Journal of Statistical Mechanics:
Theory and Experiment

PAPER

Equilibrium and non-equilibrium regimes in the learning of restricted Boltzmann machines*

To cite this article: Aurélien Decelle *et al J. Stat. Mech.* (2022) 114009

Journal of Statistical Mechanics: Theory and Experiment
An IOP and SISSA journal

PAPER: ML 2022

Equilibrium and non-equilibrium regimes in the learning of restricted Boltzmann machines[*]

Aurélien Decelle[1,2,], Cyril Furtlehner[2] and Beatriz Seoane[1]**

[1] Departamento de Física Teórica, Universidad Complutense, 28040 Madrid, Spain
[2] Université Paris-Saclay, CNRS, INRIA Tau team, LISN, 91190 Gif-sur-Yvette, France
E-mail: adecelle@ucm.es and beseoane@ucm.es

Received 6 June 2021
Accepted for publication 4 July 2022
Published 24 November 2022

Online at stacks.iop.org/JSTAT/2022/114009
https://doi.org/10.1088/1742-5468/ac98a7

Abstract. Training restricted Boltzmann machines (RBMs) have been challenging for a long time due to the difficulty of precisely computing the log-likelihood gradient. Over the past few decades, many works have proposed more or less successful training recipes but without studying the crucial quantity of the problem: the mixing time, i.e. the number of Monte Carlo iterations needed to sample new configurations from a model. In this work, we show that this mixing time plays a crucial role in the dynamics and stability of the trained model, and that RBMs operate in two well-defined regimes, namely equilibrium and out-of-equilibrium, depending on the interplay between this mixing time of the model and the number of steps, k, used to approximate the gradient. We further show empirically that this mixing time increases with the learning, which often implies a transition from one regime to another as soon as k becomes smaller than this time. In particular, we show that using the popular k (persistent) contrastive divergence approaches, with k small, the dynamics of the learned model are extremely slow and often dominated by strong out-of-equilibrium effects. On

[*] This article is an updated version of: Decelle A, Furtlehner C and Seoane B 2021 Equilibrium and non-equilibrium regimes in the learning of restricted Boltzmann machines *Advances in Neural Information Processing Systems* vol 34 ed M Ranzato, A Beygelzimer, Y Dauphin, P S Liang and J Wortman Vaughan (New York: Curran Associates) pp 5345–59.

[**] Author to whom any correspondence should be addressed.

the contrary, RBMs trained in equilibrium display faster dynamics, and a smooth convergence to dataset-like configurations during the sampling. Finally, we discuss how to exploit in practice both regimes depending on the task one aims to fulfill: (i) short k can be used to generate convincing samples in short learning times, (ii) large k (or increasingly large) is needed to learn the correct equilibrium distribution of the RBM. Finally, the existence of these two operational regimes seems to be a general property of energy based models trained via likelihood maximization.

Keywords: machine learning, memory effects

S Supplementary material for this article is available online

Contents

1. Introduction

Restricted Boltzmann machines (RBMs) are one of the generative stochastic neural networks that have been available for decades [1] and are renowned for their capacity to learn complex datasets [2] and draw similar new data. Furthermore, the hidden correlations found in the data are easily accessible [3–5], which is particularly interesting for a scientific use of Big Data. Despite these positive features, RBMs still remain hard to train and evaluate, and hence to use in practical problems. In comparison, generative adversarial networks (GANs) [6] (for which it is much harder to find a working architecture) are more commonly used, not only because of the advantages given by convolutional layers, but also because the generative process does not depend on a costly sampling procedure with an uncontrolled convergence time.

Both training and data generation of RBMs are tricky and unpredictable in inexperienced hands. One of the main difficulties is that it is hard to evaluate if the learning is progressing or not. Yet, many works in the past twenty years have proposed recipes for good practices in RBM training [7]. Unfortunately, the evaluation of these recipes relies only on the comparison of the values reached by the log-likelihood (LL) [7–9], a quantity that cannot be monitored in tractable times during the learning process. Studies properly characterizing the quality, independence, or stability of the generated samples using the different recipes are nearly absent in the literature. In fact, many works just show a set of new samples (for eye evaluation) that were either obtained after a short Markov chain Monte Carlo (MCMC) sampling initialized at the dataset, or borrowed from the permanent chain (negative particle) at the end of the training [10]. In other studies, a reconstruction error after one or several MCMC steps is used to determined if the machine was correctly trained [11]. However, none of these measures guarantee that the trained model can generate from scratch new samples nor that the dataset are typical samples of that model. At most, these tests assure you that samples of the dataset are locally stable w.r.t. few MCMC steps, unless a proper decorrelation from the initial conditions or a link to equilibrium properties are investigated. Finally, more recent works use a classification score on the activation of the latent variables [12, 13], which reflects that the relevant features were learned, but is silent about the generative power of the machine.

RBMs have been receiving increasing attention in life and pure sciences in recent years [5, 14–21] given their potential for interpretability of the patterns learned by the machine (a rare feature in machine learning). Indeed, the RBM formalism enables the extraction of the (unknown) probability distribution function of the dataset. Yet, such an approach is meaningful if the dataset is related to the equilibrium properties of the trained model. Given these new and exciting prospectives for the RBMs, it is more important than ever to establish reproducible protocols and evaluation tools to guarantee not only that the model generates good enough data, but also that the dataset are typical equilibrium samples of the model.

In this work, we demonstrate that the classical training procedures can lead to two distinct regimes of the RBM: an equilibrium and an out-of-equilibrium (OOE) one. A finding that had been recently discussed in the literature, under the name of convergent or non-convergent regimes in a different family of generative energy-based models (EBMs), is those whose energy function is a ConvNet [22]. Here, at variance to previous approaches, we show that the equilibrium regime is observed if the number of MCMC steps, k, used to estimate the LL gradient, is larger than the algorithm mixing time (MT). The latter emerges when k falls below this MT. In particular, we show that for a classical dataset such as MNIST, the MT is always rather large and grows with the learning time. This implies that learning an ‘equilibrium’ model (i.e. a RBM that operates in the equilibrium regime) is costly, and that most of the works in the literature operate without any doubt in the OOE regime. In this regime, the equilibrium distribution of the machine at the end of the learning is significantly different from that of the dataset. Yet, despite this handicap, we show the OOE regime can be exploited to generate good samples at short training times, an observation that has already been exploited in practice [23] for convolution EBM. All our conclusions rely on the study of nine datasets: MNIST [24], whose results are discussed in the main-text, FashionMNIST [25], Caltech101 Silhouettes, smallNORB dataset [26], a human genome dataset [27] (with similar dimensions as MNIST but more structured), the high-quality CelebA [28] projected in black and white, and in low definition but in color, and CIFAR [29]. The analysis of most of the last datasets is discussed only in the supplementary material (SM) (https://stacks.iop.org/JSTAT/2022/114009/mmedia) and show a similar behavior to the one observed on MNIST.

Our paper is organized as follows. RBMs, their general principles and learning equations are defined in section 2. We review previous related works in section 3. In section 4, we define the different observables used to monitor the training and sampling. Finally, we discuss experimental results in section 5.

2. Definition of the model

An RBM is a Markov random field with pairwise interactions defined on a bipartite graph of two non-interacting layers of variables: the visible nodes, $\boldsymbol{v} = v_i, i = 1, \ldots, N_v$, represent the data, while the hidden ones, $\boldsymbol{h} = h_j, j = 1, \ldots, N_h$, are the latent representations of this data and are there to build arbitrary dependencies between the visible units (see sketch in figure 1(A)). Usually, the nodes are binary-valued in $\{0, 1\}$, yet Gaussian or more arbitrary distributions on real-valued bounded support are also used, ultimately making RBMs adaptable for more heterogeneous datasets. Here, we deal only with binary $\{0, 1\}$ variables for both the visible and hidden nodes. Other approaches using truncated-Gaussian hidden units [30], giving a ReLu type of activation functions for the hidden layer, do work well, but in our experiments, we observed qualitatively the similar dynamical behavior and therefore we will stick to binary hidden units for the rest of the article. The energy function of an RBM is taken as

$$E[\boldsymbol{v}, \boldsymbol{h}\,; \boldsymbol{w}, \boldsymbol{b}, \boldsymbol{c}] = -\sum_{ia} v_i w_{ia} h_a - \sum_i b_i v_i - \sum_a c_a h_a, \tag{1}$$

with $\boldsymbol{w}$ being the weight matrix and $\boldsymbol{b}$, $\boldsymbol{c}$ the visible and hidden biases, respectively. The Boltzmann distribution is then given by

$$p[\boldsymbol{v}, \boldsymbol{h}|\boldsymbol{w}, \boldsymbol{b}, \boldsymbol{c}] = \frac{\exp(-E[\boldsymbol{v}, \boldsymbol{h}\,; \boldsymbol{w}, \boldsymbol{b}, \boldsymbol{c}])}{Z} \quad \text{with } Z = \sum_{\{\boldsymbol{v}, \boldsymbol{h}\}} \mathrm{e}^{-E[\boldsymbol{v}, \boldsymbol{h}]}, \tag{2}$$

being the partition function of the system. RBMs are usually trained using gradient ascent of the LL function of the training dataset $\mathcal{D} = \{\boldsymbol{v}^{(1)}, \ldots, \boldsymbol{v}^{(M)}\}$, with the LL being defined as

$$\mathcal{L}(\boldsymbol{w}, \boldsymbol{b}, \boldsymbol{c}|\mathcal{D}) = M^{-1} \sum_{m=1}^{M} \ln\, p(\boldsymbol{v} = \boldsymbol{v}^{(m)}|\boldsymbol{w}, \boldsymbol{b}, \boldsymbol{c}) = M^{-1} \sum_{m=1}^{M} \ln \sum_{\{\boldsymbol{h}\}} \mathrm{e}^{-E[\boldsymbol{v}^{(m)}, \boldsymbol{h}\,; \boldsymbol{w}, \boldsymbol{b}, \boldsymbol{c}]} - \ln\, Z. \tag{3}$$

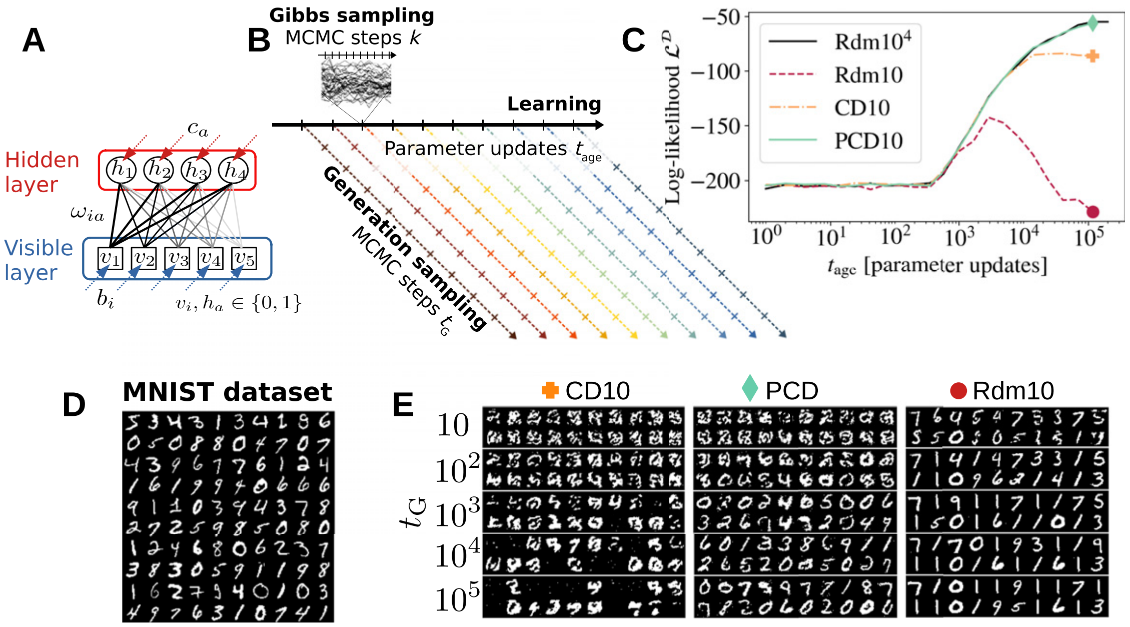

Figure 1. Sketch of an RBM in (A) and of the pipeline of our analysis in (B). (C) Evolution of the LL of the dataset during the learning for four different RBMs trained using either a very long MCMC sampling to estimate the gradient ($k = 10^4$, Rdm-10^4) or short samplings $k = 10$ following different recipes for the initialization of the MC chains (CD-10, PCD-10 and Rdm-10). The large symbols mark the LL of the RBMs used to generate new samples in (E). In (D) we show a subset of the MNIST database, and in (E) a subset of the samples drawn by the $k = 10$ RBMs marked in (C) through a MCMC sampling of the equilibrium Gibbs measure initialized at random and after 10, 10^2, 10^3, 10^4 and 10^5 MCMC steps. Note that in our setup the MC chain are initialized at random, and we can observe that RBMs trained with CD-10 do not converge to any digit despite reaching higher value of the LL than Rdm10, which instead manage to create digits quite fast. Also note the lack of diversity of Rdm10 for large sampling time.

The gradient is then composed of two terms: the first accounting for the interaction between the RBM's response and the training set, and same for the second, but using the samples drawn by the machine itself. The expression of the LL gradient w.r.t. all the parameters is given by

$$\frac{\partial \mathcal{L}}{\partial w_{ia}} = \langle v_i h_a \rangle_{\mathcal{D}} - \langle v_i h_a \rangle_{\mathcal{H}}, \qquad \frac{\partial \mathcal{L}}{\partial b_i} = \langle v_i \rangle_{\mathcal{D}} - \langle v_i \rangle_{\mathcal{H}} \quad \text{and} \quad \frac{\partial \mathcal{L}}{\partial c_a} = \langle h_a \rangle_{\mathcal{D}} - \langle h_a \rangle_{\mathcal{H}}, \tag{4}$$

where $\langle f(\boldsymbol{v}, \boldsymbol{h}) \rangle_{\mathcal{D}} = M^{-1} \sum_m \sum_{\{\boldsymbol{h}\}} f(\boldsymbol{v}^{(m)}, \boldsymbol{h}) p(\boldsymbol{h}|\boldsymbol{v}^{(m)})$ denotes an average over the dataset, and $\langle f(\boldsymbol{v}, \boldsymbol{h}) \rangle_{\mathcal{H}}$, the average over the Boltzmann measure in equation (2).

A large part of the literature on RBMs focuses on schemes to approximate the rhs of equation (4). In theory, these terms can be computed up to an arbitrary precision using parallel MCMC simulations as long as the number of MC steps are large enough to ensure a proper sampling of the equilibrium configuration space. In other words, sampling times should be larger than the MCMC MT. A naive way to implement this idea is to initialize each of these parallel Markov chains on random conditions, perform k MCMC steps, and use the final configurations to estimate the gradient. We call this scheme **Rdm-k**. This scheme is not used in practice because it would require too many k steps if the MT is too large. Many recipes have been proposed to shorten this sampling and approximate the negative term of the gradient, but since the introduction of the so-called contrastive divergence (CD) by Hinton [31], only a limited number of schemes are used in practice. The first one, **CD-k**, initializes the MCMC simulation from the same data of the mini-batch used to compute the gradient, and performs $k \sim \mathcal{O}(1)$ sampling steps. This approximation relies on the idea that the dataset must be a good approximation of the equilibrium samples of a well-trained RBM, something that is not true for a poorly trained machine, whose typical samples are quite distant from the dataset. In the second commonly used method, the last configurations of the Markov chains are saved from one parameter update to the other, and then used to initialize the subsequent chain. As in the other methods, k MCMC steps are used to estimate the gradient for each update. This method is known as persistent-CD: **PCD-k** [32], and takes advantage of the fact that the RBM's parameters change smoothly during the learning and the same is expected for the models' equilibrium distribution[3]. Finally, other approximations, such as mean-field or TAP equations [10], or more elaborate approaches, such as the parallel tempering technique [33, 34], can be used. A common feature of all these recipes is that they rely on very few simulation steps to sample the Boltzmann distribution, and the trained model often generates rather bad configurations.

To illustrate this last comment, let us discuss a provocative experiment shown in figure 1(E), where we try to sample new digits just by sampling the equilibrium measure of our trained RBMs using a MCMC initialized uniformly at random. Let us stress that this is not the standard set-up used in the field to generate new samples, but it is a strong test of the generation power of the trained model. We compare the outcome of the generated samples (as function of the sampling time) for three RBMs trained each following different recipes to compute the negative term of the gradient: CD-10, PCD-10 and Rdm-10. At this point, $k = 10$ might seem a very small number. It is however the typical order of magnitude used in the literature. The details of the learning protocol are discussed later in the text and summarized in the SM. The first striking point in figure 1(E) is the inability of the CD-10 RBM to properly sample digits in this set-up. This problem has been reported before [8, 35], but since CD is part of the standard methods, we want to insist on this point. In addition, PCD-10 trained RBM generates proper digits but only after a very long sampling time, while with Rdm-10, quite surprisingly, digits are properly generated after only 10 MCMC steps. Yet, samples obtained after much longer MCMC steps have unbalanced digits representations, like the 1 being over-represented. In fact, we observe that poorly trained models generate samples that do not respect the correct balance between the 10 digits present in the original database (see figure 1(D)). We already mentioned the 1s in the Rdm-10 run,

[3] This adiabatic scheme is thus valid as long as the relaxation time for the RBM parameters during the learning is much larger than the MCMC mixing times.

https://doi.org/10.1088/1742-5468/ac98a7

but the same happens (to a lesser extent) with the 0 in the PCD-10 case. Alternatively to these generated samples, we show the evolution of the LL during the learning of these same three RBMs in figure 1(C). By definition of the loss function, the LL is intended to be maximized, so it constitutes a traditional measure of the quality of the training. The LL of the models used for the sampling of figure 1(E) are highlighted in large symbols. This plot illustrates the lack of reliability of the LL to monitor the quality of the learning. For instance, CD-10 reaches quite high values of LL and yet, it cannot generate a single digit. As Rdm-10 is the other way around, it generates decent samples but with a very poor LL. Finally, we observe that the LL of PCD-10 reaches very high values, comparable to RBMs trained using up to $k = 10^4$ MCMC steps to estimate the gradient. We will come back to this point. Our experiment highlights, first, that the LL cannot be used all alone to quantify the generative power of an RBM, and second, that the quality of the generated samples may variate in a non-monotonic way during the generation sampling.

3. Related works

Recent years have witnessed a large number of studies trying to understand and control the learning of RBMs. In these works, meta-parameters are varied at the same time that an approximation of the LL [8, 36] of the training/test set is monitored throughout the learning process. The stability of the evolution of the LL and the maximum value achieved, are then used to compare the different recipes and parameters. For instance, in [8] the LL of different learning schemes are evaluated and the samples are inspected visually. Unfortunately, no estimator is used to assess the quality of the samples, nor is clear the precise protocol used to generate these samples. In [9], the authors investigate the stability of the learning of RBMs with binary inputs[4] by comparing the learned features. They show (mostly using the LL and an eye inspection of the learned features) that a centered version of the gradient can be much more stable. In [7], the authors study, in a systematic way, the convergence properties of CD, PCD and PT on several small toy models that can be analyzed exactly, that is, where the LL can be computed by brute-force. We can find more recent works [37–40] improving the learning scheme for RBMs and yet, still not giving much information about the quality of the generated samples nor the equilibrium properties of the trained models. In our results below, we will show that without putting on the table this information, the comparison between methods or tuning of parameters, becomes extremely unstable. Finally, we want to mention that it is particularly surprising that the CD recipe (with short k) is still used [39, 41–43]. As shown, RBMs trained with CD are not able to generate proper samples from scratch, and this behavior is true even if doing $k \approx O(100)$ MC steps at each update. The classical explanation for this is that CD is creating local energy well around the datapoint, which is separated by large energy barriers [35], since the dynamics do not have time to relax when k is small.

[4] It has been observed in MNIST that by flipping the input 0 ↔ 1, the usual gradient ascent was much slower in one case than in the other.

In the rest of the paper, we explain how, by characterizing the MT, we are able to explain the capricious behavior of RBMs with respect to the different usual schemes. Furthermore, we will see that our results are totally consistent with what is observed in other families of EBMs, such as ConvNet generative networks [22, 23] or Boltzmann machines [44, 45] when trained using short k negative chains, which suggests that this explanation applies to EBMs in general.

4. Monitoring the learning performance of RBMs

In this work we wish to address the following questions: (i) how can one quantify that an RBM has correctly learned a dataset? (ii) Has it learned the equilibrium model? (iii) When should one stop the learning and the sampling of new configurations? We already argued in section 2 that, alone, the LL of the dataset is not precise enough to address these questions. For this reason, we have considered several additional observables to quantify the generated samples' quality. We introduce shortly below the different metrics used in our analysis, the precise definitions of all of them can be found in the SM. All the observables are computed and averaged using sets of $N_s = 1000$ true or generated samples.

Log-likelihood $\mathcal{L}$: on the contrary to other machine learning methods (such as classification), the LL cannot be exactly evaluated in an RBM without an exhaustive search of the configuration space. Thus, we estimate the partition function using the annealed importance sampling (AIS) method [46]. Details of our implementation can be found in SM. We call $\mathcal{L}^{\mathcal{D}}$ the LL of the (original) dataset with respect to the model, and $\mathcal{L}^{\text{RBM}}$, the LL of the generated samples with respect to the model.

Error of the second moment $\mathcal{E}^{(2)}$: we compute the mean square error (MSE) between the covariance matrix computed with a set of samples generated by the RBM and one of the dataset.

Error in the power spectral density $\mathcal{E}^{\text{PSD}}$: for images, we compute the log average power spectrum function of radial wave number distance for both a subset of the dataset and the generated set, and then compute the MSE between the two functions. In practice, we compute the error (between true and generated samples) on the logarithm of the squared module of the image's discrete 2D-Fourier transform elements A_{kl} at a fixed distance d in Fourier space

$$P(d) = \log\left(\langle \|A_{kl}\|^2 \rangle_{(k,l)|k^2+l^2=d^2}\right). \tag{5}$$

Error in the adversarial accuracy indicator (AAI) $\mathcal{E}^{\text{AAI}}$: the AAI was recently introduced in [47] with the goal of quantifying the resemblance and the 'privacy' of data drawn by a generative model with respect to the training set. This metric is based on the idea of data nearest neighbors. We begin by defining two sets with each one containing N_s samples: (i) a 'target' set containing only generated samples $T \equiv \{\boldsymbol{v}_{\text{RBM}}^{(m)}\}_{m=1}^{N_s}$ and (ii) a 'source' with samples from the dataset $S \equiv \{\boldsymbol{v}_{\mathcal{D}}^{(m)}\}_{m=1}^{N_s}$. Then, for each sample m in groups $\{T, S\}$, we compute the closest sample—using the Euclidean distance—to the set groups $\{T, S\}$. For instance, the closest samples in $\{S\}$ for a sample m in $\{T\}$

would be $d_{TS}(m) = \min_n \left\| \boldsymbol{v}_{\text{RBM}}^{(m)} - \boldsymbol{v}_{\mathcal{D}}^{(n)} \right\|$. We can compute in the same way $d_{ST}(m)$, $d_{SS}(m)$ and $d_{TT}(m)$. Once we have all four of these distances, we can estimate how often the 'nearest neighbor' (the sample at the shortest distance) of each sample is contained in the same starting set, or in the other one. When the generated samples are statistically indistinguishable from real ones, both frequencies should converge to the value of a random guess 0.5 when N_s large. We therefore compute the MSE of these two frequencies with respect to the asymptotic value 0.5, which defines $\mathcal{E}^{\text{AAI}}$. See SM for more details.

Relative entropy $\boldsymbol{\Delta S}$: we approximate the entropy of a given set of data by its byte size when compressed using gzip [48]. In particular, we compute the entropy of a group of N_s random samples of the dataset, $S^{\mathcal{D}}$, and the entropy of another set of data of the same size, called S^{cross}, composed in equal portions by samples of the train and generated set. Finally, we compute the relative size of this cross set with respect to the dataset's one, minus the expected value for identical sets, i.e. $\Delta S = S^{\text{cross}}/S^{\mathcal{D}} - 1$. Then, a large ΔS indicates to us that the generated set is less 'ordered' than the dataset. On the contrary, a small ΔS reflects that the generated samples lack diversity.

In the SM, we show the evolution of other quality images observables along the generation sampling (such as the error in the third moment correlations, the error in the energy, or the Frechet inception distance score [49] for images). All of them show very similar trends to the ones shown here.

J. Stat. Mech. (2022) 114009

5. Results

Equipped with this set of quality measures, we now want to monitor and evaluate the learning process. To this end, we need to precisely control the learning dynamics and their interplay with the equilibration time. To do so, we try to isolate all of the different factors affecting the learning, and for this reason, at the first stage, we avoid using the common tricks to speed up the estimation of the rhs of the gradient of equation (4) during the learning. With this goal, we focus our first experiments on the Rdm-k schemes, and discuss the interplay with standard methods such as CD or PCD later on. The Rdm-k scheme is very practical to study the convergence to equilibrium for two reasons: (i) the MCMC chains are initialized from configurations that are uncorrelated with the dataset, and (ii) the sampling protocol used during the learning is identical to the one used to generate new samples from scratch with the learned RBM. The pipeline of our experiments is summarized in figure 1(B). We split the analysis into two separated stages: learning and generation sampling. During the learning, we update the parameters for $\sim 2 \times 10^5$ times using the gradient estimated following the Rdm-k scheme (with $k = 10, 50, 100, 500, 10^3$ and 10^4). We save the RBM parameters at different numbers of updates equally spaced in the logarithmic scale. The number of updates used to train a particular RBM will be referred to as its *age*, t_{age}. During the generation stage, we sample the equilibrium Gibbs measure of each RBM (of age t_{age}) with a MCMC initialized at random. We evaluate the quality of the samples generated at Gibbs sampling times t_G equally spaced in logarithmic scale. Unless something else is mentioned, we will consider RBMs with $N_\mathrm{h} = 500$ hidden nodes and trained with a fixed learning rate of $\eta = 0.01$, hence the different RBMs will differ only by the value of k and t_age. Further details on the learning parameters can be found in the SM.

Evolution of the quality of the generated samples with the sampling time—

We show in figure 2(A) the evolution of the quality estimators of the generated samples throughout the generation sampling for RBMs of different ages (i.e. following the different color lines in figure 1(B)), and trained with three different k. We recall that perfect samples, in the sense that they are indistinguishable from the dataset, would have a score of 0 in all of the quality observables except for the LL, where $\mathcal{L}^\mathrm{RBM}$ should match $\mathcal{L}^\mathcal{D}$ ($\mathcal{L}^\mathcal{D}$ is marked as a horizontal line). For all k and early learning times (ages below $\lesssim 10^3$ updates in red-yellowish colors), all the metrics describe a constant behavior with t_G at roughly the same values obtained using a completely untrained RBM, and generally far away from 0. For an intermediate range of ages (pale blue colors), we observe an important improvement of the metrics with the RBM's age, up to a certain value (dark blue lines) above which time evolution curves start to collapse. Furthermore, for $k = 10$, only young ages converge to constant curves in t_G, something that extends to intermediate ages for $k = 500$ and covers the majority of the ages for $k = 10^4$. In all the cases, the sampling of the oldest machine ($t_\mathrm{age} \sim$ 200k updates) describes a non-monotonous behavior in the quality measures, with a clear peak of best quality at $t_\mathrm{G} \sim k$. This is also the point where $\mathcal{L}^\mathcal{D}$ and $\mathcal{L}^\mathrm{RBM}$ intersect and where ΔS crosses 0. In figure 2(C), for each learning process using a different k, we plot the position of this $t_\mathrm{G}^\mathrm{best}$ (obtained for each quality metrics) against the value of k used to train the machine. We observe a perfect match between both times. At this point we want to stress that we obtain exactly the same qualitative behavior with the other considered datasets, as shown in the SM. Figure 2(C) reflects that these RBMs 'memorize' the number of MCMC steps used to estimate the gradient during the learning phase. Memory effects are a clear signal of non-equilibrium effects [50] and in fact, this effect is not present as long as k is long enough to equilibrate the system during the learning. Indeed, we see that the larger k, the longer (in terms of number of updates) it takes to observe memory effects (for $k = 10^4$ we only appreciate out-of-equilibrium effects above 100k updates, while they appear around $t_\mathrm{age} \gtrsim 10^3$ updates in the case of $k = 10$). These two regimes (equilibrium and OOE) are also reproduced when using the CD and PCD schemes during the learning. We show in figure 2(B) the evolution of $\mathcal{E}^\mathrm{PSD}$ of the generated samples as function of t_G for three RBMs trained with the three schemes and $k = 10$ steps (higher values of k are shown in the SM). We compare results with MCMC chains initialized either at random (continuous lines) or at the dataset (dashed lines). The sampling of the RBMs trained with CD-10 is also highly complex in time, and much slower than that of the Rdm-10 considering that the values of $\mathcal{E}^\mathrm{PSD}$ measured in runs from random initialization and from the dataset never converge to the same values for large t_age. The situation is rather different for PCD-10, where the error sampling curves describe a very smooth behavior in time and a very slow convergence towards an equilibrium value. Yet according to the minimum values reached in $\mathcal{E}^\mathrm{PSD}$, the equilibrium samples of that model are significantly worse than the best samples created using the Rdm scheme (even working in the OOE regime). This difference is also observed in the rest of quality estimators and at much higher values of k, as shown in figure 5 of the SM. Indeed, for MNIST, one needs to reach PCD-10^3 to start to match the qualities achieved with the Rdm-10 recipe when exploiting the OOE regime.

Quantifying the mixing time: at this point, we try to quantify the MT or relaxation time of the MCMC dynamics of our RBM as a function of t_{age}, to rationalize figure 2. To do so, we compute the time-autocorrelation function, ρ, of the averaged visible variables

$$\rho(t) = \frac{C_m(t)}{C_m(0)}, \quad \text{where} \quad C_m(t) = \frac{1}{N_v}\sum_i (m_i(t) - \bar{m}_i)(m_i(0) - \bar{m}_i) \tag{6}$$

where $m_i(t) = \mathrm{sigmoid}(\sum_a w_{ia}h_a(t) + b_i)$ and $\bar{m}_i$ is the equilibrium magnetization[5]. At this point, let us stress that in a perfectly trained RBM, $\bar{m}_i$ would be equal to $1/N_v \sum_m^M v_i^{(m)}$. Yet, this is not the case in a poorly trained machine. For this reason, we estimate these equilibrium values from a long 10^5 MCMC steps simulation. The value of $\rho(t)$ will be close to 1 if the data at $t = 0$ and t are very similar, and to 0 if they are completely decorrelated. In order to properly measure the MT, we need to compute the autocorrelation function from equilibrium configurations. Therefore, we measure ρ with equation (6) but after discarding the first MCMC $\approx 10^4$ steps[6]. We show the time-correlation curves obtained for our best trained machine (using Rdm and $k = 10^4$) in figure 3(A) for different values of t_{age}. Clearly, the older the machine, the slower the relaxation. Such a study is harder with the poorly trained RBMs because equilibration is achieved at times $t_{\mathrm{G}} > 10^5$ for most of the t_{ages}, being particularly dramatic for Rdm-10 where relaxations are extremely slow (see SM). The decay being typically exponential, we can easily extract the MT (or the exponential relaxation time), t_α, from a fit of $\rho(t)$ to $A\ \exp(-t/t_\alpha)$. We show the dependence of t_α with t_{age} in figure 3(C). This t_α (for the slowest observable) controls the time-scale of the relaxations in the system, the time necessary to sample independent (uncorrelated) configurations, which means that the number of MCMC iterations necessary to reach equilibrium from the beginning of a MCMC simulation is expected to scale with it. Typically, the thermalization time, t_{therm}, is sometimes this t_α [51]. This means that, in order to guarantee a sampling of equilibrium configurations, one needs Gibbs samplings longer than this t_{therm} (which scales with t_α). Note that t_{therm} (the distance to equilibrium) might depend on the MC starting point while t_α should not, yet t_{therm} will grow proportionally with t_α for any learning scheme. In fact, we can independently estimate the thermalisation time for the Rdm-k runs, t_{therm}, by comparing results from two independent simulations with different initializations: random and dataset, and wait until they converge to the same equilibrium expectation value. This means that one can estimate t_{therm} (for each t_{age})

[5] The time-autocorrelation function of the visible, hidden or averaged hidden units are qualitatively the same.

[6] In practice, we discard as many MCMC steps as needed to stop detecting dependence on the $t = 0$ point.

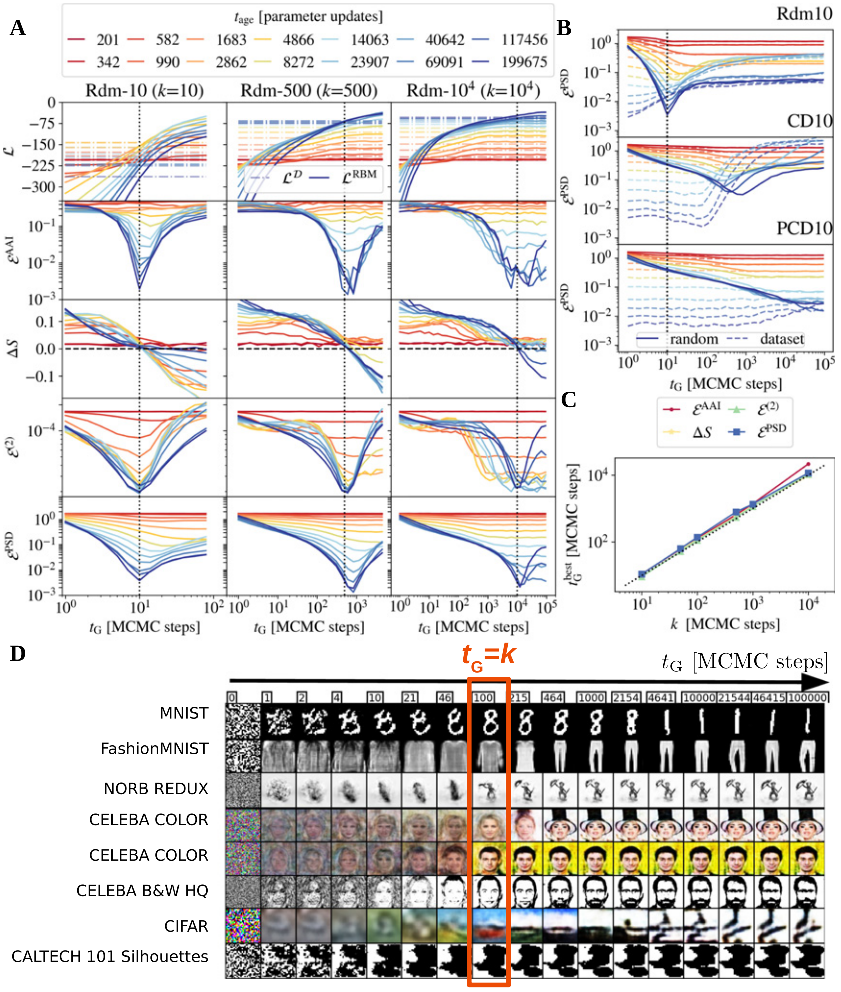

Figure 2. Evolution of the quality of the generated samples during the data generation sampling (along the color dashed lines of figure 1(B)). In (A) we show the LL and the rest of quality estimators (introduced in section 2) as function of the sampling time, $t_{\rm G}$, for chains initialized at random and using RBMs trained for an increasing number of updates $t_{\rm age}$ (age is coded by colors) using the Rdm-k scheme with $k = 10, 500, 10^4$. In general, the closer $\mathcal{E}$ and ΔS are to zero, the more similar the generated and original datasets are. For the LL, we show the value computed with the generated set ($\mathcal{L}^{\rm RBM}$, in continuous growing lines) and with the original dataset ($\mathcal{L}^{\mathcal{D}}$, in dashed horizontal lines). We observe that these two measures cross at roughly the same $t_{\rm G}$ at which all $\mathcal{E}$ reach a minimum and ΔS crosses zero. (B) We show the evolution of $\mathcal{E}^{\rm PSD}$ for three RBMs trained using the Rdm-10, CD-10 and PCD-10 recipes, for the same ages as in (A). We consider two possible initializations of the Markov chain (random in full lines and dataset in dashed lines). (C) We plot the position of the best performance peak and the time at which $\Delta S = 0$, as function of the number of MCMC steps k used for estimating the negative term of the gradient during the training. (D) We show an example of the images generated as function of $t_{\rm G}$, using RBMs trained with Rdm-100 and with different datasets. The sampling time $t_{\rm G}$ that matches k is highlighted in an orange frame.

https://doi.org/10.1088/1742-5468/ac98a7

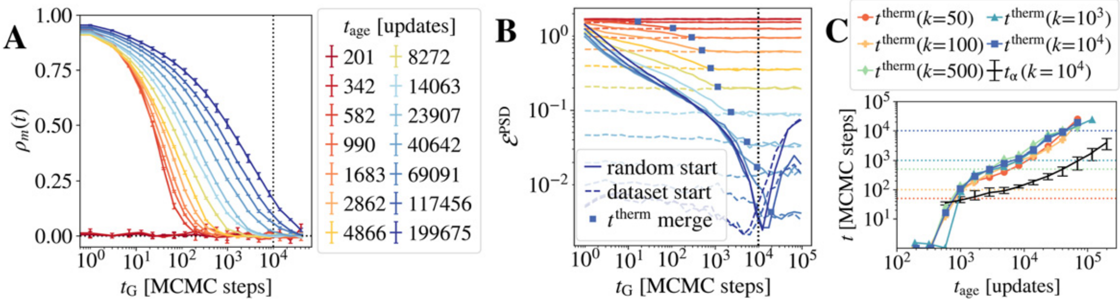

Figure 3. (A) The time-autocorrelation function of the visible mean magnetizations, ρ_m (see equation (6)), computed during the sampling of an RBM trained with Rdm-10^4 for different number of updates $t_{\rm age}$. This function carries the information about the number of MCMC steps that are needed (in equilibrium) to forget an initial configuration of the MC chain. Clearly, the time needed to decorrelate grows with the age of the RBM. In (B), for the same RBMs, we show the evolution of the generated samples' quality through the $\mathcal{E}^{\rm PSD}$ observable, as a function of the sampling time $t_{\rm G}$, and for two different initializations: from the dataset and from random. We mark the times $t^{\rm therm}$ beyond which both runs converge to the same constant value. (C) We compare the mixing times, t_α, obtained from a fit of the curves $\rho_m(t)$ in (A) to the form $\exp(-t/t_\alpha)$, with the convergence-to-equilibrium times $t^{\rm therm}$ obtained in (B) for RBMs trained with different values of k. We see that the equilibration time $t^{\rm therm}$ follows the same trend for various values of k.

by the number of MCMC steps needed to collapse the measures of the two trajectories to a constant value. See for instance figure 3(B), where we show the evolution of $\mathcal{E}^{\rm PSD}$ with $t_{\rm G}$ for the two runs. The merging point for each $t_{\rm age}$ is highlighted with a square. We show the $t_{\rm therm}$ values we obtain with this procedure in figure 3(C) for the different values of k.

As a result, these two times, t_α and $t_{\rm therm}$, provide us with a lower and a safe upper bound (for instance, when initializing at random) for the minimum number of MCMC steps (for the learning's Gibbs sampling), k, needed to learn a good equilibrium RBM model. As shown in figure 3(C), these times grow progressively with $t_{\rm age}$, and in particular, always falling out of the range of iterations used to train the $k = 10$ runs (once the RBM starts to learn the dataset for $t_{\rm age} \gtrsim 10^3$ updates), and falling out the range of scheme Rdm-500 for $t_{\rm age} \gtrsim 10^4$ and for the last two $t_{\rm age}$ for the Rdm-10^4 scheme (i.e. for $t_{\rm age} \gtrsim 10^5$ updates). In other words, at those ages, the learning enters in the OOE regime and the RBM starts to learn the dynamical process rather than the equilibrium properties. In fact, these ages mark the crossover between smooth and non-monotonic evolution of the observables in figure 2(A). This reinforces our previous claim that the apparition of the 'best performance peak' in the error estimators is a purely non-equilibrium effect. A similar comment can be made for the LL. See for instance figure 2(A) first row, for each $t_{\rm age}$, where we compare the LL measured on the generated $\mathcal{L}^{\rm RBM}$ and original dataset $\mathcal{L}^{\mathcal{D}}$ as function of its generation sampling time $t_{\rm G}$. We distinguish, again, two distinct behaviours associated with the equilibrium or OOE regime. In the first case, the LL of the generated samples converges to the value of the dataset and remains there

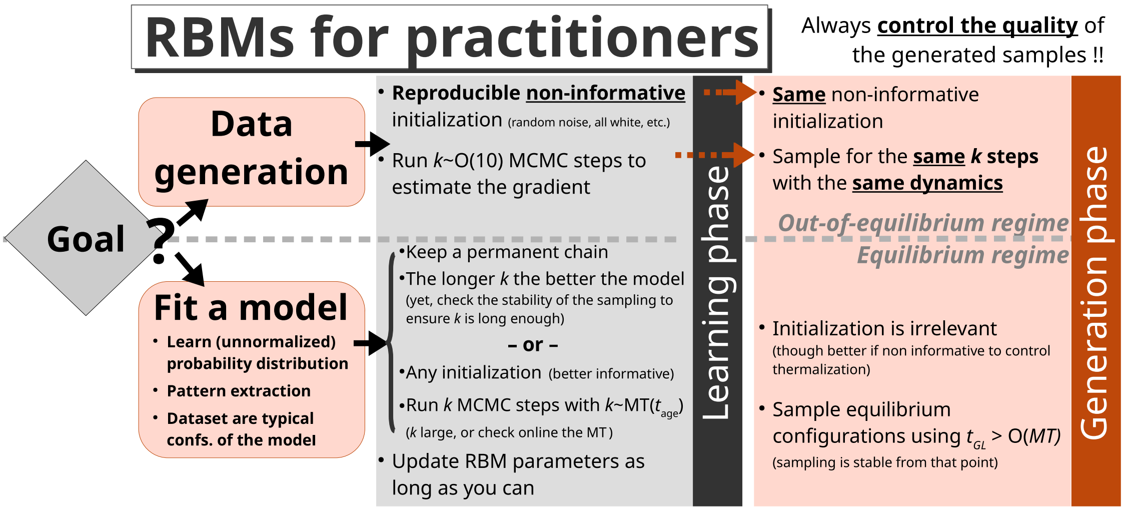
RBMs for practitioners
Always control the quality of the generated samples !!
Goal ?
Data generation
Fit a model
• Learn (unnormalized) probability distribution
• Pattern extraction
• Dataset are typical confs. of the model
• Reproducible non-informative initialization (random noise, all white, etc.)
• Run k~O(10) MCMC steps to estimate the gradient
•Keep a permanent chain
•The longer k the better the model (yet, check the stability of the sampling to ensure k is long enough)
– or –
•Any initialization (better informative)
•Run k MCMC steps with k~MT(t_age) (k large, or check online the MT)
• Update RBM parameters as long as you can
Learning phase
• Same non-informative initialization
• Sample for the same k steps with the same dynamics
Out-of-equilibrium regime
Equilibrium regime
• Initialization is irrelevant (though better if non informative to control thermalization)
• Sample equilibrium configurations using t_GL > O(MT) (sampling is stable from that point)
Generation phase

Figure 4. Flow chart of the algorithm proposed to properly train and sample an RBM depending on the final goal of the training.

even at large t_G. Whereas, in the OOE regime, $\mathcal{L}^{\mathrm{RBM}}$ crosses $\mathcal{L}^{\mathcal{D}}$ at $t_G \approx k$ and continues growing afterwards. This highlights the problem of using the LL to evaluate the sample generation power. While the fact that the likelihood increases during the learning is in general a good sign, it is not necessarily linked with the equilibrium properties of the model. In fact, the observation that $\mathcal{L}^{\mathrm{RBM}}$ tends to reach much higher values than $\mathcal{L}^{\mathcal{D}}$ should be rather interpreted as a sign of poor learning, as shown in figure 1.

As a direct consequence of the results discussed above, we can see that even in the case where sampling is openly OOE, such as with the Rdm-10 scheme, one can generate fairly good samples as long as one limits the generation sampling to $t_G \sim k$, as shown in figure 1 for $k = 10$: digits are reproduced with uniform probability, and they look OK on visual inspection. For longer samplings we start to get clear unbalanced distributions. It is also quite remarkable that when matching t_G with k in the OOE regime of Rdm schemes, we get better quality samples than with the standard CD o PCD schemes even at very long sampling times, as we can see from the comparison of $\mathcal{E}^{\mathrm{PSD}}$ in figure 2(B). This is also true for other quality observables or larger values of k as shown around figure 5 of the SM. In particular, for NMIST, the OOE recipe always generates better samples than the equilibrium regime reached with PCD trained with $k < 10^3$ steps.

6. Conclusions

In this work, we show that the training of an RBM is richer and more complex than what was previously thought. We show that the choice of the number of MCMC steps, k, used to estimate the gradient during the learning has a dramatic impact on the learned model's dynamics and equilibrium states. In practice, we find two regimes: (i) the OOE, when k is much lower than the MCMC mixing time, the trained RBM model encodes the dynamical process followed to train it, resulting in a highly non-linear, unstable and slow evolution of the quality of the generated samples with the sampling time, and a convergence towards poor and strongly biased samples. And (ii) the equilibrium regime, when k largely overpasses the mixing time, the marginal equilibrium probability distribution of the trained model (over the visible layer) matches that of the dataset. In such a case, the quality of the generated samples improves smoothly in time converging faster to the equilibrium values, converging much faster than in the OOE regime. While regime (ii) is needed to obtain good trained models, we show that regime (i) is particularly well suited for sample generation because the data sampling nonlinearities can be controlled in the Rdm-k approach, and exploited to generate high-quality data in short training times. We further show that the MC mixing time increases significantly as the learning progresses, always being much larger than the standard ks used in the literature. We summarize the best strategy to train and exploit RBMs depending on their purpose in figure 4.

The existence of a crucial competing mechanism in the training process had already been noted and exploited in EBMs with more complex energy functions (those defined with deep convolutional neural networks) [22, 23], and after our results, both regimes were also found in the simpler brother of the RBMs, the Boltzmann machines [44, 45]. This evidences that the OOE regime, which was widely unnoticed in the literature for decades, is a general consequence of the MCMC based likelihood maximization methods, so it is expected to be reproduced in many other popular models like deep Boltzmann machines. Interestingly, our results should simplify their training and use, at least in the OOE regime. More generally, this work provides a set of key points to be checked when working with EBMs, and a new fundamentally distinctive attribute to these generative models: equilibrium or OOE. We leave systematic comparisons between the models trained in each of these regimes, and in particular, the effect on the features for future works. For instance, we observed that RBMs trained with low k suffer strong aging effects extremely similar to those observed on the spin glasses [52]. Effects which are not present in equilibrium RBMs.

Acknowledgments

A D and B S were supported by the Comunidad de Madrid and the Complutense University of Madrid (Spain) through the Atracción de Talento programs (Refs. 2019-T1/TIC-13298 for A D and 2019-T1/TIC-12776 for B S), the Banco Santander and the UCM (Grant PR44/21-29937), and Ministerio de Economía y Competitividad, Agencia Estatal de Investigación and Fondo Europeo de Desarrollo Regional (FEDER) (Spain and European Union) through the Grants PID2021-125506NA-I00 and PGC2018-094684-B-C21.

References

[1] Smolensky P 1986 Information processing in dynamical systems: foundations of harmony theory *Technical Report* Colorado University at Boulder Dept of Computer Science

[2] Le Roux N and Bengio Y 2008 Representational power of restricted Boltzmann machines and deep belief networks *Neural Comput.* **20** 1631–49

[3] Hjelm R D, Calhoun V D, Salakhutdinov R, Allen E A, Adali T and Plis S M 2014 Restricted Boltzmann machines for neuroimaging: an application in identifying intrinsic networks *NeuroImage* **96** 245–60

[4] Hu X, Huang H, Peng B, Han J, Liu N, Lv J, Guo L, Guo C and Liu T 2018 Latent source mining in FMRI via restricted Boltzmann machine *Hum. Brain. Mapp.* **39** 2368–80

[5] Tubiana J, Cocco S and Monasson R 2019 Learning protein constitutive motifs from sequence data *Elife* **8** e39397

[6] Goodfellow I, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, Courville A and Bengio Y 2014 Generative adversarial nets *Advances in Neural Information Processing Systems* pp 2672–80

[7] Fischer A and Igel C 2014 Training restricted Boltzmann machines: an introduction *Pattern Recognit.* **47** 25–39

[8] Salakhutdinov R and Murray I 2008 On the quantitative analysis of deep belief networks *Proc. 25th Int. Conf. on Machine Learning* pp 872–9

[9] Melchior J, Fischer A and Wiskott L 2016 How to center deep Boltzmann machines *J. Mach. Learn. Res.* **17** 3387–447

[10] Gabrié M, Tramel E W and Krzakala F 2015 Training restricted Boltzmann machine via the Thouless–Anderson–Palmer free energy *Advances in Neural Information Processing Systems* pp 640–8

[11] de Boves Harrington P 2018 Feature expansion by a continuous restricted Boltzmann machine for near-infrared spectrometric calibration *Anal. Chim. Acta* **1010** 20–8

[12] Shen H and Li H 2019 A gradient approximation algorithm based weight momentum for restricted Boltzmann machine *Neurocomputing* **361** 40–9

[13] Savitha R, Ambikapathi ArulM and Rajaraman K 2020 Online RBM: growing restricted Boltzmann machine on the fly for unsupervised representation *Appl. Soft Comput.* **92** 106278

[14] Montúfar G 2016 Restricted Boltzmann machines: introduction and review *Information Geometry and its Applications IV* (Berlin: Springer) pp 75–115

[15] Chen J, Cheng S, Xie H, Wang L and Xiang T 2018 Equivalence of restricted Boltzmann machines and tensor network states *Phys. Rev.* B **97** 085104

[16] Decelle A, Fissore G and Furtlehner C 2018 Thermodynamics of restricted Boltzmann machines and related learning dynamics *J. Stat. Phys.* **172** 1576–608

[17] Shimagaki K and Weigt M 2019 Selection of sequence motifs and generative Hopfield–Potts models for protein families *Phys. Rev.* E **100** 032128

[18] Melko R G, Carleo G, Carrasquilla J and Ignacio Cirac J 2019 Restricted Boltzmann machines in quantum physics *Nat. Phys.* **15** 887–92

[19] Harsh M, Tubiana J, Cocco S and Monasson R 2020 'Place-cell' emergence and learning of invariant data with restricted Boltzmann machines: breaking and dynamical restoration of continuous symmetries in the weight space *J. Phys. A: Math. Theor.* **53** 174002

[20] Yelmen B, Decelle A, Ongaro L, Marnetto D, Tallec C, Montinaro F, Furtlehner C, Pagani L and Jay F 2021 Creating artificial human genomes using generative neural networks *PLoS Genet.* **17** e1009303

[21] Bravi B, Tubiana J, Cocco S, Monasson R, Mora T and Walczak A M 2021 RBM-MHC: a semi-supervised machine-learning method for sample-specific prediction of antigen presentation by HLA-I alleles *Cell Syst.* **12** 195–202

[22] Nijkamp E, Hill M, Han T, Zhu S-C and Wu Y N 2020 On the anatomy of MCMC-based maximum likelihood learning of energy-based models *Proc. AAAI Conf. on Artificial Intelligence* vol 34 pp 5272–80

[23] Nijkamp E, Hill M, Zhu S-C and Wu Y N 2019 Learning non-convergent non-persistent short-run MCMC toward energy-based model *Advances in Neural Information Processing Systems* vol 32 pp 5232–42

[24] LeCun Y, Bottou L, Bengio Y and Haffner P 1998 Gradient-based learning applied to document recognition *Proc. IEEE* **86** 2278–324

[25] Xiao H, Rasul K and Vollgraf R 2017 Fashion-MNIST: a novel image dataset for benchmarking machine learning algorithms (arXiv:1708.07747)

[26] LeCun Y, Huang F J and Leon B 2004 Learning methods for generic object recognition with invariance to pose and lighting *Proc. 2004 IEEE Computer Society Conf. on Computer Vision and Pattern Recognition, 2004. CVPR 2004* (Piscataway, NJ: IEEE) vol 2 pp II–104

[27] Colonna V, Ayub Q, Chen Y, Pagani L, Luisi P, Pybus M, Garrison E, Xue Y and Tyler-Smith C 2014 Human genomic regions with exceptionally high levels of population differentiation identified from 911 whole-genome sequences *Genome Biol.* **15** 1–14

[28] Karras T, Aila T, Laine S and Lehtinen J 2017 Progressive growing of GANs for improved quality, stability, and variation (arXiv:1710.10196)

[29] Krizhevsky A *et al* 2009 Learning multiple layers of features from tiny images *Technical Report* TR-2009 University of Toronto

[30] Nair V and Hinton G E 2010 Rectified linear units improve restricted Boltzmann machines *Proc. 27th Int. Conf. on Machine Learning (ICML-10)* pp 807–14

[31] Hinton G E 2002 Training products of experts by minimizing contrastive divergence *Neural Comput.* **14** 1771–800

[32] Tieleman T 2008 Training restricted Boltzmann machines using approximations to the likelihood gradient *Proc. 25th Int. Conf. on Machine Learning* pp 1064–71

[33] Hukushima K and Nemoto K 1996 Exchange Monte Carlo method and application to spin glass simulations *J. Phys. Soc. Japan* **65** 1604–8

[34] Salakhutdinov R R 2009 Learning in Markov random fields using tempered transitions *Advances in Neural Information Processing Systems* ed Y Bengio, D Schuurmans, J Lafferty, C Williams and A Culotta vol 22 (New York: Curran Associates)

[35] Desjardins G, Courville A, Bengio Y, Vincent P and Delalleau O 2010 Tempered Markov chain Monte Carlo for training of restricted Boltzmann machines *Proc. 13th Int. Conf. on Artificial Intelligence and Statistics JMLR Workshop and Conf. Proc.* pp 145–52

[36] Burda Y, Grosse R and Salakhutdinov R 2015 Accurate and conservative estimates of MRF log-likelihood using reverse annealing *Artificial Intelligence and Statistics* (PMLR) pp 102–10

[37] Grosse R and Salakhudinov R 2015 Scaling up natural gradient by sparsely factorizing the inverse Fisher matrix *Int. Conf. on Machine Learning* (PMLR) pp 2304–13

[38] Krause O, Fischer A and Igel C 2018 Population-contrastive-divergence: does consistency help with RBM training? *Pattern Recognit. Lett.* **102** 1–7

[39] Romero E, Mazzanti F, Delgado J and Buchaca D 2019 Weighted contrastive divergence *Neural Netw.* **114** 147–56

[40] Upadhya V and Sastry P S 2019 Efficient learning of restricted Boltzmann machines using covariance estimates *Asian Conf. on Machine Learning* (PMLR) pp 836–51

[41] Aoki K-I and Kobayashi T 2016 Restricted Boltzmann machines for the long range Ising models *Mod. Phys. Lett.* B **30** 1650401

[42] Morningstar A and Melko R G 2017 Deep learning the Ising model near criticality (arXiv:1708.04622)

[43] Yevick D and Melko R 2021 The accuracy of restricted Boltzmann machine models of Ising systems *Comput. Phys. Commun.* **258** 107518

[44] Barrat-Charlaix P, Muntoni A P, Shimagaki K, Weigt M and Zamponi F 2021 Sparse generative modeling via parameter reduction of Boltzmann machines: application to protein-sequence families *Phys. Rev.* E **104** 024407

[45] Muntoni A P, Pagnani A, Weigt M and Zamponi F 2021 adabmdca: adaptive Boltzmann machine learning for biological sequences

[46] Krause O, Fischer A and Igel C 2020 Algorithms for estimating the partition function of restricted Boltzmann machines *Artif. Intell.* **278** 103195

[47] Yale A, Dash S, Dutta R, Guyon I, Pavao A and Bennett K P 2020 Generation and evaluation of privacy preserving synthetic health data *Neurocomputing* **416** 244–55

[48] Baronchelli A, Caglioti E and Loreto V 2005 Measuring complexity with zippers *Eur. J. Phys.* **26** S69

[49] Heusel M, Ramsauer H, Unterthiner T, Nessler B and Hochreiter S 2017 GANs trained by a two time-scale update rule converge to a local Nash equilibrium *Advances in Neural Information Processing Systems* vol 30

[50] Keim N C, Paulsen J D, Zeravcic Z, Sastry S and Nagel S R 2019 Memory formation in matter *Rev. Mod. Phys.* **91** 035002

[51] Sokal A 1997 Monte Carlo methods in statistical mechanics: foundations and new algorithms *Functional Integration* (Berlin: Springer) pp 131–92

[52] Mézard M, Parisi G and Virasoro M A 1987 *Spin Glass Theory and Beyond: An Introduction to the Replica Method and Its Applications* vol 9 (Singapore: World Scientific)