernoulli RBM with $N = M = 2$ and $\sigma = 1/2$. We set the independent source signals as non-negative uniform distributions $p(s_1)$ and $p(s_2)$ in Figure 2A. The input distribution $p_0(\mathbf{v})$ in Figure 2B was generated from $\mathbf{v} = B\mathbf{s}$. Note that $p_0(\mathbf{v})$ corresponds to the source distribution rotated by the orthogonal matrix B. After CD_1 learning, we obtained the stable solution W as is described in Theorem 3. Figure 2C shows the result of the CD_1 learning by an output distribution of $\mathbf{y} = W\mathbf{v} = D\mathbf{s}$. The output y_i became the source signal s_i scaled by μ_i/σ. Because the output decides the activation probability of the hidden unit by $q(h_i = 1|\mathbf{v}) = \mathrm{sigmoid}(\sigma^{-1}y_i)$, each hidden unit detects the independent source.

It should be noted that the stable point $W = DB^T$ proved in theorem 3 is not a necessary condition for the stable points but a sufficient one. There could be different stable points from $W = DB^T$. If we began the training from initial values of W enough close to DB^T, the matrix W converged to the solution $W = DB^T$ in simulation.

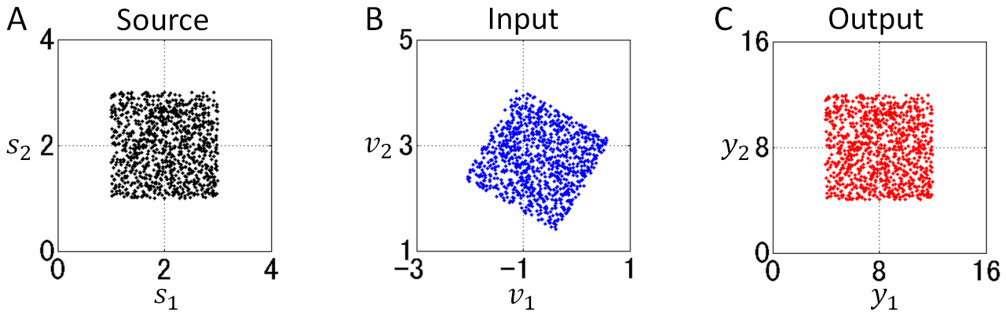

Figure 2: **Extraction of independent components by CD_1 learning in Gaussian-Bernoulli RBM.** (A) Uniform distribution of independent sources s_1 and s_2 ($N = M = 2$, $\sigma = 1/2$). (B) Distribution of input data $p_0(\mathbf{v})$ generated from $\mathbf{v} = B\mathbf{s}$. (C) Distribution of output $p(\mathbf{y})$ defined by $\mathbf{y} = W\mathbf{v}$, where W is obtained by CD_1 learning.

6 Conclusion and Future Work

In this paper, we analytically derived the equilibrium solutions of CD learning rule and carried out the stability analysis. As a result, we revealed that the Gaussian-Gaussian RBM trained with CD_n learning extracts principal components whose eigenvalues are larger than a certain. We also clarified that the analytical form and the stability of CD_n solutions coincide with those of ML solutions. Moreover, we revealed that the Gaussian-Bernoulli RBM extracts independent components by ML and CD_n learning as one of stable solutions. In particular, our analysis demonstrated that the same components extracted by ML learning are obtained simply by performing CD_1 learning.

By expanding our analytical method, we expect to identify features that are extracted by CD learning in other models such as RBMs with binary visible units or stacked RBMs. It would be also practically useful to investigate the equilibrium solutions in RBMs with a sparse prior [24] or rectified linear units [25].

References

[1] Smolensky, P. (1986). Information processing in dynamical systems: Foundations of harmony theory. *Parallel Distributed Processing*, vol. 1, chapter 6.

[2] Hinton, G. E. (2002). Training products of experts by minimizing contrastive divergence. *Neural Computation*, 14(8):1771–1800.

[3] Carreira-Perpinan, M. A. and Hinton, G. E. On contrastive divergence learning. In *Proc. Int'l Workshop on Artificial Intelligence and Statistics*, (2005).

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

[5] Salakhutdinov, R. and Hinton, G. E. Deep Boltzmann machines. In *Proc. Int'l Conf. on Artificial Intelligence and Statistics*, (2009).

[6] Bengio, Y., Courville, A., and Vincent, P. (2013). Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1798–828.

[7] Sutskever, I. and Tieleman, T. On the convergence properties of contrastive divergence. In *Proc. Int'l Conf. on Artificial Intelligence and Statistics*, (2010).

[8] Williams, C. and Agakov, F. (2002). An analysis of contrastive divergence learning in Gaussian Boltzmann machines. Technical Report EDI-INF-RR-0120, Inst. for Adaptive and Neural Computation, Univ. of Edinburgh.

[9] Yuille, A. The convergence of contrastive divergences. In *Proc. Neural Information and Processing Systems*, (2004).

[10] Movellan, J. R. (2008). Contrastive divergence in gaussian diffusions. *Neural Computation*, 20(9):2238–2252.

[11] Akaho, S. and Takabatake, K. Information geometry of contrastive divergence. In *Proc. Information Theory and Statistical Learning Workshop*, (2008).

[12] Amari, S.-I. (1977). Neural theory of association and concept-formation. *Biological Cybernetics*, 26(3):175–185.

[13] Oja, E. (1989). Neural networks, principal components, and subspaces. *International Journal of Neural Systems*, 1(01):61–68.

[14] Chen, T., Amari, S.-I., and Lin, Q. (1998). A unified algorithm for principal and minor components extraction. *Neural Networks*, 11(3):385–390.

[15] Chen, T. and Amari, S.-I. (2001). Unified stabilization approach to principal and minor components extraction algorithms. *Neural Networks*, 14(10):1377–1387.

[16] Tipping, M. E. and Bishop, C. M. (1999). Probabilistic principal component analysis. *Journal of the Royal Statistical Society: Series B*, 61(3):611–622.

[17] Amari, S.-I., Chen, T.-P., and Cichocki, A. (1997). Stability analysis of learning algorithms for blind source separation. *Neural Networks*, 10(8):1345–1351.

[18] Hyvärinen, A., Karhunen, J., and Oja, E. *Independent Component Analysis*. Wiley, (2001).

[19] Cho, K., Ilin, A., and Raiko, T. Improved learning of Gaussian-Bernoulli restricted Boltzmann machines. In *Proc. Int'l Conf. Artificial Neural Networks*. (2011).

[20] Hinton, G. E. (2010). A practical guide to training restricted Boltzmann machines. Technical Report 2010-003,Dept. of Computer Science, Univ. of Toronto.

[21] Williams, R. (1985). Feature discovery through error-correction learning.Technical Report 8501,UCSD, Institute of Cognitive Science.

[22] Chen, T., Hua, Y., and Yan, W.-Y. (1998). Global convergence of Oja's subspace algorithm for principal component extraction. *IEEE Transactions on Neural Networks*, 9(1):58–67.

[23] Oja, E. (1992). Principal components, minor components, and linear neural networks. *Neural Networks*, 5(6):927–935.

[24] Lee, H., Ekanadham, C., and Ng, A. Y. Sparse deep belief net model for visual area V2. In *Proc. Neural Information and Processing Systems*, (2007).

[25] Nair, V. and Hinton, G. E. Rectified linear units improve restricted boltzmann machines. In *Proc. Int'l Conf. on Machine Learning*, (2010).