

Ratio divergence learning using target energy in restricted Boltzmann machines: Beyond Kullback-Leibler divergence learning

Yuichi Ishida^{1,*}, Yuma Ichikawa^{1,2,†}, Aki Dote^{1,‡}, Toshiyuki Miyazawa^{1,§} and Koji Hukushima^{2,||}

¹*Fujitsu Ltd., Kawasaki, Japan*

²*Graduate School of Arts and Sciences, University of Tokyo, Tokyo, Japan*



(Received 9 October 2024; accepted 24 July 2025; published 7 October 2025)

We propose ratio divergence (RD) learning for discrete energy-based models, a method that utilizes both training data and a tractable target energy function. We apply RD learning to restricted Boltzmann machines (RBMs), which are a minimal model that satisfies the universal approximation theorem for discrete distributions. RD learning combines the strength of both forward and reverse Kullback–Leibler divergence (KLD) learning, effectively addressing the “notorious” issues of underfitting with the forward KLD and mode collapse with the reverse KLD. Since the summation of forward and reverse KLD seems to be sufficient to combine the strength of both approaches, we include this learning method as a direct baseline in numerical experiments to evaluate its effectiveness. Numerical experiments demonstrate that RD learning outperforms other learning methods in terms of energy function fitting, mode-covering, and learning stability across various discrete energy-based models. Moreover, the performance gaps between RD learning and the other learning methods become more pronounced as the dimensions of target models increase.

DOI: [10.1103/PhysRevE.112.045306](https://doi.org/10.1103/PhysRevE.112.045306)

I. INTRODUCTION

Discrete high-dimensional distributions, where the energy function is tractable but the normalization factor is not, appear in various fields, including condensed matter physics [1,2], Bayesian inference [3–5], and optimization [6]. For these applications, it is crucial to understand the statistical properties of these distributions and to draw samples from them efficiently. However, these tasks often present considerable computational issues, even when the energy function is tractable. Therefore, approximating target distributions using generative models, which can generate samples with low computational cost, has gained attention. This approach has been successfully applied in various *downstream tasks*, such as low-dimensional compression [7], discovery of critical clusters [8,9], enhancement of Markov chain Monte Carlo (MCMC) simulations [10–12], and molecular dynamics [13–15].

Learning methods for these generative models can be categorized into two types based on the Kullback–Leibler divergence (KLD): forward KLD learning, which employs only samples drawn from the target distribution, and reverse KLD

learning, which uses the target energy function. Nonetheless, Ciarella *et al.* [16] identified limitations in both the learning methods when applied to complex distributions with exponential peaks. While the forward KLD learning results in “underfitting” of the target energy function, particularly in the presence of multiple peaks, the reverse KLD learning results in “mode collapse,” even in simple probability distributions.

To address these issues, we propose a learning method using *ratio divergence (RD)*, a symmetrized divergence that naturally incorporates the product of the forward and reverse KLDs while eliminating the intractable normalization factor of the target energy. Although a linear combination of the forward and reverse KLDs might seem sufficient to address the issues, it is difficult to determine the coefficients. Additionally, minimizing RD has the advantage of increasing the acceptance probability of the Metropolis-Hastings algorithm in MCMC simulations, which is one of the downstream tasks.

Because RD learning requires both training data and an energy function, it may appear to be a limitation compared to purely data-driven or variational approaches. However, in many practical scenarios, for example, in physical models or optimization problems, a target energy model is often available. Moreover, the existence of a target energy model does not necessarily imply that generating samples from it is straightforward. Therefore, it is important to construct a surrogate model that can generate samples more efficiently than the original energy model. We then apply RD learning to the restricted Boltzmann machine (RBM) [17,18], a minimal model satisfying the universal approximation theorem [19] and employed for numerous downstream tasks [10,20,21].

Our numerical experiments demonstrate that RD learning more effectively approximates target distributions than the forward and reverse KLD learning methods in various

*Contact author: ishida-yuichi@fujitsu.com

†Contact author: ichikawa.yuma@fujitsu.com

‡Contact author: dote.aki@fujitsu.com

§Contact author: miyazawa.toshi@fujitsu.com

||Contact author: k-hukushima@g.ecc.u-tokyo.ac.jp

discrete models, such as the ferromagnetic Ising model, the Sherrington–Kirkpatrick (SK) model, the maximum independent set (MIS) problem, and the maximum cut problem (MCP) on various graphs. Particularly for the MIS, which is known for its exponential number of local optima and is noted by Ciarella *et al.* [16] that approximating the distribution is difficult using autoregressive generative models [22,23], RD learning achieves improved performance in terms of both energy fitting and mode covering abilities, compared to the forward and reverse KLD learning. This improvement arises from RD learning’s ability to combine the strengths of the two methods, overcoming the limitations of each method individually. Similar results are also observed for the MCPs on diverse graphs. Furthermore, our experiments show that the advantages of RD learning become more significant as the dimensionality of the target distribution increases, indicating its effectiveness in high-dimensional modeling tasks.

This paper is organized as follows. Section II introduces the structure of RBM and its learning methods based on forward and reverse KLDs. Section III discusses the practical issues with these learning methods and proposes a learning method based on RD, emphasizing its features. Section IV reviews other learning methods related to our approach. Section V presents numerical experiments, applying them to several example models to compare the learning methods based on KLDs and RD. Finally, the discussion and conclusion are provided in Sec. VI.

II. BACKGROUND

A. Problem setting

We begin by explaining the problem settings of this study. Let $\hat{P}(\mathbf{x}) = \exp(-\hat{E}(\mathbf{x}))/\hat{Z}$ represent the target discrete distribution over a discrete state space $\mathcal{X}$, where $\hat{Z} \triangleq \sum_{\mathbf{x} \in \mathcal{X}} \exp(-\hat{E}(\mathbf{x}))$ is the normalization factor. While computing the energy $\hat{E}(\mathbf{x})$ for a given $\mathbf{x} \in \mathcal{X}$ is relatively inexpensive, calculating the normalization factor $\hat{Z}$ and the expected value with respect to $\hat{P}(\mathbf{x})$ can be computationally challenging. In the following, let $\mathcal{D} = \{\mathbf{x}_\mu\}_{\mu=1}^M$, $\mathbf{x}_\mu \sim \hat{P}$ denote the training dataset with M being the number of the data. $\hat{P}_{\mathcal{D}}$ denotes the empirical distribution. Under these settings, this study aims to approximate this target distribution using a machine-learning model.

B. Restricted Boltzmann machines

A Boltzmann machine is an energy-based model consisting of two layers of binary variables: $\mathbf{x} \in \{0, 1\}^{N_x}$ and $\mathbf{h} \in \{0, 1\}^{N_h}$, which are referred to as visible and hidden layers, respectively. The probability distribution over these variables is given by

$$P(\mathbf{x}, \mathbf{h}; \boldsymbol{\theta}) = \frac{1}{Z(\boldsymbol{\theta})} e^{-E(\mathbf{x}, \mathbf{h}; \boldsymbol{\theta})}, \quad (1a)$$

$$Z(\boldsymbol{\theta}) \triangleq \sum_{\mathbf{x}, \mathbf{h}} e^{-E(\mathbf{x}, \mathbf{h}; \boldsymbol{\theta})}, \quad (1b)$$

where $E(\mathbf{x}, \mathbf{h}; \boldsymbol{\theta})$ represents the energy function of the model. To accelerate both training and inference, an energy function that includes interactions only between different layers, with

no interactions within the same layer, is known as a restricted Boltzmann machine [17,18]. Specifically, the energy function is defined as

$$E(\mathbf{x}, \mathbf{h}; \boldsymbol{\theta}) = -\mathbf{b}^\top \mathbf{x} - \mathbf{c}^\top \mathbf{h} - \mathbf{x}^\top W \mathbf{h}, \quad (2)$$

where $W \in \mathbb{R}^{N_x \times N_h}$, $\mathbf{b} \in \mathbb{R}^{N_x}$, and $\mathbf{c} \in \mathbb{R}^{N_h}$ are learning parameters collectively denoted as $\boldsymbol{\theta}$. Under this restriction, the marginal distribution $P(\mathbf{x}; \boldsymbol{\theta})$ is expressed as

$$P(\mathbf{x}; \boldsymbol{\theta}) = \frac{1}{Z(\boldsymbol{\theta})} e^{-F(\mathbf{x}; \boldsymbol{\theta})}, \quad (3a)$$

$$F(\mathbf{x}; \boldsymbol{\theta}) = -\mathbf{b}^\top \mathbf{x} - \sum_{m=1}^{N_h} \ln \left(1 + e^{c_m + \sum_{i=1}^{N_x} x_i W_{im}} \right). \quad (3b)$$

The learning method of RBMs can be categorized into two methods.

(i) *Forward KLD learning.* One learning approach involves training RBMs by minimizing the forward KLD defined by

$$D_{\text{KL}}[\hat{P}_{\mathcal{D}}(\mathbf{x}) \| P(\mathbf{x}; \boldsymbol{\theta})] = \mathbb{E}_{\mathcal{D}} \left[\ln \frac{\hat{P}_{\mathcal{D}}(\mathbf{x})}{P(\mathbf{x}; \boldsymbol{\theta})} \right], \quad (4)$$

where $\mathbb{E}_{\mathcal{D}}[f(\mathbf{x})] = M^{-1} \sum_{\mu=1}^M f(\mathbf{x}_\mu)$ denotes the average over the dataset. The gradients of the forward KLD with respect to the parameters are given by

$$\begin{aligned} & \frac{\partial D_{\text{KL}}[\hat{P}_{\mathcal{D}}(\mathbf{x}) \| P(\mathbf{x}; \boldsymbol{\theta})]}{\partial \boldsymbol{\theta}} \\ &= \mathbb{E}_{P(\mathbf{x}, \mathbf{h}; \boldsymbol{\theta})} \left[\frac{\partial E(\mathbf{x}, \mathbf{h}; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \right] - \mathbb{E}_{\mathcal{D}} \mathbb{E}_{P(\mathbf{h}|\mathbf{x})} \left[\frac{\partial E(\mathbf{x}, \mathbf{h}; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \right], \end{aligned} \quad (5)$$

where $\mathbb{E}_{P(\mathbf{h}|\mathbf{x})}[f(\mathbf{x}, \mathbf{h})] = \sum_{\mathbf{h}} f(\mathbf{x}, \mathbf{h}) p(\mathbf{h}|\mathbf{x})$ denotes the average over the hidden variables for given $\mathbf{x}$ and $\mathbb{E}_{P(\mathbf{x}, \mathbf{h}; \boldsymbol{\theta})}[f(\mathbf{x}, \mathbf{h})]$ denotes the average over the Boltzmann measure. Minimizing this divergence is computationally challenging due to the intractable summation of all variables on the right-hand side of Eq. (5). Therefore, these expectation values are approximated by the *self-samples* drawn from the Boltzmann measure in Eq. (1). One of the most widely used approximation methods is the persistent contrastive divergence (PCD) method [17,24], in which the right-hand side of the gradient is estimated after a few k steps of the block Gibbs sampling (BGS). In practice, the step is typically with $k = 1$. The final state of the BGS under the current parameters is then used as the initial state for the next BGS after the parameters are updated.

(ii) *Reverse KLD learning.* The other approach is minimizing the reverse KLD defined by

$$D_{\text{KL}}[P(\mathbf{x}; \boldsymbol{\theta}) \| \hat{P}_{\mathcal{D}}(\mathbf{x})] = \mathbb{E}_{P(\mathbf{x}; \boldsymbol{\theta})} \left[\ln \frac{P(\mathbf{x}; \boldsymbol{\theta})}{\hat{P}_{\mathcal{D}}(\mathbf{x})} \right]. \quad (6)$$

Puente and Eremin [25] proved that the gradients of the reverse KLD are equivalent to those of the following loss function:

$$\mathbb{E}_{P(\mathbf{x}; \boldsymbol{\theta})} \left[\frac{1}{2} (\hat{E}(\mathbf{x}) - F(\mathbf{x}) - C(\boldsymbol{\theta}))^2 \right], \quad (7a)$$

$$C(\boldsymbol{\theta}) = \mathbb{E}_{P(\mathbf{x}; \boldsymbol{\theta})} [(\hat{E}(\mathbf{x}) - F(\mathbf{x}))^2]. \quad (7b)$$

To compute this loss function and its gradients, these expectation values over the Boltzmann measure are

approximated by the self-samples drawn from the Boltzmann measure, just as when calculating the gradients of the forward KLD. Note that this method is only applicable when the target energy function is explicitly provided.

III. METHOD

In this section, we identify two practical issues associated with forward and reverse KLD learning. To address these challenges, we propose a learning method, *ratio divergence (RD) learning*, which we define and discuss below.

A. Practical issues of forward and reverse KLD learning

Underfitting in forward KLD learning. Forward KLD learning relies solely on the training dataset $\mathcal{D}$ and does not utilize the target energy function $\hat{E}(\mathbf{x})$ even when $\hat{E}(\mathbf{x})$ is provided. As a result, the regression performance for the target energy is not improved with training models. Indeed, Ciarella *et al.* [16] demonstrated that while forward KLD learning can cover many peaks of the target probability distribution $\hat{P}(\mathbf{x})$ (i.e., mode-covering), it fails to capture the detailed structure of these peaks, which adversely affects downstream tasks.

Mode collapse in reverse KLD learning. Reverse KLD learning relies solely on the target energy function $\hat{E}(\mathbf{x})$ and does not leverage the training dataset $\hat{\mathcal{D}}$. For this reason, once a high-probability sample is found, it becomes difficult to generate samples far from it. As a result, reverse KLD learning can approximate only one or a few peaks of the target probability distribution $\hat{P}(\mathbf{x})$. This phenomenon is a frequently observed issue in reverse KLD learning, commonly referred to as a mode collapse by Ciarella *et al.* [16].

B. Ratio divergence learning

To address these two issues, we propose a learning method that minimizes a symmetrized divergence, termed *ratio divergence*. RD effectively integrates both the forward and reverse KLDs as follows:

$$\mathcal{L}(\hat{P}, P; \theta) = \sum_{\mathbf{x}', \mathbf{x} \in \mathcal{X}} \hat{P}(\mathbf{x}') P(\mathbf{x}; \theta) \left(\log \frac{\hat{P}(\mathbf{x}') P(\mathbf{x}; \theta)}{P(\mathbf{x}'; \theta) \hat{P}(\mathbf{x})} \right)^2. \quad (8)$$

The proof that this function satisfies the condition of a symmetric divergence is provided in Appendix A 1. When training RBMs, Eq. (8) is approximated using the empirical distribution of the data and self-samples, similar to the gradient calculation of forward KLD.

This divergence is designed to incorporate the expectation of both the target distribution $\hat{P}(\mathbf{x})$ and the RBM distribution $P(\mathbf{x}; \theta)$ with equal weights as follows:

$$\begin{aligned} \mathcal{L}(\hat{P}, P; \theta) = & 2D_{\text{KL}}[\hat{P} \| P] D_{\text{KL}}[P \| \hat{P}] \\ & + D_{\text{KL}_2}[\hat{P} \| P] + D_{\text{KL}_2}[P \| \hat{P}], \end{aligned} \quad (9)$$

where

$$D_{\text{KL}_2}[\hat{P} \| P] \triangleq \sum_{\mathbf{x}} \hat{P}(\mathbf{x}) \left(\log \frac{\hat{P}(\mathbf{x})}{P(\mathbf{x}; \theta)} \right)^2, \quad (10)$$

which also satisfies the condition of probabilistic divergence. The first term in Eq. (9), representing the product of the

forward and reverse KLDs, aims to minimize both the forward and reverse KLDs. The second and third terms are constructed to ensure $\hat{P}(\mathbf{x}) \approx P(\mathbf{x}; \theta)$ for samples $\mathbf{x}$ drawn from both P and $\hat{P}$. This introduces implicit regularization to generate samples efficiently using BGS. Furthermore, RD, as a divergence defined based on the ratio of probability distributions, does not require computing the target normalizing factors $\hat{Z}$ or those of RBM $Z(\theta)$.

Moreover, this divergence has the following properties.

Theorem 1. For any discrete probability distributions $\hat{P}$ and P parametrized by θ , the following inequality is satisfied:

$$e^{-(\mathcal{L}(\hat{P}, P; \theta))^{1/2}} \leq \mathbb{E}_{\hat{P}(\mathbf{x}'), P(\mathbf{x}; \theta)} \left[\min \left(1, \frac{\hat{P}(\mathbf{x}') P(\mathbf{x}; \theta)}{P(\mathbf{x}'; \theta) \hat{P}(\mathbf{x})} \right) \right]. \quad (11)$$

The detailed derivation is provided in Appendix A 2. The right-hand side of the inequality can be regarded as the expected value of the acceptance probability in the Metropolis-Hastings algorithm, where $\hat{P}(\mathbf{x})$ represents the target distribution for a state $\mathbf{x}$, and $P(\mathbf{x}'; \theta)$ represents the proposal probability for a proposed state $\mathbf{x}'$. Consequently, minimizing $\mathcal{L}(\hat{P}, P; \theta)$ effectively maximizes the acceptance ratio. In contrast to both the forward and reverse KLD learning, the proposed divergence guarantees a relatively high acceptance rate for the samples generated by a surrogate model.

For simplicity, we adopt a quadratic function for the RD divergence; however, the formulation readily generalizes to any higher-order convex polynomial. Replacing the quadratic with a higher-degree polynomial preserves both the convexity and the location of the minimizer, altering only the sensitivity to the gradient of the loss function in the training process. Since an adaptive optimizer such as Adam can compensate for such sensitivity differences, the qualitative behavior would remain unchanged. A more comprehensive empirical evaluation across polynomial orders is left to future work.

IV. RELATED WORK

Extensive research has been conducted on training generative models using target energy functions. This section reviews relevant works related to the two main training methods: forward and reverse KLD learning. Additionally, we discuss training methods that use other divergences.

Forward KLD learning. McNaughton *et al.* [26] and Wu *et al.* [27] demonstrated the effectiveness of forward KLD learning in approximating complex probability distributions, even those with multiple modes. However, Ciarella *et al.* [16] pointed out that the forward KLD learning can lead to underfitting the target energy function due to its failure to incorporate it within the learning process. This limitation may impact the efficiency of simulations in various downstream tasks.

Reverse KLD learning. Huang and Wang [10] and Puente and Eremin [25] introduced reverse KLD learning in RBMs, significantly enhancing the efficiency of MCMC simulations for various problems. Additionally, Wu *et al.* [28] proposed the reverse KLD learning in autoregressive models (ARM), demonstrating its capability to generate samples efficiently from diverse statistical models and the small-scale

Sherrington–Kirkpatrick (SK) model without the need for a training dataset. This method also accelerated optimization algorithms for specific discrete optimization problems [29,30]. However, Ciarella *et al.* [16] and Inack *et al.* [31] highlighted a significant limitation: in high-dimensional but relatively simple probability distributions with multiple peaks, the reverse KLD learning can lead to mode collapse.

Other learning methods. In addition to forward and reverse KLD learning, several other methods have been proposed. Hyvärinen [32] proposed a learning method with a loss function that is a quadratic summation of the probability ratio, similar to our approach. However, unlike our methods, this approach does not incorporate the target energy function, potentially leading to underfitting, similar to the forward KLD learning. Furthermore, Midgley *et al.* [33] introduced α -divergence learning, where the divergence approximates the forward KLD as $\alpha \rightarrow 1$ and the reverse KLD as $\alpha \rightarrow 0$. In contrast to our methods, α -divergence learning requires calculating the normalization factor, which is computationally intensive for MCMC methods in general.

V. EXPERIMENTS

In this section, we numerically evaluate the performance of RD learning across four binary energy-based models: the 2D ferromagnetic Ising model, the Sherrington–Kirkpatrick (SK) model, the maximum independent set (MIS) problem, and the maximum cut problem (MCP). Each of them has distinct experimental significance. First, the Ising model is included as the simple case where the ground states are analytically well known and, in the simplest case, split into only two fundamental modes, all spins up or all spins down. Despite this apparent simplicity, existing methods often find it difficult to reproduce the complete distribution of states without bias, indicating that even a well-understood system can be difficult to generate samples. Next, in contrast to the simpler Ising model, the SK model exhibits replica symmetry breaking and can have multiple modes that depend on temperature. This added complexity provides a robust test of the ability of our method to handle systems with complex free energy landscapes. Furthermore, the MIS problem is known to produce complex distributions that are computationally difficult to approximate [16]. By demonstrating good performance in this context, we show the effectiveness of our method in tackling hard combinatorial tasks. Finally, the MCP benchmark extends our approach beyond strictly physical systems, indicating potential utility in broader combinatorial optimization problems.

All experiments included the forward and reverse KLD learning as direct baselines. Moreover, we add a method in which a loss function is a summation of the forward and reverse KLD to the direct baselines. Hereafter, this method is referred to as the summation KLD learning.

A. Setting

Evaluation metrics. To assess the quality of samples generated by the RBMs, we compared the empirical target energy distribution in the training dataset with that generated by the RBM trained using each learning method. The Wasserstein metric was used to measure the distance between these

empirical distributions quantitatively. For the detailed formulation of this metric, see Appendix B 2. Furthermore, for evaluating the regression performance over the validation dataset $\mathcal{D}_{\text{val}}$, we define the following error function:

$$R(\theta) = \frac{1}{|\mathcal{D}_{\text{val}}|^2} \sum_{\mathbf{x}', \mathbf{x} \in \mathcal{D}_{\text{val}}} (\Delta F(\mathbf{x}', \mathbf{x}; \theta) - \Delta \hat{E}(\mathbf{x}', \mathbf{x}))^2, \quad (12a)$$

$$\Delta F(\mathbf{x}', \mathbf{x}; \theta) = F(\mathbf{x}'; \theta) - F(\mathbf{x}; \theta), \quad (12b)$$

$$\Delta \hat{E}(\mathbf{x}', \mathbf{x}) = \hat{E}(\mathbf{x}') - \hat{E}(\mathbf{x}). \quad (12c)$$

At first glance, it seems that the average of $F(\mathbf{x}; \theta) - \hat{E}(\mathbf{x})$ is sufficient for evaluating regression performance. However, in most downstream tasks, the energy difference between two states is more important than the absolute value of the energy itself. Therefore, we introduce Eq. (12) as a mean squared error of the energy difference between two states.

We also conducted a principal component analysis (PCA) on both the generated samples and the training dataset to investigate mode collapse. We then visualized the generated samples in the two-dimensional space defined by the dominant principal components. Furthermore, we analyze the empirical distribution defined by $\{d_H(\mathbf{x}_\mu, \mathbf{x}_\nu)/N_x\}_{\mu < \nu}^K$, where d_H denotes the Hamming distance and K is the number of samples. This approach is commonly used as a diversity measure in combinatorial optimization problems [34,35].

Datasets. In all experiments, the training dataset, comprising 16 384 samples, and the validation dataset, comprising 1024 samples, were generated using the exchange MCMC simulations [36]. The exchange MCMC simulations were conducted for a sufficient duration to ensure sampling from the stationary distributions of these target energy-based models. For further details on these simulations, refer to Appendix B 1.

Implementation. The number of hidden variables is set to be equal to the number of visible variables for each target energy-based model. The size of the self-sample set is set equal to that of the training dataset. The Adam optimizer is employed with the following configurations: $\text{lr} = 0.001$, $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\epsilon = 1.0 \times 10^{-8}$, with the minibatch size set to 128 [37]. The model is trained for up to 1000 epochs. After training the RBMs, we generated 16 384 samples using BGS with 100 steps. Here, the training dataset is used as the initial condition because it is known that in many downstream tasks, using the training dataset as the initial condition stabilizes the sampling procedure and effectively shortens the burn-in period. Training and sampling were performed five times with different random seeds. To evaluate the empirical distribution of the Hamming distance, we randomly extracted 1000 samples from the samples generated by the RBMs and from the training dataset. All experiments were conducted on an Intel(R) Xeon(R) Gold 6240 CPU @ 2.60GHz with a Tesla V100.

B. 2D Ising model

To demonstrate the performance gaps between RD learning and KLD learning, even for a simple distribution, we consider the Ising model on a 2D square lattice graph, denoted as $G(V, E)$, whose energy function is defined by

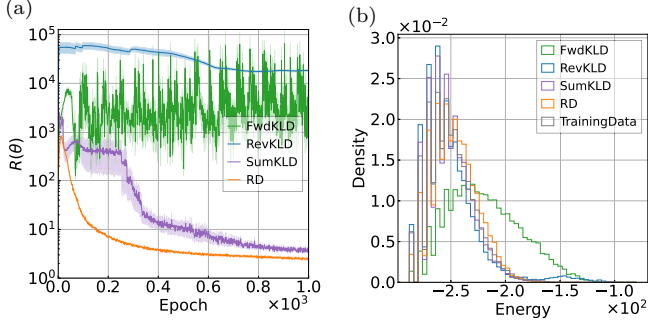


FIG. 1. (a) $R(\theta)$ as a function of epochs during the training process, and (b) empirical energy distributions of the generated samples by each method compared to those in the training dataset for the Ising model on a 12×12 square lattice.

$\hat{E}(\mathbf{x}) = -J \sum_{(i,j) \in E} x_i x_j$, where $\mathbf{x} \in \{-1, +1\}^{N_x}$ denotes the state configuration and J is a positive scalar representing the interaction strength between neighboring variables. This is the minimal model for ferromagnetic materials that exhibit a *phase transition*. As β increases, the data distribution transitions from nonclustered to two-clustered states. Here, we set the graph G to a 12×12 square lattice graph with periodic boundary conditions and J to 1. We show the performance of RBM at $\beta = 0.5$, corresponding to the two clustered states. The results are summarized as follows.

Figure 1(a) shows the evolution of $R(\theta)$ as functions of epochs during the training process. RD learning not only converges faster than the other learning methods by 1000 epochs but also demonstrates a more stable learning process over the epochs. This stability is one of the key advantages of RD learning. Moreover, Fig. 1(b) compares the empirical distribution of the generated samples with the target energy distributions from the training dataset, showing the accuracy of the learned distributions after training. The Wasserstein metric for RD learning is 1.6, which is lower than that of the forward KLD learning at 8.9, the reverse KLD learning at 9.8, and the summation KLD learning at 2.4. These results indicate that RD learning overcomes the other methods in terms of sampling performance.

We also examine the dimension-dependence of $R(\theta)$ at 1000 epochs in Fig. 2(a), and the Wasserstein metric in

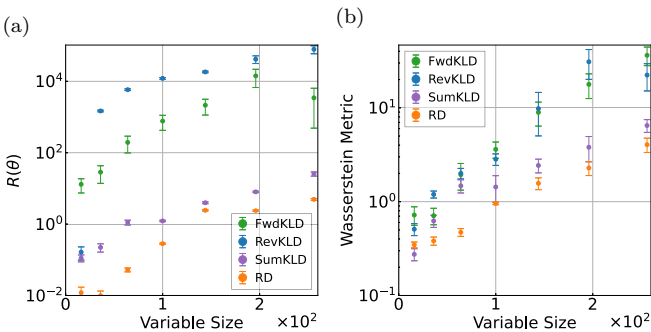


FIG. 2. (a) $R(\theta)$ and (b) Wasserstein metric of each method as a function of the dimension of the 2D Ising models. The error bars represent the standard error.

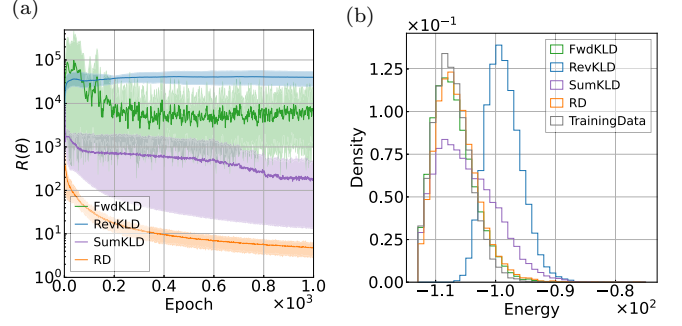


FIG. 3. (a) $R(\theta)$ as functions of epochs during the training process, and (b) empirical energy distributions of the generated samples by each method and of those in the training dataset on the SK model with $N_x = 144$.

Fig. 2(b), for each learning method. Across different numbers of variables, RD learning outperforms the other methods. Notably, when employing RD learning, the rate of increase in the number of variables for $R(\theta)$ and the Wasserstein metric is lower, underscoring its efficiency compared to the other methods. These results indicate that RD learning achieves both high energy regression accuracy and effective mode covering of the distribution.

C. Sherrington–Kirkpatrick model

Next, to demonstrate the advantages of RD learning with a more complex model than the 2D Ising model, we evaluate the performance on the SK model, defined by the energy function $\hat{E}(\mathbf{x}) = -\sum_{i < j} J_{ij} x_i x_j$, where $\mathbf{x} \in \{-1, +1\}^{N_x}$ represents the state configuration and J_{ij} represents interactions drawn from an independent and identically distributed normal distribution $\mathcal{N}(0, 1/N_x)$. Unlike the Ising model, this model is defined on a complete graph, where every pair of distinct nodes is connected. This randomness introduces a large number of peaks in the state space $\mathcal{X}$ for large β . We set $N_x = 144$ and $\beta = 2.0$, where the SK model exhibits multiclustered states. These experiments used five sets of $\{J_{ij}\}$ values.

Figure 3(a) demonstrates that RD learning outperforms the other methods at 1000 epochs in terms of $R(\theta)$. Moreover, RD learning exhibits smaller error bars for $R(\theta)$, demonstrating robust performance across a variety of problems. In contrast, summation divergence learning shows larger errors in $R(\theta)$, which can lead to substantial performance degradation depending on the problem. As shown in Fig. 3(b), the empirical energy distribution of the samples generated by the RBM trained using RD learning more accurately replicates the target energy distribution in the training dataset. See Table I for the value of the Wasserstein metric. Furthermore, Fig. 4 shows that RD learning consistently outperforms other learning methods in terms of $R(\theta)$ across different numbers of variables. The rate of increase in the $R(\theta)$ metric for RD learning is lower than other methods. Furthermore, the increase in the Wasserstein metric for RD learning is comparable to that of summation and reverse KLD learning. These results indicate that RD learning achieves both high energy regression and effective mode covering of the distribution.

TABLE I. Wasserstein metric between the empirical target energy distributions of the samples generated by RBMs, trained using each learning method, and the distribution of the training dataset. The model sizes are indicated in parentheses.

	FwdKLD	RevKLD	SumKLD	RD
ISING MODEL (144)	8.9 ± 2.6	9.8 ± 4.9	2.4 ± 0.4	1.6 ± 0.2
SK MODEL (144)	1.9 ± 0.5	1.5 ± 1.5	1.1 ± 0.7	0.5 ± 0.1
MIS (250)	2.0 ± 0.2	57.0 ± 18.7	5.7 ± 0.3	4.8 ± 0.7
GSET G1 (800)	34.6 ± 1.8	85.9 ± 7.9	22.7 ± 9.0	619.1 ± 36.1
GSET G6 (800)	13.3 ± 2.1	182.9 ± 9.8	5.7 ± 0.7	124.6 ± 39.6
GSET G14 (800)	20.5 ± 1.5	67.0 ± 3.2	31.7 ± 8.5	66.0 ± 33.4
GSET G18 (800)	17.8 ± 2.2	92.5 ± 5.5	5.9 ± 0.4	9.1 ± 0.7

D. Maximum independent set

Next, we consider the MIS problem, which involves finding the largest possible independent set in a given graph $G(V, E)$, and it is noted by Ciarella *et al.* [16] that approximating the distribution is difficult. The energy function for this problem consists of two terms: the size of the independent set and a penalty term, defined as $\hat{E}(x) = -\sum_{i \in V} x_i + \alpha \sum_{(i,j) \in E} x_i x_j$, where $x_i = 0, 1$ indicates whether node i is part of the independent set, and α represents the penalty strength. The MIS problem is characterized by an exponential number of local optima. Consequently, this study uses the MIS problem to evaluate whether RD learning can capture the exponential number of peaks in this distribution. The experiments utilize regular random graphs with 250 nodes and a degree of 20, analyzing the performance of the RBM at $\alpha = 2$ and $\beta = 2.0$, corresponding to a distribution with an exponential number of peaks. The experiments were conducted on five different graphs.

Figure 5(a) demonstrates that RD learning outperforms both the forward and reverse KLD learning in terms of $R(\theta)$. Additionally, RD learning shows a consistent decrease in $R(\theta)$. In contrast, the $R(\theta)$ curves for forward and reverse KLD learning initially decrease but then increase, suggesting that these methods lead to overfitting of the target distribution. This result implies the presence of implicit regularization in RD learning. Although the summation KLD learning has these features, the performance of RD learning is more robust to changes in graph structures compared to the summation KLD learning, as evidenced by the smaller shaded area representing the standard deviation. As shown in Fig. 5(b), the

sampling performance of RD learning is comparable to that of the forward KLD learning, with the Wasserstein metric for RD learning at 4.8 being less than 2.5 times that of the 2.0 achieved by the forward KLD learning. Additionally, Figs. 6(a)–6(d) present two-dimensional maps of generated samples using PCA. These figures illustrate that the forward and reverse KLD learning cause the RBMs to generate samples only within a limited portion of the region occupied by the training dataset. In contrast, RD learning and the summation KLD learning enable the RBM to generate

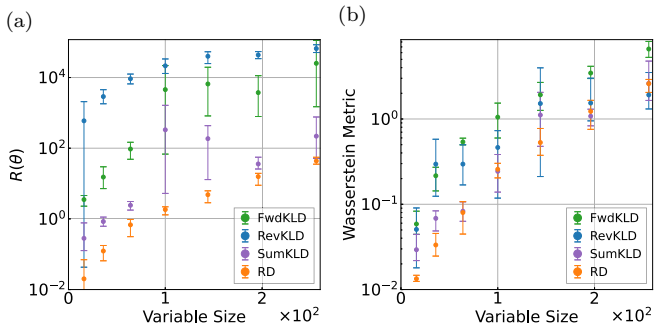


FIG. 4. (a) $R(\theta)$ and (b) Wasserstein metric of each method as a function of the dimension of the SK models. The error bar represents the minimum and maximum values of different graphs.

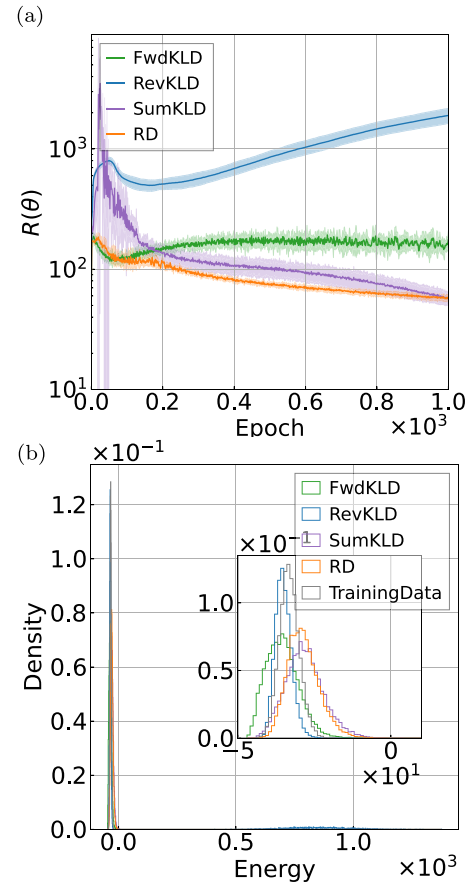


FIG. 5. (a) $R(\theta)$ as a function of epochs during the training process of RBMs, and (b) empirical energy distributions of samples generated by each model compared to those in the training dataset on the MIS of a regular random graph with 250 nodes and a degree of 20.

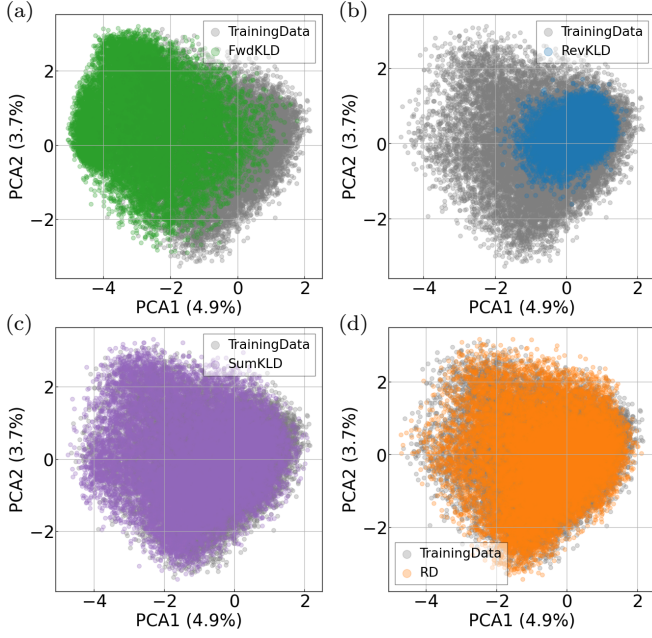


FIG. 6. (a)–(d) Two-dimensional PCA mapping of generated samples by each method for the MIS on a regular random graph with 250 nodes and a degree of 20.

samples uniformly across the entire region. Furthermore, Fig. 7 demonstrates that the Hamming distance of samples generated by the model trained with RD learning closely matches that of the training dataset samples, outperforming the other learning methods. These results indicate that RD learning achieves both high energy regression and effective mode covering of the distribution.

E. Maximum cut problems

An MCP is a discrete optimization problem that aims to find the cut that maximizes the total weight of edges between two partitions of a given graph. The energy function is defined as $\hat{E}(\mathbf{x}) = -\sum_{i < j} w_{ij}(x_i - x_j)^2$, where w_{ij} is the weight of edge (i, j) , and $x_i = 0, 1$ indicates the side of the cut on which node i is located. We evaluate our method on various heterogeneous graph topologies using MCPs: G1, G6, G14, and G18 from the Gset benchmark [38] as practical examples. G1 is a random graph with 800 nodes and a density of 6%,

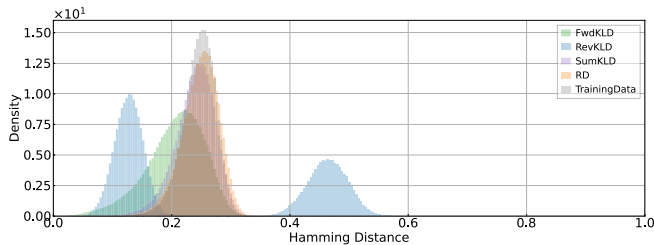


FIG. 7. Empirical distribution of the hamming distance of the generated samples by each method and those in the training dataset for the MIS on a regular random graph with 250 nodes and a degree of 20.

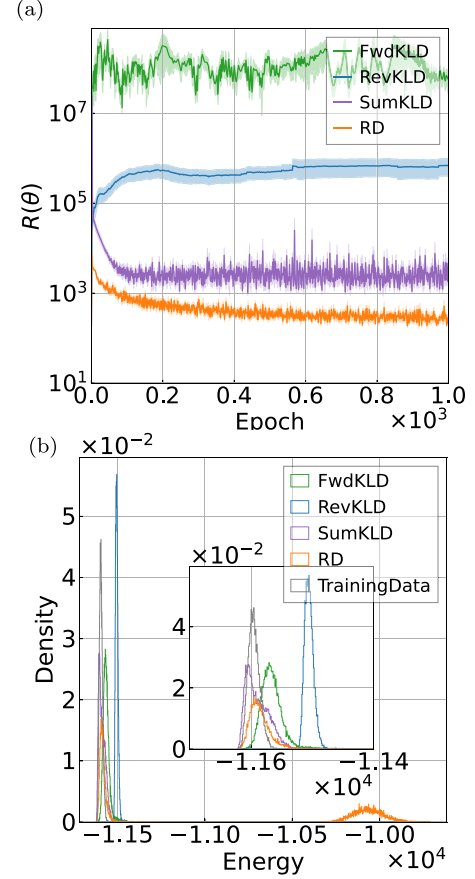


FIG. 8. (a) $R(\theta)$ as functions of epochs during the training process of RBMs and (b) energy distributions of generated samples by each model and those in the training dataset on Gset G1 of the MCP.

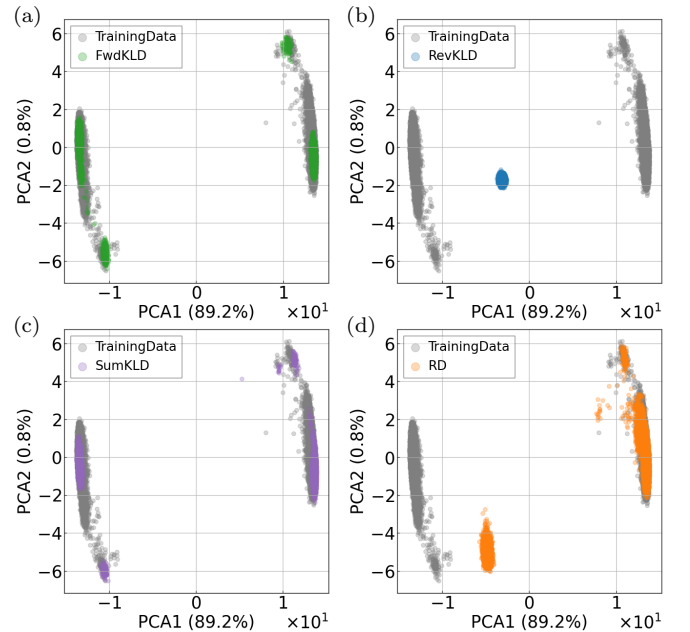


FIG. 9. (a)–(d) Two-dimensional PCA mapping of generated samples by each method for the G1 of MCP.

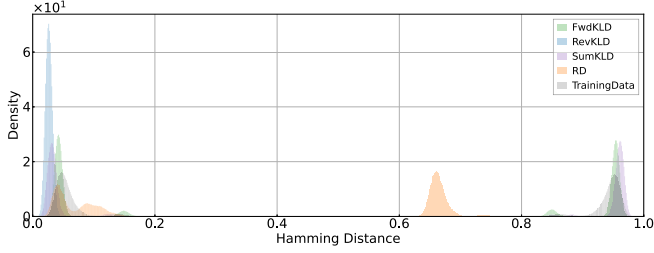


FIG. 10. Empirical distribution of the Hamming distance of the generated samples by each method and those in the training dataset for the G1 of MCP.

where all weights are set to $+1$. G6 has the same structure as G1 but with edge weights of either $+1$ or -1 . G14 is a union graph composed of two random planar graphs with 800 nodes and a density of 99%, where all weights are set to $+1$. G18 has the same structure as G14 but with weights of either $+1$ or -1 . The results for G1 are presented in the main text, while additional results can be found in Appendix C 5. In this study, we analyze the performance of the RBM at $\beta = 1.0$, which corresponds to the conditions under which an optimal solution is typically found using the exchange Monte Carlo method within a realistic time frame.

Figure 8(a) shows that RD learning outperformed the other learning methods at 1000 epochs in terms of $R(\theta)$. Moreover, RD learning exhibits high learning stability for $R(\theta)$. In contrast, summation KLD learning shows fluctuating learning dynamics in $R(\theta)$, which may be due to confusion between the dominance of forward KLD or reverse KLD, caused by the simplistic additive approach.

While Table I shows RD learning is not necessarily better than other learning methods, Figs. 9(d) and 10 indicate that RD learning effectively captures two peaks in the PCA space, although it does not recover one of the peaks as accurately as forward KLD and summation KLD learning. Note that forward KLD and summation KLD learning suffer from lower energy regression performance compared to RD learning concerning $R(\theta)$. As a result, while RD learning may not consistently appear superior in every measurement, its overall performance in terms of mode diversity, energy regression, and learning stability makes it a more reliable and effective approach. Therefore, we consider RD learning to be a

promising tool for practical applications, providing a relevant balance of relatively high stability and performance.

VI. DISCUSSION AND CONCLUSION

This study addresses practical challenges in approximating high-dimensional probability distributions by proposing RD learning, which reduces *underfitting* observed in forward KLD learning and *mode collapse* common to reverse KLD learning. While RD learning accurately approximates true energy functions and effectively trains RBMs that closely model target probability distributions, there is an unaddressed issue—that is, realizing the ease of reaching equilibrium states within the learned models. In particular, RBMs are well known to pose difficulties in reaching equilibrium using conventional BGS [39–41]. This issue requires improvements in sampling methods as well as learning methods. Therefore, a comprehensive evaluation of equilibrium-state accessibility within RD-trained RBMs is left for future work.

Nevertheless, we conducted preliminary experiments where samples were generated by BGS with random initialization, as summarized in Appendix C 4. The results indicate that RD-trained RBMs do not become trapped near their initial states. This supports the view that RD learning is a promising candidate for training RBMs, particularly when novel sampling techniques are developed in the future.

RD learning effectively combines the strengths of both forward and reverse KLD learning while maintaining low computational cost. Moreover, this approach provides a theoretical guarantee of improved acceptance probabilities in Metropolis-Hastings algorithms, leading to enhanced MCMC simulations. Numerical experiments demonstrate that RD learning overcomes the limitations of modeling complex distributions and scales efficiently with increasing dimensionality. Furthermore, RD learning exhibits more stable regression performance compared to summation KLD learning.

In summary, RD learning outperforms traditional forward and reverse KLD learning, as well as their naive combination. The proposed framework is both practical and theoretically grounded, providing a basis for the enhancement of generative models through the use of tractable target energy functions. This advancement opens new avenues for large-scale simulations previously considered infeasible and offers deeper insights into high-dimensional energy landscapes.

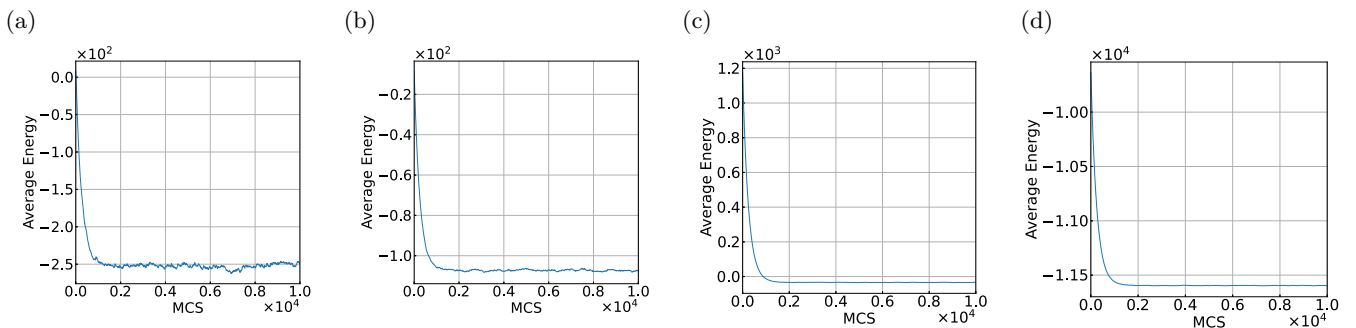


FIG. 11. Average energies as a function of MCS during the burn-in period for (a) the Ising model on a 12×12 square lattice, (b) the SK model with $N_x = 144$, (c) the MIS of a regular random graph with 250 nodes and a degree of 20, and (d) Gset G1, respectively.

TABLE II. Exchange MCMC simulation settings.

TARGET MODEL	N_β	β_1	β_{N_β}
ISING MODEL	4	0.25	0.5
SK MODEL	8	0.5	2.0
MIS	40	0.1	2.0
Gset G1	16	0.25	1.0
GSET G6	16	0.25	2.0
GSET G14	32	0.25	4.0
GSET G18	32	0.25	2.5

ACKNOWLEDGMENTS

This work was supported by JST Grant No. JPMJPF2221 and JPSJ Grant-in-Aid for Scientific Research No. 23H01095. Additionally, Y. Ichikawa was supported by the WINGS-FMSP program at the University of Tokyo.

APPENDIX A: PROOF OF RATIO DIVERGENCE PROPERTIES

1. Symmetric divergence

In this Appendix, we prove that Eq. (8) is a symmetric divergence. To this end, since it is trivial that Eq. (8) is symmetric and non-negative, we need to prove $\mathcal{L}(\hat{P}, P; \theta) \Leftrightarrow \hat{P} = P$. Equation (8) sums up the square of $\log \frac{\hat{P}(\mathbf{x}')P(\mathbf{x}; \theta)}{P(\mathbf{x}'; \theta)\hat{P}(\mathbf{x})}$. Therefore, if $\mathcal{L}(\hat{P}, P; \theta) = 0$, then the following function satisfies for all $\mathbf{x}$ and $\mathbf{x}'$:

$$\log \frac{\hat{P}(\mathbf{x}')P(\mathbf{x}; \theta)}{P(\mathbf{x}'; \theta)\hat{P}(\mathbf{x})} = 0 \Leftrightarrow \frac{\hat{P}(\mathbf{x}')}{\hat{P}(\mathbf{x})} = \frac{P(\mathbf{x}'; \theta)}{P(\mathbf{x}; \theta)}. \quad (\text{A1})$$

This equation straightforwardly leads to $\hat{P}(\mathbf{x}) = P(\mathbf{x}; \theta)$ for all $\mathbf{x}$ since $\hat{P}$ and P are probabilities. On the other hand, it is trivial that $\hat{P} = P \Rightarrow \mathcal{L}(\hat{P}, P; \theta) = 0$.

2. Guarantee of increasing acceptance probability

We denote $A(\mathbf{x}', \mathbf{x})$ as the acceptance probability in the Metropolis-Hastings algorithm, that is,

$$A(\mathbf{x}', \mathbf{x}) = \min \left(1, \frac{\hat{P}(\mathbf{x}')P(\mathbf{x}; \theta)}{P(\mathbf{x}'; \theta)\hat{P}(\mathbf{x})} \right). \quad (\text{A2})$$

For any $\mathbf{x}'$ and $\mathbf{x}$, $\hat{P}$ and P satisfy the following inequality:

$$\begin{aligned} \left(\log \frac{\hat{P}(\mathbf{x}')P(\mathbf{x}; \theta)}{P(\mathbf{x}'; \theta)\hat{P}(\mathbf{x})} \right)^2 &\geq \left(\min \left(0, \log \frac{\hat{P}(\mathbf{x}')P(\mathbf{x}; \theta)}{P(\mathbf{x}'; \theta)\hat{P}(\mathbf{x})} \right) \right)^2 \\ &= (\log A(\mathbf{x}', \mathbf{x}))^2. \end{aligned} \quad (\text{A3})$$

Taking the expectation with respect to $\hat{P}$ and P over $\mathbf{x}'$ and $\mathbf{x}$ respectively, and using Jensen's inequality, we obtain the following inequality:

$$\begin{aligned} \mathcal{L}(\hat{P}, P; \theta) &\geq \mathbb{E}_{\hat{P}(\mathbf{x}'), P(\mathbf{x}; \theta)} [(\log A(\mathbf{x}', \mathbf{x}))^2] \\ &\geq (\mathbb{E}_{\hat{P}(\mathbf{x}'), P(\mathbf{x}; \theta)} [\log A(\mathbf{x}', \mathbf{x})])^2. \end{aligned} \quad (\text{A4})$$

Here, considering $\mathcal{L}(\hat{P}, P; \theta) \geq 0$ and $\mathbb{E}_{\hat{P}(\mathbf{x}'), P(\mathbf{x}; \theta)} [\log A(\mathbf{x}', \mathbf{x})] \leq 0$, and using Jensen's inequality, we obtain the following inequality:

$$\begin{aligned} -\sqrt{\mathcal{L}(\hat{P}, P; \theta)} &\leq \mathbb{E}_{\hat{P}(\mathbf{x}'), P(\mathbf{x}; \theta)} [\log A(\mathbf{x}', \mathbf{x})] \\ &\leq \log \mathbb{E}_{\hat{P}(\mathbf{x}'), P(\mathbf{x}; \theta)} [A(\mathbf{x}', \mathbf{x})]. \end{aligned} \quad (\text{A5})$$

TABLE III. $R(\theta)$ at 1000 epochs. The model sizes are indicated in parentheses. The best value in each row is marked in bold.

	DATASET SIZE	FWDKLD	REVKLD	SUMKLD	RD
ISING MODEL (144)	4096	$(32.6 \pm 24.1) \times 10^3$	$(33.3 \pm 11.1) \times 10^3$	$(24.6 \pm 4.4) \times 10^1$	9.7 ± 0.3
	8192	$(42.1 \pm 15.0) \times 10^1$	$(31.3 \pm 2.9) \times 10^3$	6.6 ± 0.3	4.2 ± 0.1
	16384	$(21.4 \pm 10.1) \times 10^2$	$(18.3 \pm 1.0) \times 10^3$	4.0 ± 0.2	2.4 ± 0.1
SK MODEL (144)	4096	$(24.9 \pm 24.3) \times 10^2$	$(36.8 \pm 14.5) \times 10^3$	$(57.1 \pm 70.6) \times 10^1$	19.8 ± 7.7
	8192	$(49.1 \pm 61.0) \times 10^2$	$(42.8 \pm 14.8) \times 10^3$	$(30.1 \pm 39.0) \times 10^1$	10.9 ± 3.6
	16384	$(65.7 \pm 81.2) \times 10^2$	$(39.9 \pm 10.8) \times 10^3$	$(18.6 \pm 22.2) \times 10^1$	4.8 ± 1.4
MIS (250)	4096	$(26.4 \pm 1.7) \times 10^1$	$(52.5 \pm 4.7) \times 10^1$	$(13.4 \pm 1.7) \times 10^1$	$(13.2 \pm 0.6) \times 10^1$
	8192	$(24.6 \pm 1.4) \times 10^1$	$(79.4 \pm 7.9) \times 10^1$	$(11.6 \pm 1.8) \times 10^1$	$(11.3 \pm 0.6) \times 10^1$
	16384	$(17.6 \pm 3.2) \times 10^1$	$(19.0 \pm 2.7) \times 10^2$	58.0 ± 7.0	56.7 ± 3.5
GSET G1 (800)	4096	$(14.9 \pm 7.7) \times 10^7$	$(80.7 \pm 26.8) \times 10^4$	$(22.8 \pm 7.6) \times 10^2$	$(45.6 \pm 4.9) \times 10^1$
	8192	$(10.2 \pm 4.1) \times 10^7$	$(44.7 \pm 20.0) \times 10^4$	$(29.2 \pm 16.9) \times 10^2$	$(36.1 \pm 4.0) \times 10^1$
	16384	$(55.5 \pm 30.0) \times 10^6$	$(70.0 \pm 29.1) \times 10^4$	$(37.6 \pm 14.6) \times 10^2$	$(26.5 \pm 2.9) \times 10^1$
GSET G6 (800)	4096	$(27.4 \pm 11.2) \times 10^7$	$(41.0 \pm 14.5) \times 10^5$	$(12.9 \pm 5.0) \times 10^3$	$(19.4 \pm 5.1) \times 10^1$
	8192	$(44.0 \pm 34.7) \times 10^7$	$(12.9 \pm 5.1) \times 10^5$	$(56.8 \pm 15.0) \times 10^2$	90.9 ± 12.1
	16384	$(15.9 \pm 6.4) \times 10^7$	$(45.9 \pm 10.1) \times 10^5$	$(34.9 \pm 25.4) \times 10^3$	87.0 ± 24.8
GSET G14 (800)	4096	$(19.6 \pm 5.8) \times 10^5$	$(44.4 \pm 15.2) \times 10^4$	$(84.4 \pm 65.9) \times 10^4$	$(11.5 \pm 2.1) \times 10^3$
	8192	$(11.8 \pm 2.4) \times 10^5$	$(47.6 \pm 20.6) \times 10^4$	$(31.2 \pm 30.1) \times 10^4$	$(11.6 \pm 1.1) \times 10^2$
	16384	$(36.4 \pm 12.1) \times 10^5$	$(54.8 \pm 21.3) \times 10^4$	$(27.3 \pm 14.2) \times 10^3$	$(95.0 \pm 11.4) \times 10^1$
GSET G18 (800)	4096	$(53.2 \pm 23.6) \times 10^5$	$(15.7 \pm 4.8) \times 10^4$	$(30.3 \pm 4.0) \times 10^4$	$(10.8 \pm 0.1) \times 10^4$
	8192	$(22.8 \pm 5.4) \times 10^5$	$(14.9 \pm 4.6) \times 10^4$	$(19.8 \pm 4.4) \times 10^4$	$(14.2 \pm 0.9) \times 10^3$
	16384	$(16.5 \pm 5.5) \times 10^5$	$(22.2 \pm 14.5) \times 10^4$	$(22.0 \pm 14.6) \times 10^4$	$(52.3 \pm 11.7) \times 10^1$

TABLE IV. Wasserstein metric between the empirical target energy distributions of the samples generated by RBMs, trained using each learning method, and the distribution of the training dataset. The model sizes are indicated in parentheses. The best value in each row is marked in bold.

	DATASET SIZE	FwdKLD	RevKLD	SUMKLD	RD
ISING MODEL (144)	4096	10.1 ± 3.9	12.2 ± 0.3	3.2 ± 0.5	1.4 ± 0.1
	8192	5.5 ± 1.6	8.1 ± 0.6	1.5 ± 0.4	1.2 ± 0.1
	16 384	8.9 ± 2.6	9.8 ± 4.8	2.4 ± 0.4	1.6 ± 0.2
SK MODEL (144)	4096	3.1 ± 0.9	2.8 ± 0.9	1.4 ± 0.4	2.1 ± 0.6
	8192	2.5 ± 0.8	2.1 ± 1.2	1.2 ± 0.2	1.0 ± 0.3
	16 384	1.9 ± 0.5	1.5 ± 1.5	1.1 ± 0.7	0.5 ± 0.1
MIS (250)	4096	13.8 ± 0.7	592.3 ± 37.9	28.4 ± 2.6	14.8 ± 0.2
	8192	10.0 ± 0.6	399.7 ± 40.1	15.8 ± 0.4	11.6 ± 0.5
	16 384	2.0 ± 0.2	57.0 ± 18.7	5.7 ± 0.3	4.8 ± 0.7
GSET G1 (800)	4096	60.3 ± 10.3	100.1 ± 7.1	19.2 ± 3.1	328.6 ± 7.9
	8192	42.4 ± 4.3	104.8 ± 8.5	16.2 ± 1.4	322.7 ± 12.2
	16 384	34.6 ± 1.8	85.9 ± 7.9	22.7 ± 9.0	619.1 ± 36.1
GSET G6 (800)	4096	34.7 ± 12.6	170.1 ± 8.7	16.7 ± 5.3	154.4 ± 26.9
	8192	15.4 ± 3.2	163.5 ± 8.3	11.0 ± 2.0	186.3 ± 19.2
	16 384	13.3 ± 2.1	182.9 ± 9.8	5.7 ± 0.7	124.6 ± 39.6
GSET G14 (800)	4096	34.3 ± 4.6	67.1 ± 4.9	26.5 ± 7.2	238.0 ± 61.0
	8192	26.0 ± 2.3	66.0 ± 2.9	21.3 ± 2.2	53.5 ± 40.0
	16 384	20.5 ± 1.5	67.0 ± 3.2	31.7 ± 8.5	66.0 ± 33.4
GSET G18 (800)	4096	24.6 ± 2.3	89.9 ± 7.5	9.3 ± 1.5	9.6 ± 0.5
	8192	18.3 ± 1.7	93.5 ± 5.4	8.7 ± 1.8	8.1 ± 0.4
	16 384	17.8 ± 2.2	92.5 ± 5.5	5.5 ± 0.4	9.1 ± 0.7

TABLE V. Time required to train for 1000 epochs. The model sizes are indicated in parentheses.

	FwdKLD[s]	RevKLD[s]	SUMKLD[s]	RD[s]
ISING MODEL (144)	201.5 ± 0.4	378.5 ± 0.6	438.6 ± 0.6	527.2 ± 0.5
SK MODEL (144)	214.3 ± 12.6	391.4 ± 7.6	458.2 ± 15.2	534.1 ± 10.4
MIS (250)	220.5 ± 13.0	398.3 ± 15.5	461.7 ± 16.4	534.6 ± 15.0
GSET G1 (800)	216.1 ± 1.0	397.2 ± 4.2	476.9 ± 8.5	536.2 ± 2.7
GSET G6 (800)	212.2 ± 1.0	392.8 ± 0.8	459.2 ± 1.3	536.6 ± 1.5
GSET G14 (800)	217.9 ± 2.9	394.9 ± 1.4	465.0 ± 3.4	534.2 ± 1.3
GSET G18 (800)	224.0 ± 4.3	412.6 ± 5.5	472.3 ± 6.3	553.2 ± 4.7

TABLE VI. Wasserstein metric between the empirical target energy distributions of the samples generated by RBMs trained using each learning method, and the distribution of the training dataset. Randomly initialized states were used as the starting point for sampling. The model sizes are indicated in parentheses. The best value in each row is marked in bold.

	FwdKLD	RevKLD	SUMKLD	RD
ISING MODEL (144)	7.0 ± 1.6	16.0 ± 9.2	17.9 ± 1.4	3.8 ± 0.5
SK MODEL (144)	4.9 ± 2.5	1.6 ± 1.6	8.2 ± 6.9	19.3 ± 18.4
MIS (250)	2.6 ± 0.4	474.7 ± 77.5	7.2 ± 0.4	6.5 ± 0.4
GSET G1 (800)	1683.4 ± 770.6	86.0 ± 7.9	1081.7 ± 177.9	469.1 ± 28.9
GSET G6 (800)	458.3 ± 249.3	182.9 ± 9.8	618.9 ± 243.0	173.8 ± 13.1
GSET G14 (800)	137.8 ± 62.2	67.0 ± 3.2	393.1 ± 56.1	341.7 ± 52.9
GSET G18 (800)	225.7 ± 72.5	92.5 ± 5.5	238.2 ± 49.6	239.2 ± 20.3

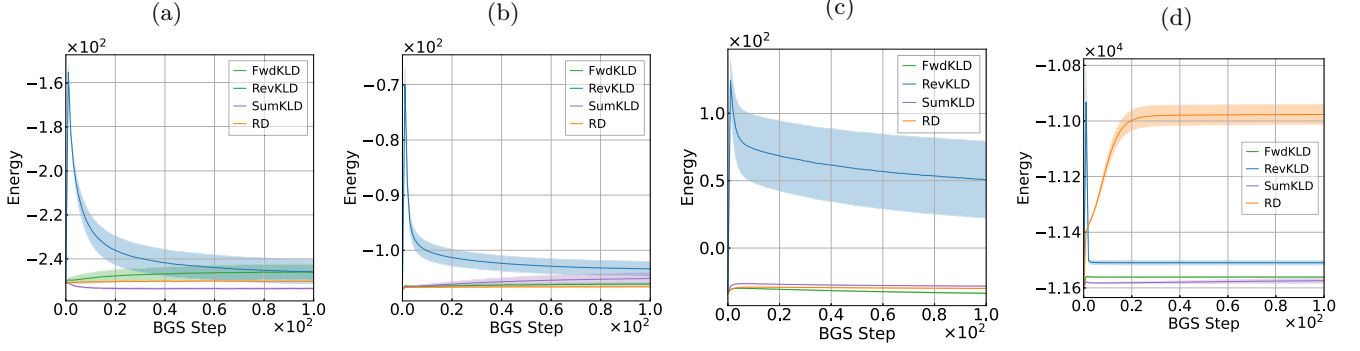


FIG. 12. Average energies as a function of the BGS steps for (a) the Ising model on a 12×12 square lattice, (b) the SK model with $N_x = 144$, (c) the MIS of a regular random graph with 250 nodes and a degree of 20, and (d) Gset G1.

Then, we obtain the following inequality:

$$\exp(-\sqrt{\mathcal{L}(\hat{P}, P; \theta)}) \leq \mathbb{E}_{\hat{P}(x'), P(x; \theta)}[A(x', x)]. \quad (\text{A6})$$

This is identical to inequality (11).

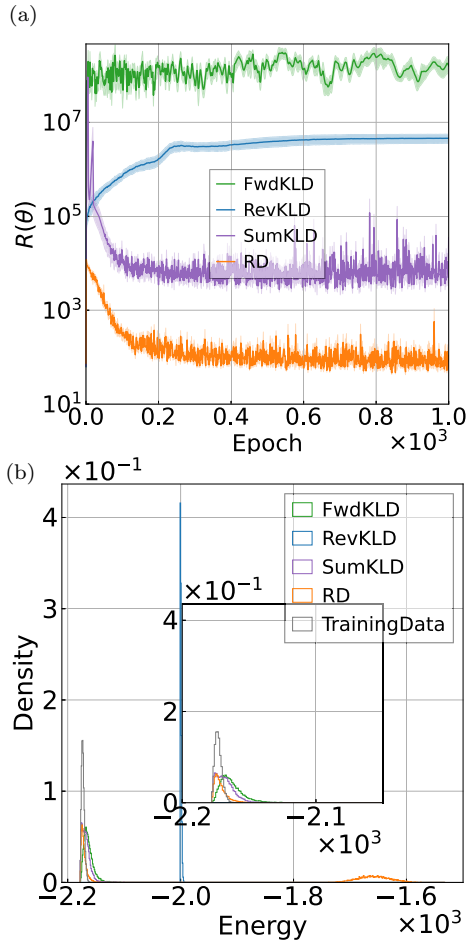


FIG. 13. (a) $R(\theta)$ as functions of epochs during the training process of RBMs and (b) energy distributions of generated samples by each model and those in the training dataset on Gset G6 of the MCP.

APPENDIX B: ADDITIONAL EXPERIMENT DETAILS

1. How to prepare a training dataset using exchange MCMC simulations

In all experiments, we ran exchange MCMC simulations for 10^6 Monte Carlo steps (MCS), performing replica exchanges every 1 MCS and recording samples every 10 MCS, resulting in 10^5 recorded samples. After discarding the first 10^4 recorded samples as a burn-in period, we adopted the next 16 384 samples as the training dataset and the last 1024 samples as the validation dataset. The validity of the length of the burn-in period was confirmed by convergence of an average energy, as shown in Fig. 11. For each target model, we set the number of replicas, the minimum temperatures, and the maximum temperature as shown in Table II. We set the intermediate temperatures using the following formula to ensure the temperature ratio is constant:

$$\beta_i = \beta_1 \exp\left(\frac{i-1}{N-1} \log \frac{\beta_{N_\beta}}{\beta_1}\right), \quad (\text{B1})$$

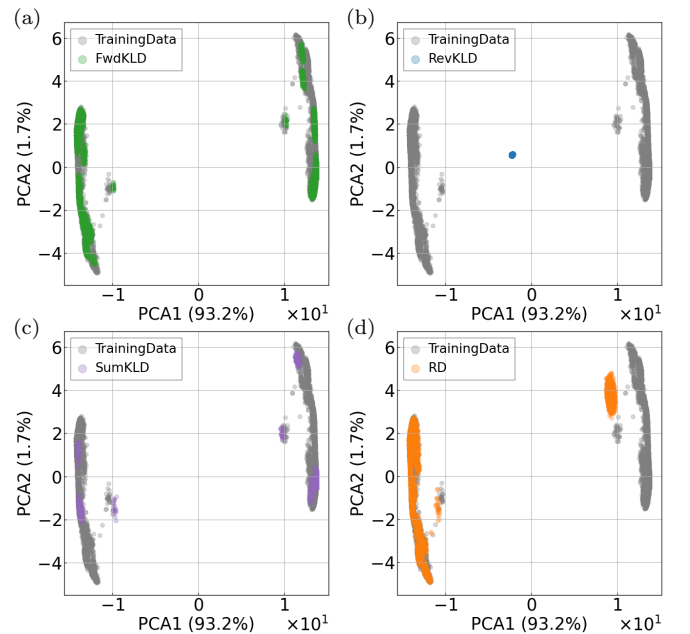


FIG. 14. (a)–(d) Two-dimensional PCA mapping of generated samples by each method for the G6 of MCP.

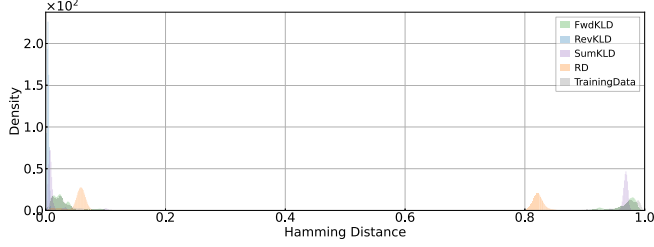


FIG. 15. Empirical distribution of the Hamming distance of the generated samples by each method and those in the training dataset for the G6 of MCP.

where N_β is the number of replicas, $\{\beta_i\}_{i=1}^{N_\beta}$ is the temperature.

2. Wasserstein metric

Wasserstein metric of the $L1$ norm between probabilities $P(x)$ and $Q(x)$ on $x \in \mathbb{R}$ is written as follows:

$$D_W(P, Q) = \int |F_P(x) - F_Q(x)| dx, \quad (\text{B2})$$

where F_P and F_Q is the cumulative distribution function of P and Q , respectively. In all experiments, we used SciPy version 1.7.0 to calculate this metric.

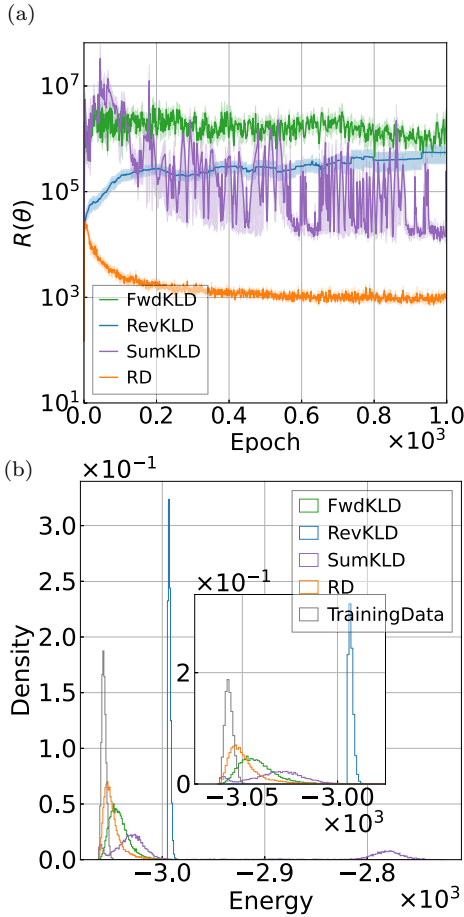


FIG. 16. (a) $R(\theta)$ as functions of epochs during the training process of RBMs and (b) energy distributions of generated samples by each model and those in the training dataset on Gset G14 of the MCP.

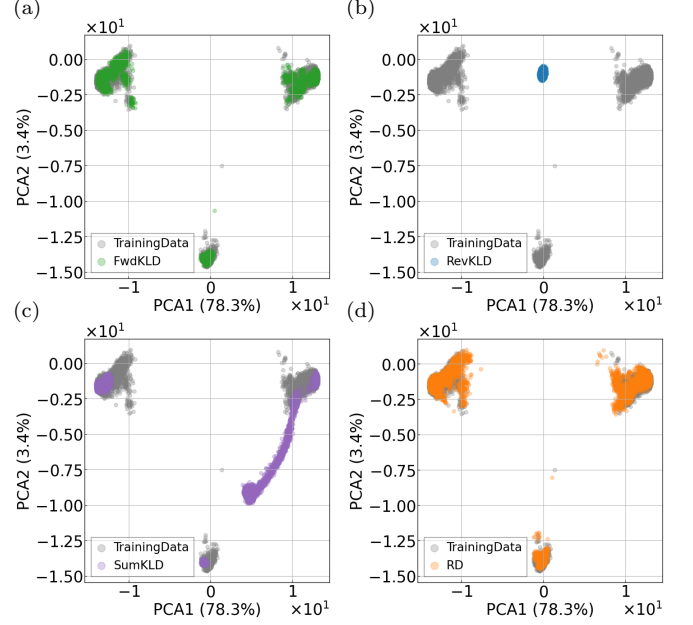


FIG. 17. (a)–(d) Two-dimensional PCA mapping of generated samples by each method for the G14 of MCP.

APPENDIX C: ADDITIONAL RESULTS

1. Dependence on training dataset size

In the main text, we present results for a fixed dataset size of 16384. To examine how performance varies with the number of training samples, we measure $R(\theta)$ to evaluate how well the learned RBM distribution aligns with the target energy function. Additionally, we use the Wasserstein metric to measure how closely the generated samples are consistent with the trained distribution across different dataset sizes. The corresponding results for different dataset sizes are summarized in Tables III and IV, respectively. The findings confirm that the trends discussed in the main text remain consistent even when the dataset size is varied.

2. Training time

Training time is a common metric for evaluating the practicality of a learning method. In this Appendix, a comparison is made of the time required by each learning method to perform 10^3 training epochs. Table V presents the measured training time across different systems. While the RD method requires

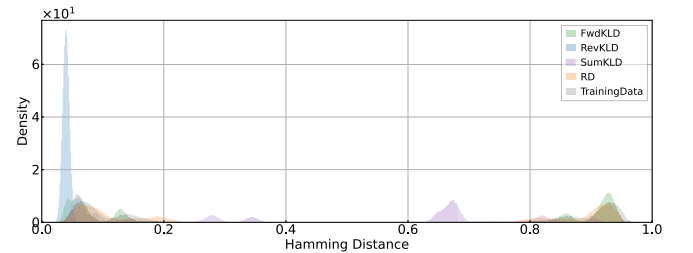


FIG. 18. Empirical distribution of the Hamming distance of the generated samples by each method and those in the training dataset for the G14 of MCP.

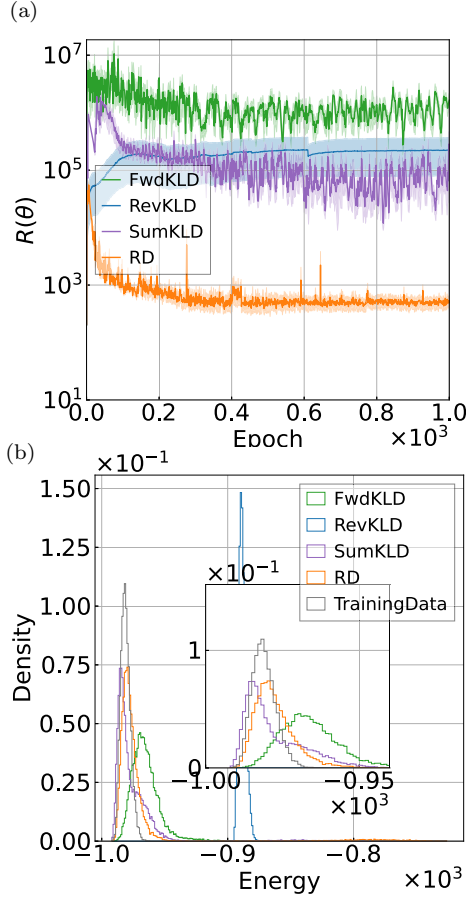


FIG. 19. (a) $R(\theta)$ as functions of epochs during the training process of RBMs and (b) energy distributions of generated samples by each model and those in the training dataset on Gset G18 of the MCP.

more training time than other methods, it has been observed to converge more quickly in practice, making this added computational overhead less problematic in many scenarios.

3. Convergence of the block Gibbs sampling

In this Appendix, we present additional results on generating samples from trained RBMs using block Gibbs sampling (BGS). Figures 12 show the evolution of the average energy as a function of the BGS steps for each learning method. Notably, among the learning methods evaluated, the energy in RD learning reaches a stable value well before 100 steps. This rapid convergence suggests that RD learning enables RBMs to generate samples efficiently, demonstrating their ability for downstream applications where fast sampling is crucial.

4. Block Gibbs sampling with random initialization

In the main text, we performed BGS starting from the training dataset, since this approach is practical for using surrogate models in downstream tasks. However, this experiment does not rule out the possibility that RBMs may remain stuck near their initial states. To confirm that the trained RBMs can generate samples distinct from their initial states, we examine the behavior of RBMs trained using each method, including our proposed RD method, starting from random initializations.

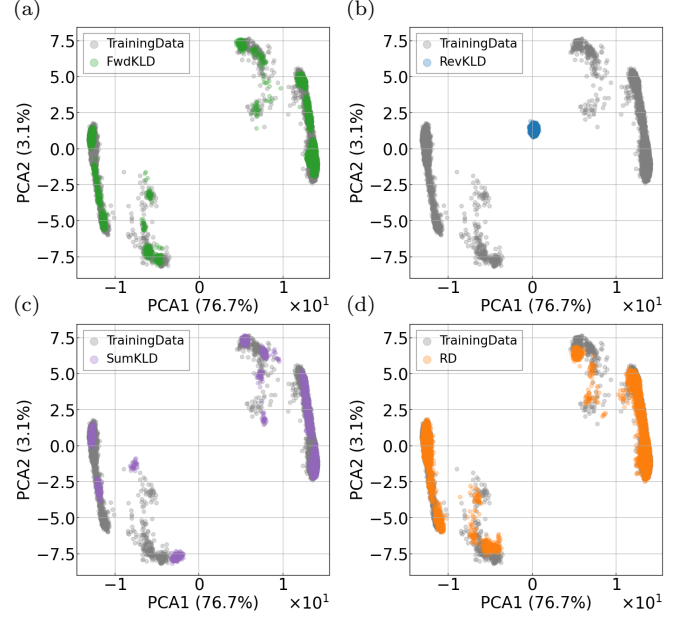


FIG. 20. (a)–(d) Two-dimensional PCA mapping of generated samples by each method for the G18 of MCP.

Table VI presents the Wasserstein metrics computed from the samples generated under these random initializations. While the Wasserstein metrics with random initialization are higher than those obtained when initializing from the training dataset, this result indicates that the RBMs do not remain stuck near their initial states.

5. Experimental results of Gset on distinct graphs

In the main text, we presented results for the G1 instance of the Gset benchmark. Here, we discuss additional findings for other Gset graphs (G6, G14, and G18) to provide a comprehensive overview, as shown in Table I. While RD learning does not achieve the lowest Wasserstein metric, Figs. 13–21 show that RD learning outperforms forward KLD learning in terms of $R(\theta)$. Moreover, by projecting both the training dataset and generated samples into a reduced feature space obtained using PCA and examining their Hamming distance distributions, we find that RD learning effectively covers multiple modes in these graphs.

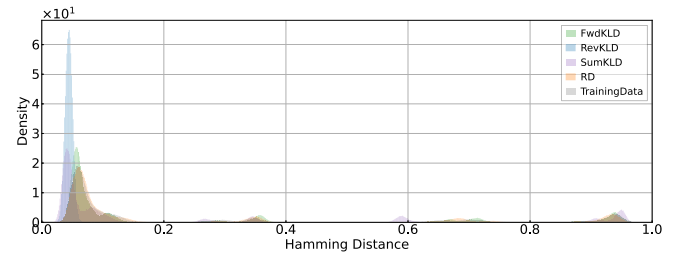


FIG. 21. Empirical distribution of the Hamming distance of the generated samples by each method and those in the training dataset for the G18 of MCP.

- [1] A. Baumgärtner, A. N. Burkitt, D. M. Ceperley, H. De Raedt, A. M. Ferrenberg, D. W. Heermann, H. J. Herrmann, D. P. Landau, D. Levesque, W. von der Linden *et al.*, *The Monte Carlo Method in Condensed Matter Physics* (Springer Science & Business Media, New York, 2012), Vol. 71.
- [2] K. Binder, D. Heermann, L. Roelofs, A. J. Mallinckrodt, and S. McKay, Monte Carlo simulation in statistical physics, *Comput. Phys.* **7**, 156 (1993).
- [3] D. Foreman-Mackey, D. W. Hogg, D. Lang, and J. Goodman, emcee: the MCMC hammer, *Publ. Astron. Soc. Pac.* **125**, 306 (2013).
- [4] A. Gelman, J. B. Carlin, H. S. Stern, and D. B. Rubin, *Bayesian Data Analysis* (Chapman and Hall/CRC Press, Boca Raton, Florida, 1995).
- [5] A. D. Martin, K. M. Quinn, and J. H. Park, MCMCpack: Markov chain Monte Carlo in R, *J. Stat. Softw.* **42**, 1 (2011).
- [6] K. Dabiri, M. Malekmohammadi, A. Sheikholeslami, and H. Tamura, Replica exchange MCMC hardware with automatic temperature selection and parallel trial, *IEEE Trans. Parallel Distrib. Syst.* **31**, 1681 (2020).
- [7] J. Gu and K. Zhang, Thermodynamics of the Ising model encoded in restricted Boltzmann machines, *Entropy* **24**, 1701 (2022).
- [8] W. Chen, A. R. Tan, and A. L. Ferguson, Collective variable discovery and enhanced sampling using autoencoders: Innovations in network architecture and error function design, *J. Chem. Phys.* **149**, 072312 (2018).
- [9] J. M. L. Ribeiro, P. Bravo, Y. Wang, and P. Tiwary, Reweighted autoencoded variational Bayes for enhanced sampling (RAVE), *J. Chem. Phys.* **149**, 072301 (2018).
- [10] L. Huang and L. Wang, Accelerated Monte Carlo simulations with restricted Boltzmann machines, *Phys. Rev. B* **95**, 035105 (2017).
- [11] J. Liu, Y. Qi, Z. Y. Meng, and L. Fu, Self-learning Monte Carlo method, *Phys. Rev. B* **95**, 041101(R) (2017).
- [12] X. Y. Xu, Y. Qi, J. Liu, L. Fu, and Z. Y. Meng, Self-learning quantum Monte Carlo method in interacting fermion systems, *Phys. Rev. B* **96**, 041119(R) (2017).
- [13] V. Stimper, B. Schölkopf, and J. M. Hernández-Lobato, Resampling base distributions of normalizing flows, in *International Conference on Artificial Intelligence and Statistics* (PMLR, 2022), pp. 4915–4936.
- [14] P. Wirsberger, G. Papamakarios, B. Ibarz, S. Racanière, A. J. Ballard, A. Pritzel, and C. Blundell, Normalizing flows for atomic solids, *Mach. Learn.: Sci. Technol.* **3**, 025009 (2022).
- [15] H. Wu, J. Köhler, and F. Noé, Stochastic normalizing flows, in *Advances in Neural Information Processing Systems*, edited by H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin (Curran Associates, Inc., 2020), Vol. 33, pp. 5933–5944.
- [16] S. Ciarella, J. Trinquier, M. Weigt, and F. Zamponi, Machine-learning-assisted Monte Carlo fails at sampling computationally hard problems, *Mach. Learn.: Sci. Technol.* **4**, 010501 (2023).
- [17] G. E. Hinton, Training products of experts by minimizing contrastive divergence, *Neural Comput.* **14**, 1771 (2002).
- [18] D. E. Rumelhart, J. L. McClelland, and PDP Research Group, Information processing in dynamical systems: Foundations of Harmony theory, in *Parallel Distributed Processing: Explorations in the Microstructure of Cognition: Foundations* (MIT Press, Cambridge, Massachusetts, 1986), Vol. 1, pp.194–281.
- [19] N. Le Roux and Y. Bengio, Representational power of restricted Boltzmann machines and deep belief networks, *Neural Comput.* **20**, 1631 (2008).
- [20] R. D. Hjelm, V. D. Calhoun, R. Salakhutdinov, E. A. Allen, T. Adali, and S. M. Plis, Restricted Boltzmann machines for neuroimaging: an application in identifying intrinsic networks, *NeuroImage* **96**, 245 (2014).
- [21] R. G. Melko, G. Carleo, J. Carrasquilla, and J. I. Cirac, Restricted Boltzmann machines in quantum physics, *Nat. Phys.* **15**, 887 (2019).
- [22] A. Khajenezhad, H. Madani, and H. Beigy, Masked autoencoder for distribution estimation on small structured data sets, *IEEE Trans. Neural Netw. Learn. Syst.* **32**, 4997 (2020).
- [23] B. Uria, M.-A. Côté, K. Gregor, I. Murray, and H. Larochelle, Neural autoregressive distribution estimation, *J. Mach. Learn. Res.* **17**, 1 (2016).
- [24] T. Tieleman, Training restricted Boltzmann machines using approximations to the likelihood gradient, in *Proceedings of the 25th International Conference on Machine Learning ICML '08*, (Association for Computing Machinery, New York, NY, USA, 2008), pp. 1064–1071.
- [25] D. A. Puente and I. M. Eremin, Convolutional restricted Boltzmann machine aided Monte Carlo: an application to Ising and Kitaev models, *Phys. Rev. B* **102**, 195148 (2020).
- [26] B. McNaughton, M. V. Milošević, A. Perali, and S. Pilati, Boosting Monte Carlo simulations of spin glasses using autoregressive neural networks, *Phys. Rev. E* **101**, 053312 (2020).
- [27] D. Wu, R. Rossi, and G. Carleo, Unbiased Monte Carlo cluster updates with autoregressive neural networks, *Phys. Rev. Res.* **3**, L042024 (2021).
- [28] D. Wu, L. Wang, and P. Zhang, Solving statistical mechanics using variational autoregressive networks, *Phys. Rev. Lett.* **122**, 080602 (2019).
- [29] C. Fan, M. Shen, Z. Nussinov, Z. Liu, Y. Sun, and Y.-Y. Liu, Finding spin glass ground states through deep reinforcement learning, *arXiv:2109.14411*.
- [30] M. Hibat-Allah, E. M. Inack, R. Wiersema, R. G. Melko, and J. Carrasquilla, Variational neural annealing, *Nat Mach Intell* **3**, 952 (2021).
- [31] E. M. Inack, S. Morawetz, and R. G. Melko, Neural annealing and visualization of autoregressive neural networks in the Newman–Moore model, *Condens. Matter* **7**, 38 (2022).
- [32] A. Hyvärinen, Some extensions of score matching, *Comput. Stat. Data Anal.* **51**, 2499 (2007).
- [33] L. I. Midgley, V. Stimper, G. N. C. Simm, B. Schölkopf, and J. M. Hernández-Lobato, Flow annealed importance sampling bootstrap, *arXiv:2208.01893*.
- [34] H. Fernau, P. Golovach, M.-F. Sagot *et al.*, Algorithmic enumeration: Output-sensitive, input-sensitive, parameterized, approximative (Dagstuhl seminar 18421), in *Dagstuhl Reports* (Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany, 2019), Vol. 8, pp. 63–86.
- [35] T. Hanaka, M. Kiyomi, Y. Kobayashi, Y. Kobayashi, K. Kurita, and Y. Otachi, A framework to design approximation algorithms for finding diverse solutions in combinatorial

- problems, *Proc. AAAI Conf. Artif. Intell.* **37**, 3968 (2023).
- [36] K. Hukushima and K. Nemoto, Exchange Monte Carlo method and application to spin glass simulations, *J. Phys. Soc. Jpn.* **65**, 1604 (1996).
- [37] D. P. Kingma and J. Ba, Adam: A method for stochastic optimization, [arXiv:1412.6980](https://arxiv.org/abs/1412.6980).
- [38] Y. Ye, Gset (2003), <http://web.stanford.edu/~yyye/yyye/Gset/>.
- [39] N. Béréux, A. Decelle, C. Furtlehner, L. Rosset, and B. Seoane, Fast, accurate training and sampling of restricted Boltzmann machines, [arXiv:2405.15376](https://arxiv.org/abs/2405.15376).
- [40] A. Decelle, C. Furtlehner, and B. Seoane, Equilibrium and non-equilibrium regimes in the learning of restricted Boltzmann machines, in *Proceedings of the 35th International Conference on Neural Information Processing System* (Curran Associates Inc., Red Hook, NY, 2021), pp. 5345–5359.
- [41] E. Nijkamp, M. Hill, S.-C. Zhu, and Y. N. Wu, Learning non-convergent non-persistent short-run MCMC toward energy-based model, in *Advances in Neural Information Processing Systems* (Curran Associates, Inc., Red Hook, New York, 2019), Vol. 32.