Mach Learn (2013) 93:53–69
DOI 10.1007/s10994-013-5390-3

The flip-the-state transition operator for restricted Boltzmann machines

Kai Brügge · Asja Fischer · Christian Igel

Received: 14 February 2013 / Accepted: 3 June 2013 / Published online: 3 July 2013

Abstract Most learning and sampling algorithms for restricted Boltzmann machines (RMBs) rely on Markov chain Monte Carlo (MCMC) methods using Gibbs sampling. The most prominent examples are Contrastive Divergence learning (CD) and its variants as well as Parallel Tempering (PT). The performance of these methods strongly depends on the mixing properties of the Gibbs chain. We propose a Metropolis-type MCMC algorithm relying on a transition operator maximizing the probability of state changes. It is shown that the operator induces an irreducible, aperiodic, and hence properly converging Markov chain, also for the typically used periodic update schemes. The transition operator can replace Gibbs sampling in RBM learning algorithms without producing computational overhead. It is shown empirically that this leads to faster mixing and in turn to more accurate learning.

Keywords Restricted Boltzmann machine · Markov chain Monte Carlo · Gibbs sampling · Mixing rate · Contrastive divergence learning · Parallel tempering

1 Introduction

Restricted Boltzmann machines (RBMs, Smolensky 1986; Hinton 2002) are undirected graphical models describing stochastic neural networks. They have raised much attention

Editors: Hendrik Blockeel, Kristian Kersting, Siegfried Nijssen, and Filip Železný.

K. Brügge
Department of Computer Science, University of Helsinki, P.O. Box 68, 00014 Helsinki, Finland
e-mail: Kai.Bruegge@cs.helsinki.fi

K. Brügge
Helsinki Institute for Information Technology HIIT, P.O. Box 68, 00014 Helsinki, Finland

A. Fischer
Institut für Neuroinformatik, Ruhr-Universität Bochum, 44780 Bochum, Germany
e-mail: asja.fischer@ini.rub.de

C. Igel (✉)
Department of Computer Science, University of Copenhagen, 2100 Copenhagen, Denmark
e-mail: igel@diku.dk

recently as building blocks of deep belief networks (Hinton and Salakhutdinov 2006). Learning an RBM corresponds to maximizing the likelihood of the parameters given data. Training large RBMs by steepest ascent on the log-likelihood gradient is in general computationally intractable, because the gradient involves averages over an exponential number of terms. Therefore, the computationally demanding part of the gradient is approximated by Markov chain Monte Carlo (MCMC, see, e.g., Neal 1993) methods usually based on Gibbs sampling (e.g., Hinton 2002; Tieleman and Hinton 2009; Desjardins et al. 2010). The higher the mixing rate of the Markov chain, the fewer sampling steps are usually required for a proper MCMC approximation. For RBM learning algorithms it has been shown that the bias of the approximation increases with increasing absolute values of the model parameters (Bengio and Delalleau 2009; Fischer and Igel 2011) and that this can indeed lead to severe distortions of the learning process (Fischer and Igel 2010). Thus, increasing the mixing rate of the Markov chains in RBM training is highly desirable.

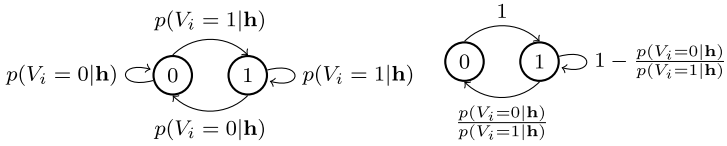
In this paper, we propose to employ a Metropolis-type transition operator for RBMs that maximizes the probability of state changes in the framework of periodic sampling and can lead to a faster mixing Markov chain. This operator is related to the *Metropolized Gibbs* sampler introduced by Liu (1996) and the *flip-the-spin* operator with Metropolis acceptance rule used in Ising models (see related methods in Sect. 3) and is, thus, referred to as *flip-the-state* operator. In contrast to these methods, our main theoretical result is that the proposed operator is also guaranteed to lead to an ergodic and thus properly converging Markov chain when using a periodic updating scheme (i.e., a deterministic scanning policy). It can replace Gibbs sampling in existing RBM learning algorithms without introducing computational overhead.

After a brief overview over RBM training and Gibbs sampling in Sects. 2, 3 introduces the flip-the-state transition operator and shows that the induced Markov chain converges to the RBM distribution. In Sect. 4 we empirically analyze the mixing behavior of the proposed operator compared to Gibbs sampling by looking at the second largest eigenvector modulus (SLEM), the autocorrelation time, and the frequency of class changes in sample sequences. While the SLEM describes the speed of convergence to the equilibrium distribution, the autocorrelation time concerns the variance of an estimate when averaging over several successive samples of the Markov chain. The class changes quantify mixing between modes in our test problems. Furthermore, the effects of the proposed sampling procedure on learning in RBMs is studied. We discuss the results and conclude in Sects. 5 and 6.

2 Background

An RBM is an undirected graphical model with a bipartite structure (Smolensky 1986; Hinton 2002) consisting of one layer of m visible variables $\mathbf{V} = (V_1, \ldots, V_m)$ and one layer of n hidden variables $\mathbf{H} = (H_1, \ldots, H_n)$ taking values $(\mathbf{v}, \mathbf{h}) \in \Omega := \{0, 1\}^{m+n}$. The modeled joint distribution is $p(\mathbf{v}, \mathbf{h}) = e^{-\mathcal{E}(\mathbf{v},\mathbf{h})} / \sum_{\mathbf{v},\mathbf{h}} e^{-\mathcal{E}(\mathbf{v},\mathbf{h})}$ with energy $\mathcal{E}$ given by $\mathcal{E}(\mathbf{v}, \mathbf{h}) = -\sum_{i=1}^{n} \sum_{j=1}^{m} w_{ij} h_i v_j - \sum_{j=1}^{m} b_j v_j - \sum_{i=1}^{n} c_i h_i$ with weights w_{ij} and biases b_j and c_i for $i \in \{1, \ldots, n\}$ and $j \in \{1, \ldots, m\}$, jointly denoted as $\boldsymbol{\theta}$. By $\mathbf{v}_{-i}$ and $\mathbf{h}_{-i}$ we denote the vectors of the states of all visible and hidden variables, respectively, except the ith one.

Typical RBM training algorithms perform steepest ascent on approximations of the log-likelihood gradient. One of the most popular is Contrastive Divergence (CD, Hinton 2002), which approximates the gradient of the log-likelihood by $-\sum_{\mathbf{h}} p(\mathbf{h}|\mathbf{v}^{(0)}) \frac{\partial \mathcal{E}(\mathbf{v}^{(0)},\mathbf{h})}{\partial \boldsymbol{\theta}} + \sum_{\mathbf{h}} p(\mathbf{h}|\mathbf{v}^{(k)}) \frac{\partial \mathcal{E}(\mathbf{v}^{(k)},\mathbf{h})}{\partial \boldsymbol{\theta}}$, where $\mathbf{v}^{(k)}$ is a sample gained after k steps of Gibbs sampling starting from a training example $\mathbf{v}^{(0)}$.



(a) Gibbs sampling (b) flip-the-state operator

Fig. 1 Transition diagrams for a single variable V_i (**a**) updated by Gibbs sampling (**b**) updated by the flip-the-state transition operator (here $p(V_i = 0|\mathbf{h}) < p(V_i = 1|\mathbf{h})$)

Several variants of CD have been proposed. For example, in Persistent Contrastive Divergence (PCD, Tieleman 2008) and its refinement Fast PCD (Tieleman and Hinton 2009) the Gibbs chain is not initialized by a training example but maintains its current value between approximation steps. Parallel Tempering (PT, also known as replica exchange Monte Carlo sampling) has also been applied to RBMs (Cho et al. 2010; Desjardins et al. 2010; Salakhutdinov 2010). It introduces supplementary Gibbs chains that sample from more and more smoothed variants of the true probability distribution and allows samples to swap between chains. This leads to faster mixing, but introduces computational overhead.

In general, a homogeneous Markov chain on a finite state space Ω with N elements can be described by an $N \times N$ transition probability matrix $\mathbf{A} = (a_{\mathbf{x},\mathbf{y}})_{\mathbf{x},\mathbf{y}\in\Omega}$, where $a_{\mathbf{x},\mathbf{y}}$ is the probability that the Markov chain being in state $\mathbf{x}$ changes its state to $\mathbf{y}$ in the next time step. We denote the one step transition probability $a_{\mathbf{x},\mathbf{y}}$ by $A(\mathbf{x},\mathbf{y})$, the n-step transition probability (the corresponding entry of the matrix $\mathbf{A}^n$) by $A^n(\mathbf{x},\mathbf{y})$. The transition matrices are also referred to as transition operators. We write $\mathbf{p}$ for the N-dimensional probability vector corresponding to some distribution p over Ω.

When performing periodic Gibbs sampling in RBMs, we visit all hidden and all visible variables alternately in a block-wise fashion and update them according to their conditional probability given the state of the other layer (i.e., $p(h_i|\mathbf{v})$, $i = 1,\ldots,n$ and $p(v_j|\mathbf{h})$, $j = 1,\ldots,m$, respectively). Thus, the Gibbs transition operator $\mathbf{G}$ can be decomposed into two operators $\mathbf{G}_h$ and $\mathbf{G}_v$ (with $\mathbf{G} = \mathbf{G}_h\mathbf{G}_v$) changing only the state of the hidden layer or the visible layer, respectively. The two operators can be further decomposed into a set of basic transition operators $\mathbf{G}_k$, $k = 1,\ldots,(n+m)$, each updating just a single variable based on the conditional probabilities. An example of such a transition of a single variable based on these probabilities is depicted in the transition diagram in Fig. 1(a).

3 The flip-the-state transition operator

In order to increase the mixing rate of the Markov chain, it seems desirable to change the basic transition operator of the Gibbs sampler in such a way that each single variable tends to change its state rather than sticking to the same state. This can be done by making the sample probability of a single neuron dependent on its current state. Transferring this idea to the transition graph shown in Fig. 1, this means that we wish to decrease the probabilities associated to the self-loops and increase the transition probabilities between different states as much as possible. Of course, we have to ensure that the resulting Markov chain remains ergodic and still has the RBM distribution p as equilibrium distribution.

The transition probabilities are maximized by scaling the probability for a single variable to change from the less probable state to the more probable state to one (making this transition deterministic) while increasing the transition in the reverse direction accordingly with the same factor. In the—in practice not relevant but for theoretical considerations

important—case of two states with the exact same conditional probability, we use the transition probabilities of Gibbs sampling to avoid a non-ergodic Markov chain.

These considerations can be formalized by first defining a variable v_i^* that indicates what the most probable state of the random variable V_i is or if both states are equally probable:

$$v_i^* = \begin{cases} 1, & \text{if } p(V_i = 1|\mathbf{h}) > p(V_i = 0|\mathbf{h}) \\ 0, & \text{if } p(V_i = 1|\mathbf{h}) < p(V_i = 0|\mathbf{h}) \\ -1, & \text{if } p(V_i = 1|\mathbf{h}) = p(V_i = 0|\mathbf{h}). \end{cases} \tag{1}$$

Now we define the flip-the-state transition operator $\mathbf{T}$ as follows:

Definition 1 For $i = 1, \ldots, m$, let the basic transition operator $\mathbf{T}_i$ for the visible unit V_i be defined through its transition probabilities: $T_i((\mathbf{v}, \mathbf{h}), (\mathbf{v}', \mathbf{h}')) = 0$ if $(\mathbf{v}, \mathbf{h})$ and $(\mathbf{v}', \mathbf{h}')$ differ in another variable than V_i and as

$$T_i\big((\mathbf{v}, \mathbf{h}), (\mathbf{v}', \mathbf{h}')\big) = \begin{cases} \frac{p(v_i'|h)}{p(v_i|h)}, & \text{if } v_i^* = v_i \neq v_i' \\ 1 - \frac{p(v_i'|h)}{p(v_i|h)}, & \text{if } v_i^* = v_i = v_i' \\ 1, & \text{if } v_i \neq v_i' = v_i^* \\ 0, & \text{if } v_i = v_i' \neq v_i^* \\ \frac{1}{2}, & \text{if } v_i^* = -1 \end{cases} \tag{2}$$

otherwise. The transition matrix containing the transition probabilities of the visible layer is given by $\mathbf{T}_v = \prod_i \mathbf{T}_i$. The transition matrix for the hidden layer $\mathbf{T}_h$ is defined analogously, and the flip-the-state transition operator is given by $\mathbf{T} = \mathbf{T}_h \mathbf{T}_v$.

3.1 Activation function & computational complexity

An RBM corresponds to a stochastic, recurrent neural network with activation function $\sigma(x) = 1/(1 + e^{-x})$. Similarly, the transition probabilities defined in (2) can be interpreted as resulting from an activation function depending not only on the weighted input to a neuron and its bias but also on the neuron's current state V_j (or analogously H_i):

$$\sigma'(x) = \begin{cases} \min\{e^x, 1\} & \text{if } V_j = 0 \\ \max\{1 - e^{-x}, 0\} & \text{if } V_j = 1. \end{cases} \tag{3}$$

Corresponding graphs are shown in Fig. 2.

The differences in computational complexity between the activation functions σ and σ' can be neglected. The transition operator described here requires a switch based on the current state of the neuron on the one hand, but saves the computationally expensive call to the random generator in deterministic transitions on the other hand. Furthermore, in the asymptotic case, the running time of a sampling step is dominated by the matrix multiplications, while the number of activation function evaluations in one step increases only linearly with the number of neurons.

If the absolute value of the sum of the weighted inputs and the bias is large (i.e., extreme high conditional probability for one of the two states), the transition probabilities between states under Gibbs sampling are already almost deterministic. Thus, the difference between $\mathbf{G}$ and $\mathbf{T}$ decreases in this case. This is illustrated in Fig. 2.

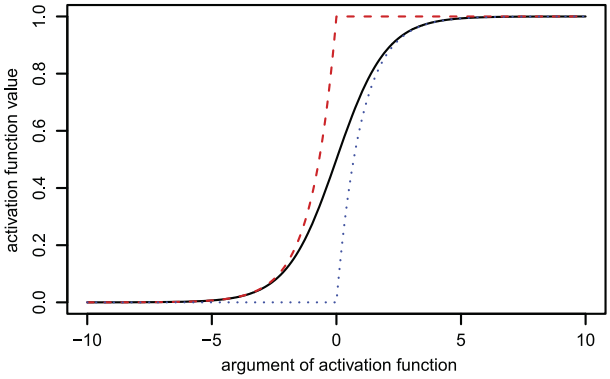


Fig. 2 Activation function for Gibbs sampling (*black*) and for transitions based on **T** when the current state is 0 (*red*, *dashed*) or 1 (*blue*, *dotted*) (Color figure online)

3.2 Related work

Both **G** and **T** are (local) Metropolis algorithms (Neal 1993). A Metropolis algorithm proposes states with a proposal distribution and accepts them in a way which ensures detailed balance. In this view, Gibbs sampling corresponds to using the proposal distribution "flip current state" and the Boltzmann acceptance probability $\frac{p(\mathbf{x}')}{p(\mathbf{x})+p(\mathbf{x}')}$, where $\mathbf{x}$ and $\mathbf{x}'$ denote the current and the proposed state, respectively. This proposal distribution has also been used with the Metropolis acceptance probability $\min(1, \frac{p(\mathbf{x}')}{p(\mathbf{x})})$ for sampling from Ising models. The differences between the two acceptance functions are discussed, for example, by Neal (1993). He comes to the conclusion that "the issues still remain unclear, though it appears that common opinion favours using the Metropolis acceptance function in most circumstances" (p. 69).

The work by Peskun (1973) and Liu (1996) shows that the Metropolis acceptance function is optimal with respect to the asymptotic variance of the Monte Carlo estimate of the quantity of interest. This result only holds if the variables to be updated are picked randomly in each step of the (local) algorithm. Thus, they are not applicable in the typical RBM training scenario, where block-wise sampling in a predefined order is used. In this scenario, it can indeed happen that the flip-the-state proposal combined with the Metropolis acceptance function leads to non-ergodic chains as shown by the counter-examples given by Neal (1993, p. 69).

The transition operator **T** also uses the Metropolis acceptance probability, but the proposal distribution differs from the one used in Ising models in one detail, namely that it selects a state at random if the conditional probabilities of both states are equal. This is important from a theoretical point of view, because it ensures ergodicity as proven in the next section. This is the reason why our method does not suffer from the problems mentioned above.

Furthermore, Breuleux et al. (2011) discuss a similar idea to the one underlying our transition operator as a theoretic framework for understanding fast mixing, where one increases the probability to change states by defining a new transition matrix $\mathbf{A}'$ based on an existing transition matrix $\mathbf{A}$ by $\mathbf{A}' = (\mathbf{A} - \lambda\mathbf{I})(1-\lambda)^{-1}$, where $\lambda \leq \min_{\mathbf{x}\in\Omega} A(\mathbf{x},\mathbf{x})$ and $\mathbf{I}$ is the identity matrix. Our method corresponds to applying this kind of transformation, not to the whole transition matrix, but rather to the transition probabilities of a single binary variable (i.e., the base transition operator). This makes the method not only computationally feasible in practice, but even more effective, because it allows us to redistribute more probability mass (because the redistribution is not limited by $\min_{\mathbf{x}\in\Omega} A(\mathbf{x},\mathbf{x})$), so that more than one entry of the new transition matrix is 0.

3.3 Properties of the transition operator

To prove that a Markov chain based on the suggested transition operator $\mathbf{T}$ converges to the probability distribution p defined by the RBM, it has to be shown that p is invariant with respect to $\mathbf{T}$ and that the Markov chain is irreducible and aperiodic.

As stated above, the described transition operator belongs to the class of local Metropolis algorithms. This implies that detailed balance holds for all the base transition operators (see, e.g., Neal 1993). If p is invariant w.r.t. the basic transition operators it is also invariant w.r.t. the concatenated transition matrix $\mathbf{T}$.

However, there is no general proof of ergodicity of Metropolis algorithms if neither the proposal distribution nor the acceptance distribution are strictly positive and the base transitions are applied deterministically in a fixed order. Therefore irreducibility and aperiodicity still remain to be proven (see, e.g., Neal 1993, p. 56).

To show irreducibility, we need some definitions and a lemma first. For a fixed hidden state $\mathbf{h}$ let us define $\mathbf{v}_{\max}(\mathbf{h})$ as the visible state that maximizes the probability of the whole state,

$$\mathbf{v}_{\max}(\mathbf{h}) := \arg\max_{\mathbf{v}} p(\mathbf{v}, \mathbf{h}), \tag{4}$$

and analogously

$$\mathbf{h}_{\max}(\mathbf{v}) := \arg\max_{\mathbf{h}} p(\mathbf{v}, \mathbf{h}). \tag{5}$$

We assume that arg max is unique and that ties are broken by taking the greater state according to some arbitrary predefined strict total order $\prec$.

Furthermore, let $\mathcal{M}$ be the set of states, for which the probability can not be increased by changing either only the hidden or only the visible states:

$$\mathcal{M} = \big\{(\mathbf{v}, \mathbf{h}) \in \Omega \,|\, (\mathbf{v}, \mathbf{h}) = \big(\mathbf{v}_{\max}(\mathbf{h}), \mathbf{h}\big) = \big(\mathbf{v}, \mathbf{h}_{\max}(\mathbf{v})\big)\big\}. \tag{6}$$

Note, that $\mathcal{M}$ is not the empty set, since it contains at least the most probable state $\arg\max_{(\mathbf{v},\mathbf{h})} p(\mathbf{v}, \mathbf{h})$. Now we have:

Lemma 1 *From every state* $(\mathbf{v}, \mathbf{h}) \in \Omega$ *one can reach* $(\mathbf{v}_{\max}(\mathbf{h}), \mathbf{h})$ *by applying the visible transition operator* $\mathbf{T}_v$ *once and* $(\mathbf{v}, \mathbf{h}_{\max}(\mathbf{v}))$ *in one step of* $\mathbf{T}_h$. *It is possible to reach every state* $(\mathbf{v}, \mathbf{h}) \in \Omega$ *in one step of* $\mathbf{T}_v$ *from* $(\mathbf{v}_{\max}(\mathbf{h}), \mathbf{h})$ *and in one step of* $\mathbf{T}_h$ *from* $(\mathbf{v}, \mathbf{h}_{\max}(\mathbf{v}))$.

Proof From the definition of $\mathbf{v}_{\max}(\mathbf{h})$ and the independence of the conditional probabilities of the visible variables given the state of the hidden layer it follows:

$$p\big(\mathbf{v}_{\max}(\mathbf{h})|\mathbf{h}\big) = \max_{v_1,\ldots,v_n} \prod_i p(v_i|\mathbf{h}). \tag{7}$$

Thus, in $\mathbf{v}_{\max}(\mathbf{h})$ every single visible variable is in the state with the higher conditional probability (i.e., in v_i^*) or both states are equally probable (in which case $v_i^* = -1$). By looking at the definition of the base transitions (2) it becomes clear that this means that $T_i((\mathbf{v}, \mathbf{h}), (v_{\max}(\mathbf{h})_i, \mathbf{v}_{-i}, \mathbf{h})) > 0$ and $T_i((\mathbf{v}_{\max}(\mathbf{h}), \mathbf{h}), (v_i, \mathbf{v}_{\max}(\mathbf{h})_{-i}, \mathbf{h})) > 0$. So we get for all $(\mathbf{v}, \mathbf{h}) \in \Omega$:

$$T_v\big((\mathbf{v}, \mathbf{h}), \big(\mathbf{v}_{\max}(\mathbf{h}), \mathbf{h}\big)\big) = \prod_i T_i\big((\mathbf{v}, \mathbf{h}), \big(v_{\max}(\mathbf{h})_i, \mathbf{v}_{-i}, \mathbf{h}\big)\big) > 0 \tag{8}$$

$$T_v\big((\mathbf{v}_{\max}(\mathbf{h}),\mathbf{h}),(\mathbf{v},\mathbf{h})\big)=\prod_i T_i\big((\mathbf{v}_{\max}(\mathbf{h}),\mathbf{h}),(v_i,\mathbf{v}_{\max}(\mathbf{h})_{-i},\mathbf{h})\big)>0. \tag{9}$$

This holds equivalently for the hidden transition operator $\mathbf{T}_h$ and $(\mathbf{v},\mathbf{h}_{\max}(\mathbf{v}))$. For all $(\mathbf{v},\mathbf{h})\in\Omega$:

$$T_h\big((\mathbf{v},\mathbf{h}),(\mathbf{v},\mathbf{h}_{\max}(\mathbf{v}))\big)=\prod_i T_i\big((\mathbf{v},\mathbf{h}),(\mathbf{v},h_{\max}(\mathbf{v})_i,\mathbf{h}_{-i})\big)>0 \tag{10}$$

$$T_h\big((\mathbf{v},\mathbf{h}_{\max}(\mathbf{v})),(\mathbf{v},\mathbf{h})\big)=\prod_i T_i\big((\mathbf{v},\mathbf{h}_{\max}(\mathbf{v})),(\mathbf{v},h_i,\mathbf{h}_{\max}(\mathbf{v})_{-i})\big)>0. \tag{11}$$

□

Now we prove the irreducibility:

Theorem 1 *The Markov chain induced by* $\mathbf{T}$ *is irreducible*:

$$\forall(\mathbf{v},\mathbf{h}),(\mathbf{v}',\mathbf{h}')\in\Omega:\exists n>0:\quad T^n\big((\mathbf{v},\mathbf{h}),(\mathbf{v}',\mathbf{h}')\big)>0. \tag{12}$$

Proof The proof is divided into three steps showing:

(i) from every state $(\mathbf{v},\mathbf{h})\in\Omega$ one can reach an element of $\mathcal{M}$ in a finite number of transitions, i.e., $\forall(\mathbf{v},\mathbf{h})\in\Omega\ \exists(\mathbf{v}^*,\mathbf{h}^*)\in\mathcal{M}$ and $n\in\mathbb{N}$, with $T^n((\mathbf{v},\mathbf{h}),(\mathbf{v}^*,\mathbf{h}^*))>0$,
(ii) for every state $(\mathbf{v},\mathbf{h})\in\Omega$ there exists a state $(\mathbf{v}^*,\mathbf{h}^*)\in\mathcal{M}$ from which it is possible to reach $(\mathbf{v},\mathbf{h})\in\Omega$ in a finite number of transitions, i.e., $\forall(\mathbf{v},\mathbf{h})\in\Omega\ \exists(\mathbf{v}^*,\mathbf{h}^*)\in\mathcal{M}$ and $n\in\mathbb{N}$ with $T^n((\mathbf{v}^*,\mathbf{h}^*),(\mathbf{v},\mathbf{h}))>0$, and
(iii) any transition between two arbitrary elements in $\mathcal{M}$ is possible, i.e., $\forall(\mathbf{v}^*,\mathbf{h}^*)$, $(\mathbf{v}^{**},\mathbf{h}^{**})\in\mathcal{M}:T((\mathbf{v}^*,\mathbf{h}^*),(\mathbf{v}^{**},\mathbf{h}^{**}))>0$.

Step (i): Let us define a sequence $((\mathbf{v}_k,\mathbf{h}_k))_{k\in\mathbb{N}}$ with $\mathbf{v}_0:=\mathbf{v}$, $\mathbf{h}_0:=\mathbf{h}$ and $\mathbf{h}_k:=\mathbf{h}_{\max}(\mathbf{v}_{k-1})$ and $\mathbf{v}_k:=\mathbf{v}_{\max}(\mathbf{h}_k)$ for $k>0$. From the definition of $\mathbf{v}_{\max}$ and $\mathbf{h}_{\max}$ it follows that $(\mathbf{v}_{k-1},\mathbf{h}_{k-1})\neq(\mathbf{v}_k,\mathbf{h}_k)$ unless $(\mathbf{v}_{k-1},\mathbf{h}_{k-1})\in\mathcal{M}$ and that no state in $\Omega\setminus\mathcal{M}$ is visited twice. The latter follows from the fact that in the sequence two successive states $(\mathbf{v}_k,\mathbf{h}_k)$ and $(\mathbf{v}_{k+1},\mathbf{h}_{k+1})$ from $\Omega\setminus\mathcal{M}$ have either increasing probabilities or $(\mathbf{v}_k,\mathbf{h}_k)\prec(\mathbf{v}_{k+1},\mathbf{h}_{k+1})$. Since Ω is a finite set, such a sequence must reach a state $(\mathbf{v}_n,\mathbf{h}_n)=(\mathbf{v}_{n+i},\mathbf{h}_{n+i})\in\mathcal{M}, i\in\mathbb{N}$ after a finite number of steps n.

Finally, this sequence can be produced by $\mathbf{T}$ since from Eq. (8) and Eq. (10) it follows that $\forall k>0$:

$$\begin{aligned}&T\big((\mathbf{v}_{k-1},\mathbf{h}_{k-1}),(\mathbf{v}_k,\mathbf{h}_k)\big)\\&\quad=T_h\big((\mathbf{v}_{k-1},\mathbf{h}_{k-1}),(\mathbf{v}_{k-1},\mathbf{h}_k)\big)\cdot T_v\big((\mathbf{v}_{k-1},\mathbf{h}_k),(\mathbf{v}_k,\mathbf{h}_k)\big)>0.\end{aligned} \tag{13}$$

Hence, one can get from $(\mathbf{v}_{k-1},\mathbf{h}_{k-1})$ to $(\mathbf{v}_k,\mathbf{h}_k)$ in one step of the transition operator $\mathbf{T}$.

Step (ii): We now consider a similar sequence $((\mathbf{v}_k,\mathbf{h}_k))_{k\in\mathbb{N}}$ with $\mathbf{v}_0:=\mathbf{v}$, $\mathbf{h}_0:=\mathbf{h}$ and $\mathbf{v}_k:=\mathbf{v}_{\max}(\mathbf{h}_{k-1})$ and $\mathbf{h}_k:=\mathbf{h}_{\max}(\mathbf{v}_k)$, for $k>0$. Again, there exists $n\in\mathbb{N}$, so that $(\mathbf{v}_n,\mathbf{h}_n)=(\mathbf{v}_{n+i},\mathbf{h}_{n+i})\in\mathcal{M}, i\in\mathbb{N}$. From equations (9) and (11) it follows that $\forall k>0$:

$$\begin{aligned}&T\big((\mathbf{v}_k,\mathbf{h}_k),(\mathbf{v}_{k-1},\mathbf{h}_{k-1})\big)\\&\quad=T_h\big((\mathbf{v}_k,\mathbf{h}_k),(\mathbf{v}_k,\mathbf{h}_{k-1})\big)\cdot T_v\big((\mathbf{v}_k,\mathbf{h}_{k-1}),(\mathbf{v}_{k-1},\mathbf{h}_{k-1})\big)>0.\end{aligned} \tag{14}$$

That is, one can get from $(\mathbf{v}_{k+1}, \mathbf{h}_{k+1})$ to $(\mathbf{v}_k, \mathbf{h}_k)$ in one step of the transition operator $\mathbf{T}$ and follow the sequence backwards from $(\mathbf{v}_n, \mathbf{h}_n) \in \mathcal{M}$ to $(\mathbf{v}, \mathbf{h})$.

Step (iii): From equations (8)–(11) it follows directly that a transition between two arbitrary points in $\mathcal{M}$ is always possible. □

Showing the aperiodicity is straight-forward:

Theorem 2 *The Markov chain induced by* $\mathbf{T}$ *is aperiodic*.

Proof For every state $(\mathbf{v}^*, \mathbf{h}^*)$ in the nonempty set $\mathcal{M}$ it holds that

$$T\big((\mathbf{v}^*, \mathbf{h}^*), (\mathbf{v}^*, \mathbf{h}^*)\big) > 0, \tag{15}$$

so the state is aperiodic. This means that the whole Markov chain is aperiodic, since it is irreducible (see, e.g., Brémaud 1999). □

Theorems 1 and 2 show that the Markov chain induced by the operator $\mathbf{T}$ has p as its equilibrium distribution, i.e., the Markov chain is ergodic with stationary distribution p.

4 Experiments

First, we experimentally compare the mixing behavior of the flip-the-state method with Gibbs sampling by analyzing $\mathbf{T}$ and $\mathbf{G}$ for random RBMs. Then, we study the effects of replacing $\mathbf{G}$ by $\mathbf{T}$ in different RBM learning algorithms applied to benchmark problems. After that, the operators are used to sample sequences from trained RBMs. The autocorrelation times and the number of class changes reflecting mode changes are compared. Training and sampling the RBMs was implemented using the open-source machine learning library Shark (Igel et al. 2008).

4.1 Analysis of the convergence rate

The convergence speed of an ergodic, homogeneous Markov chain with finite state space is governed by the second largest eigenvector modulus (SLEM). This is a direct consequence of the Perron-Frobenius theorem. Note that the SLEM computation considers absolute values, in contrast to the statements by Liu (1996) referring to the signed eigenvalues. We calculated the SLEM for transition matrices of Gibbs sampling and the new transition operator for small, randomly generated RBMs by solving the eigenvector equation of the resulting transition matrices $\mathbf{G}$ and $\mathbf{T}$. To handle the computational complexity we had to restrict our considerations to RBMs with only 2, 3, and 4 visible and hidden neurons, respectively. The weights of these RBMs were drawn randomly and uniformly from $[-c; c]$, with $c \in \{1, \ldots, 10\}$, and bias parameters were set to zero. For each value of c we generated 100 RBMs and compared the SLEMs of $\mathbf{G}$ and $\mathbf{T}$.

4.2 Log-likelihood evolution during training

We study the evolution of the exact log-likelihood, which is tractable if either the number of the hidden or the visible units is chosen to be small enough, during gradient-based training of RBMs using CD, PCD, or PT based on samples produced by Gibbs sampling and the flip-the-state transition operator.

We used three benchmark problems taken from the literature. Desjardins et al. (2010) consider a parametrized artificial problem, referred to as *Artificial Modes* in the following, for studying mixing properties. The inputs are 4×4 binary images. The observations are distributed around four equally likely basic modes, from which samples are generated by flipping pixels. The probability of flipping a pixel is given by the parameter p_{mut}, controlling the "effective distance between each mode" (Desjardins et al. 2010). In our experiments, p_{mut} was either 0.01 or 0.1. Furthermore, we used a 4×4 pixel version of *Bars and Stripes* (MacKay 2002) and finally the MNIST data set of handwritten digits.

In the small toy problems (*Artificial Modes* and *Bars and Stripes*) the number of hidden units was set to be the same as the number of visible units, i.e., $n = 16$. For MNIST the number of hidden units was set to 10. The RBMs were initialized with weights and biases drawn uniformly from a Gaussian distribution with 0 mean and standard deviation 0.01.

The models were trained on all benchmark problems using gradient ascent on the gradient approximation of either CD or PCD with k sampling steps (which we refer to as CD_k or PCD_k) or PT. Note that Contrastive Divergence learning with $k = 1$ does not seem to be a reasonable scenario for applying the new operator. The performance of PT depends on the number t of tempered chains and on the number of sampling steps k carried out in each tempered chain before swapping samples between chains. We call PT with t temperatures and k sampling steps t-PT_k. The inverse temperatures were distributed uniformly between 0 and 1. Samples for each learning method where either obtained by $\mathbf{G}$ or $\mathbf{T}$.

We performed mini-batch learning with a batch size of 100 training examples in the case of MNIST and *Artificial Modes* and batch learning for *Bars and Stripes*. The number of samples used for the gradient approximation was set to be equal to the number of training examples in a (mini) batch. We tested different learning rates $\eta \in \{0.01, 0.05, 0.1\}$ and used neither weight decay nor a momentum parameter. All experiments were run for a length of 20000 update steps and repeated 25 times. We calculated the log-likelihood every 100th step of training. In the following, all reported log-likelihood values are averaged over the training examples.

4.3 Autocorrelation analysis

To measure the mixing properties of the operators on larger RBMs, we performed an autocorrelation analysis.

We estimated the autocorrelation function

$$R(\Delta t) = \frac{\mathbb{E}[\mathcal{E}(\mathbf{V}_k, \mathbf{H}_k)\mathcal{E}(\mathbf{V}_{k+\Delta t}, \mathbf{H}_{k+\Delta t})]}{\sigma_{\mathcal{E}(\mathbf{V}_k, \mathbf{H}_k)}\sigma_{\mathcal{E}(\mathbf{V}_{k+\Delta t}, \mathbf{H}_{k+\Delta t})}}. \tag{16}$$

The random variables $\mathbf{V}_k$ and $\mathbf{H}_k$ are the state of the visible and hidden variables after running the chain for k steps. The expectation $\mathbb{E}$ is over all $k > 0$, and the standard deviation of a random variable X is denoted by σ_X. The autocorrelation function is always defined with respect to a specific function on the state space. Here the energy function $\mathcal{E}$ is a natural choice.

The autocorrelation time is linked to the asymptotic variance of an estimator based on averaging over consecutive samples from a Markov chain. It is defined as

$$\tau = \sum_{\Delta t=-\infty}^{\infty} R(\Delta t). \tag{17}$$

An estimator based on $l\tau$ consecutive samples from a Markov chain has the same variance as an estimator based on l independent samples (see, e.g., Neal 1993). In this sense τ consecutive samples are equivalent to one independent sample.

For the autocorrelation experiments we trained 25 RBMs on each of the previously mentioned benchmark problems with 20-PT_{10}. In addition to the RBMs with 10 hidden units we trained 24 RBMs with 500 hidden neurons on MNIST for 2000 parameter updates. To estimate the autocorrelations we sampled these RBMs for one million steps using **G** and **T**, respectively. We followed the recommendations by Thompson (2010) and, in addition to calculating and plotting the autocorrelations directly, fitted AR-models to the times series to estimate the autocorrelation time using the software package SamplerCompare (Thompson 2011).

4.4 Frequency of class changes

To access the ability of the two operators to mix between different modes, we observed the class changes in sample sequences, similar to the experiments by Bengio et al. (2013). We trained 25 RBMs with CD-5 on *Artificial Modes* with $p_{\text{mut}} = 0.01$ and $p_{\text{mut}} = 0.1$. After training, we sampled from the RBMs using either **T** or **G** as transition operator and analyzed how often subsequent samples belong to different classes. We considered four classes. Each class was defined by one of the four basic modes used to generate the dataset. A sample belongs to the same class as the mode to which it has the smallest Hamming distance. Ambiguous samples which could not be assigned to a single class, because they were equally close to at least two of the modes, were discarded. In one experimental setting, all trained RBMs were initialized 1000 times with samples drawn randomly from the training distribution (representing the starting distribution of CD learning), and the number of sampling steps before the first class change was measured. In a second setting, for each RBM one chain was started with all visible units set to one and run for 10000 steps. Afterwards, the number of class changes was counted.

5 Results and discussion

5.1 Analysis of the convergence rate

The upper plot in Fig. 3 shows the fraction of RBMs (out of 100) for which the corresponding transition operator **T** has a smaller SLEM than the Gibbs operator **G** (and therefore **T** promises a faster mixing Markov chain than **G**) in dependence on the value of c, which upper bounds the weights. If all the weights are equal to zero, Gibbs sampling is always better, but the higher the weights get the more often **T** has a better mixing rate. This effect is the more pronounced the more neurons the RBM has, which suggests that the results of our analysis can be transfered to real world RBMs.

In the hypothetical case that all variables are independent (corresponding to an RBM where all weights are zero), Gibbs sampling is optimal and converges in a single step. With the flip-the-state operator, however, the probability of a neuron to be in a certain state would oscillate and converge exponentially by a factor of $\frac{1-p(v_i^*)}{p(v_i^*)}$ (i.e., the SLEM of the base transition matrix in this case) to the equilibrium distribution. As the variables get more and more dependent, the behavior of Gibbs sampling is no longer optimal and the Gibbs chain converges more slowly than the Markov chain induced by **T**. Figure 3 directly supports our claim that in this relevant scenario changing states more frequently by the flip-the-state method can improve mixing.

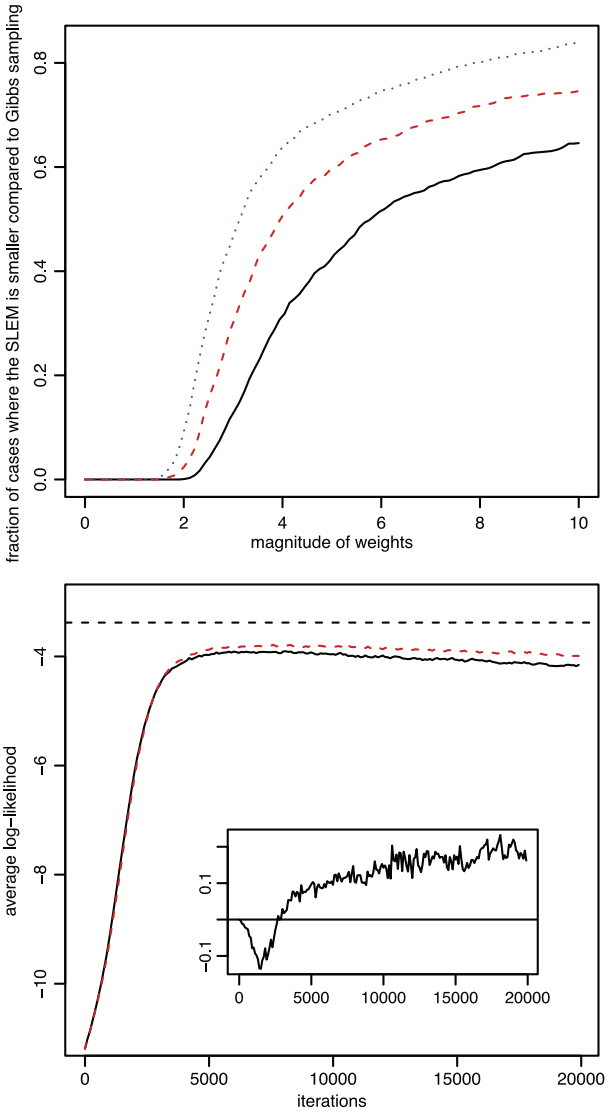


Fig. 3 The *upper figure* compares the mixing rates for 2×2 RBMs (*black*), 3×3 RBMs (*red*, *dashed*) and 4×4 RBMs (*blue*, *dotted*). The lower figure depicts the learning curves for CD_5 on *Bars and Stripes* with learning rate 0.05 using **G** (*black*) or **T** (*red*, *dashed*). The *inset* shows the difference between the two and is positive if the *red curve* is higher. The *dashed horizontal line* indicates the maximum possible value of the average log-likelihood (Color figure online)

5.2 Log-likelihood evolution during training

To summarize all trials of one experiment into a single value we calculated the maximum log-likelihood value reached during each run and finally calculated the median over all runs. The resulting maximum log-likelihood values for different experimental settings for learning the *Bars and Stripes* and the MNIST data set with CD and PT are shown in Table 1. Similar results were found for PCD and for experiments on *Artificial Modes*, see appendix. For most experimental settings, the RBMs reaches statistically significant higher likelihood values during training with the new transition operator (Wilcoxon signed-rank test, $p < 0.05$).

If we examine the evolution of likelihood values over time (as shown, e.g., in the lower plot of Fig. 3) more closely, we see that the proposed transition operator is better in the end

Table 1 Median maximum log-likelihood values on *Bars and Stripes* (top) and MNIST (bottom). Significant differences are marked with a star

Bars and Stripes			
Algorithm	η	Gibbs	**T**
CD_5	0.01	−4.070406	−3.986813*
CD_5	0.05	−3.832875	−3.727781*
CD_5	0.1	−3.838406	−3.732438*
CD_{10}	0.01	−3.963563	−3.930687*
CD_{10}	0.05	−3.640219	−3.57625*
CD_{10}	0.1	−3.635781	−3.589219*
5-PT_1	0.01	−4.011406	−4.0095
5-PT_1	0.05	−3.675312	−3.636125*
5-PT_1	0.1	−3.8255	−3.77825
5-PT_5	0.01	−3.928125	−3.918781
5-PT_5	0.05	−3.515719	−3.500844*
5-PT_5	0.1	−3.565281	−3.540625*
20-PT_1	0.01	−3.974219	−3.977562
20-PT_1	0.05	−3.548969	−3.524406*
20-PT_1	0.1	−3.577812	−3.549812*
20-PT_5	0.01	−3.917969	−3.923781
20-PT_5	0.05	−3.470094	−3.466188*
20-PT_5	0.1	−3.478844	−3.472594*
MNIST			
Algorithm	η	Gibbs	**T**
CD_5	0.01	−178.716	−177.958*
CD_5	0.05	−179.345	−178.873
CD_5	0.1	−179.007	−178.446*
CD_{10}	0.01	−176.495	−175.638*
CD_{10}	0.05	−176.844	−176.476
CD_{10}	0.1	−177.925	−176.586
10-PT_2	0.01	−182.283	−180.272*
10-PT_2	0.05	−182.303	−181.379*
10-PT_2	0.1	−181.727	−180.164
10-PT_5	0.01	−178.71	−178.215*
10-PT_5	0.05	−179.625	−178.708*
10-PT_5	0.1	−179.051	−178.504*

of training, but Gibbs sampling is actually slightly better in the beginning when weights are close to their small initialization. Learning curves as in Fig. 3 also show that if divergence occurs with Gibbs sampling (Fischer and Igel 2010), it will be slowed down, but not completely avoided with the new transition operator.

It is not surprising that Gibbs sampling mixes better at the beginning of the training, because the variables are almost independent when the weights are still close to their initial values near zero. Still, the results confirm that the proposed transition operator mixes better in the difficult phase of RBM training and that the faster mixing helps reaching better learning results.

Table 2 Mean autocorrelation times $\tau_{\mathbf{G}}$ and $\tau_{\mathbf{T}}$ for single Markov chains using the Gibbs sampler and the flip-the-state operator. The last column shows the gain defined as $1 - \frac{\tau_{\mathbf{T}}}{\tau_{\mathbf{G}}}$

	Gibbs $\tau_{\mathbf{G}}$	**T** $\tau_{\mathbf{T}}$	Gain in %
Bars and Stripes	22.46	20.06	10.67
Artificial Modes, $p_{\text{mut}} = 0.1$	3.16	2.19	30.73
Artificial Modes, $p_{\text{mut}} = 0.01$	6.00	5.94	1.06
MNIST, $n = 10$	488.26	445.84	8.69
MNIST, $n = 500$	522.39	432.12	17.28

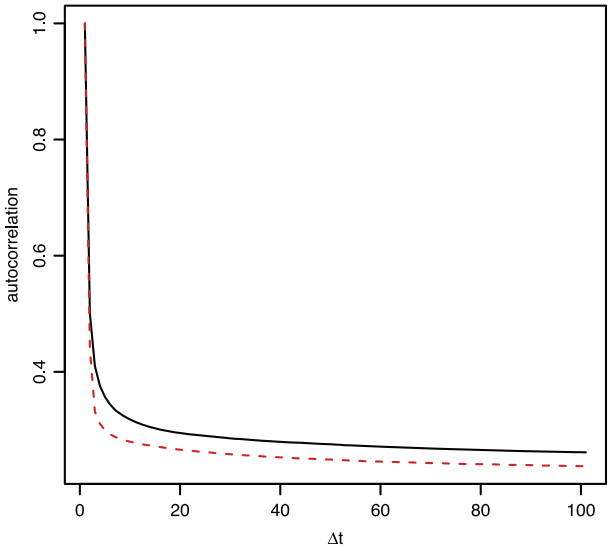


Fig. 4 Autocorrelation function $R(\Delta t)$ for RBMs with 500 hidden neurons trained on MNIST based on 24 trials, sampled 10^6 steps each. The *dotted line* corresponds to **T** and the *solid one* to **G** (Color figure online)

The results suggest that it may be reasonable to mix the two operators. Either, one could start with **G** and switch to **T** as the weights grow larger, or one can softly blend between the basic operators and consider $\mathbf{T}_i^{\alpha} = \alpha \mathbf{T}_i + (1 - \alpha)\mathbf{G}_i$, $\alpha \in [0, 1]$.

5.3 Autocorrelation analysis

The autocorrelation analysis revealed that sampling using the flip-the-state operator leads to shorter autocorrelation times in the considered benchmark problems, see Table 2 and Fig. 4. For example, an RBM trained on MNIST with 500 hidden neurons needed on average to be sampled for 17.28 % fewer steps to achieve the same variance of the estimate if **T** is used instead of **G**—without overhead in computation time or implementation complexity. The results with $n = 500$ demonstrate that our previous findings carry over to larger RBMs.

5.4 Frequency of class changes

The numbers of class changes observed in sequences of 10000 samples starting from the visible nodes set to one produced by **G** and **T** are given in Table 3. Table 4 shows the number of samples before the first class change when initializing the Markov chain with samples randomly drawn from the training distribution. Markov chains based on **T** led to more and faster class changes than chains using Gibbs sampling. As the modes in the training set get

Table 3 Frequencies of class changes for the Gibbs sampler and the flip-the-state operator in sequences of 10000 samples (medians and quantiles over samples from 25 RBMs)

	25 % quantile	Median	75 % quantile
Artificial Modes, $p_{\text{mut}} = 0.1$, **G**	615	637	655
Artificial Modes, $p_{\text{mut}} = 0.1$, **T**	919	944	958
Artificial Modes, $p_{\text{mut}} = 0.01$, **G**	134	148	162
Artificial Modes, $p_{\text{mut}} = 0.01$, **T**	175	186	199

Table 4 Number of samples before the first class change when starting a Markov chain with samples from the training distribution (medians and quantiles over samples from 25 RBMs)

	25 % quantile	Median	75 % quantile
Artificial Modes, $p_{\text{mut}} = 0.1$, **G**	6	13	25
Artificial Modes, $p_{\text{mut}} = 0.1$, **T**	3	7	14
Artificial Modes, $p_{\text{mut}} = 0.01$, **G**	16	41	96
Artificial Modes, $p_{\text{mut}} = 0.01$, **T**	10	27	63

more distinct (comparing $p_{\text{mut}} = 0.1$ to $p_{\text{mut}} = 0.01$) class changes get less frequent and more sampling steps are needed to yield a class change. Nevertheless, **T** is superior to **G** even in this setting.

6 Conclusions

We proposed the flip-the-state transition operator for MCMC-based training of RBMs and proved that it induces a converging Markov chain. Large weights lead to slow mixing Gibbs chains that can severely harm RBM training. In this scenario, the proposed flip-the-state method increases the mixing rate compared to Gibbs sampling. The way of sampling is generally applicable in the sense that it can be employed in every learning method for binary RBMs relying on Gibbs sampling, for example Contrastive Divergence learning and its variants as well as Parallel Tempering. As empirically shown, the better mixing indeed leads to better learning results in practice. As the flip-the-state sampling does not introduce computational overhead, we see no reason to stick to standard Gibbs sampling.

Acknowledgements This work has been supported by the German Federal Ministry of Education and Research within the National Network Computational Neuroscience under grant number 01GQ0951 (Bernstein Fokus "Learning behavioral models: From human experiment to technical assistance").

Appendix: Log-likelihood values for different problems, algorithms, and experimental settings

Table 5 Median maximum log-likelihood values for different experimental settings for learning *Bars and Stripes* (top) and MNIST (bottom). Significant differences are marked with a star

Bars and Stripes			
Algorithm	η	Gibbs	**T**
PCD_1	0.01	−4.944813*	−5.131875
PCD_1	0.05	−4.917219	−4.754625*
PCD_1	0.1	−5.285469	−5.176563*
PCD_5	0.01	−4.067437	−4.000844*
PCD_5	0.05	−4.050906	−3.915938*
PCD_5	0.1	−4.209375	−4.124625*
PCD_{10}	0.01	−3.972812	−3.945219*
PCD_{10}	0.05	−3.8425	−3.769938*
PCD_{10}	0.1	−4.02	−3.917937*
MNIST			
Algorithm	η	Gibbs	**T**
PCD_1	0.01	−185.536	−185.378*
PCD_1	0.05	−181.382	−180.905
PCD_1	0.1	−179.572	−180.502
PCD_5	0.01	−179.061	−177.939*
PCD_5	0.05	−178.195	−177.675
PCD_5	0.1	−176.897	−175.946
PCD_{10}	0.01	−175.799	−174.919*
PCD_{10}	0.05	−176.301	−175.446*
PCD_{10}	0.1	−176.208	−174.94

Table 6 Median maximum log-likelihood values for different experimental settings for learning *Artificial Modes*. The top table shows the results for datasets generated with a probability p_{mut} of permuting each pixel of 0.1, the bottom table for $p_{\text{mut}} = 0.01$. Significant differences are marked with a star

Artificial Modes, $p_{\text{mut}} = 0.1$

Algorithm	η	Gibbs	**T**
CD_5	0.01	−6.79103	−6.78603*
CD_5	0.05	−6.80241	−6.79473*
CD_{10}	0.01	−6.78564	−6.7833*
CD_{10}	0.05	−6.79646	−6.79682*
PCD_5	0.01	−6.79176	−6.78537*
PCD_5	0.05	−6.80292	−6.79679*
PCD_{10}	0.01	−6.78512	−6.78329*
PCD_{10}	0.05	−6.79575	−6.79372*
10-PT_2	0.01	−6.78282	−6.78325
10-PT_2	0.05	−6.79839	−6.79575
10-PT_5	0.01	−6.78206	−6.78213
10-PT_5	0.05	−6.7929	−6.79351
10-PT_{10}	0.01	−6.7827	−6.78203
10-PT_{10}	0.05	−6.79101	−6.79196

Artificial Modes, $p_{\text{mut}} = 0.01$

Algorithm	η	Gibbs	**T**
CD_5	0.01	−4.01644	−3.64562*
CD_5	0.05	−4.00452	−3.68796*
CD_{10}	0.01	−3.48928	−3.23056*
CD_{10}	0.05	−3.51262	−3.27728*
PCD_5	0.01	−3.9956	−3.6295*
PCD_5	0.05	−3.94864	−3.61392*
PCD_{10}	0.01	−3.46321	−3.21007*
PCD_{10}	0.05	−3.37534	−3.11163*
10-PT_2	0.01	−2.40041	−2.40195
10-PT_2	0.05	−2.40682	−2.40798
10-PT_5	0.01	−2.39929	−2.3992
10-PT_5	0.05	−2.40648	−2.40072*
10-PT_{10}	0.01	−2.39973	−2.40012
10-PT_{10}	0.05	−2.40188	−2.40078

References

Bengio, Y., & Delalleau, O. (2009). Justifying and generalizing contrastive divergence. *Neural Computation*, *21*(6), 1601–1621.

Bengio, Y., Mesnil, G., Dauphin, Y., & Rifai, S. (2013). Better mixing via deep representations. *Journal of Machine Learning Research Workshop and Conference Proceedings*, *28*(1), 552–560.

Brémaud, P. (1999). *Markov chains: Gibbs fields*, *Monte Carlo simulation, and queues*. Berlin: Springer.

Breuleux, O., Bengio, Y., & Vincent, P. (2011). Quickly generating representative samples from an RBM-derived process. *Neural Computation*, *23*(8), 2058–2073.

Cho, K., Raiko, T., & Ilin, A. (2010). Parallel tempering is efficient for learning restricted Boltzmann machines. In *Proceedings of the international joint conference on neural networks (IJCNN 2010)* (pp. 3246–3253). New York: IEEE Press.

Desjardins, G., Courville, A., Bengio, Y., Vincent, P., & Dellaleau, O. (2010). Parallel tempering for training of restricted Boltzmann machines. *Journal of Machine Learning Research Workshop and Conference Proceedings 9*(AISTATS 2010), 145–152.

Fischer, A., & Igel, C. (2010). Empirical analysis of the divergence of Gibbs sampling based learning algorithms for Restricted Boltzmann Machines. In K. Diamantaras, W. Duch, & L. S. Iliadis (Eds.), *LNCS: Vol. 6354. International conference on artificial neural networks (ICANN 2010)* (pp. 208–217). Berlin: Springer.

Fischer, A., & Igel, C. (2011). Bounding the bias of contrastive divergence learning. *Neural Computation*, *23*, 664–673.

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

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

Igel, C., Glasmachers, T., & Heidrich-Meisner, V. (2008). Shark. *Journal of Machine Learning Research*, *9*, 993–996.

Liu, J. S. (1996). Metropolized independent sampling with comparisons to rejection sampling and importance sampling. *Statistics and Computing*, *6*, 113–119.

MacKay, D. J. C. (2002). *Information theory, inference & learning algorithms*. Cambridge: Cambridge University Press.

Neal, R. M. (1993). *Probabilistic inference using Markov chain Monte Carlo methods*. Technical Report CRG-TR-93-1, Department of Computer Science, University of Toronto.

Peskun, P. H. (1973). Optimum Monte-Carlo sampling using Markov chains. *Biometrika*, *60*(3), 607–612.

Salakhutdinov, R. (2010). Learning in Markov random fields using tempered transitions. In Y. Bengio, D. Schuurmans, J. Lafferty, C. K. I. Williams, & A. Culotta (Eds.), *Advances in neural information processing systems* (Vol. 22, pp. 1598–1606).

Smolensky, P. (1986). Information processing in dynamical systems: foundations of harmony theory. In D. E. Rumelhart & J. L. McClelland (Eds.), *Parallel distributed processing: explorations in the microstructure of cognition, Vol. 1: foundations* (pp. 194–281). Cambridge: MIT Press.

Thompson, M. B. (2010). *A comparison of methods for computing autocorrelation time*. Tech. Rep. 1007, Department of Statistics, University of Toronto.

Thompson, M. B. (2011). Introduction to SamplerCompare. *Journal of Statistical Software*, *43*(12), 1–10.

Tieleman, T. (2008). Training restricted Boltzmann machines using approximations to the likelihood gradient. In W. W. Cohen, A. McCallum, & S. T. Roweis (Eds.), *International conference on machine learning (ICML)* (pp. 1064–1071). New York: ACM.

Tieleman, T., & Hinton, G. E. (2009). Using fast weights to improve persistent contrastive divergence. In A. Pohoreckyj Danyluk, L. Bottou, & M. L. Littman (Eds.), *International conference on machine learning (ICML)* (pp. 1033–1040). New York: ACM.