Training Energy-Based Models with Parallel Trajectory Tempering

Anonymous Author(s)
Affiliation
Address
email

Abstract

We introduce a novel training protocol for energy-based models that accelerates the equilibration of Markov chains used in maximum-likelihood training, enabling stable and accurate learning on highly clustered, multimodal datasets. The method extends Trajectory Parallel Tempering, inspired by parallel tempering and Hamiltonian exchange Monte Carlo, by dynamically exchanging model parameters with earlier stages, faster-mixing stages to enhance exploration. A reservoir-based strategy reuses equilibrium samples from previous models, reducing memory costs and achieving speeds comparable to Persistent Contrastive Divergence when combined with optimized gradient schedulers such as Nesterov Accelerated Gradient. Experiments on clustered datasets show consistently higher test log-likelihoods and markedly improved sample quality in Restricted Boltzmann Machines compared to standard methods.

1 Introduction

Energy-based models (EBMs) offer a powerful framework for modeling complex systems by defining probability distributions through an energy function, rather than an explicit likelihood or a dynamical denoising process. This formulation provides the flexibility to capture intricate dependencies in high-dimensional data without restrictive parametric assumptions, making EBMs particularly well suited for scientific modeling. Their expressive power is especially evident in simple and interpretable architectures such as Restricted Boltzmann Machines (RBMs). These models have been widely applied in computational biology di Sarra et al. [2025], neuroscience van der Plas et al. [2023], statistical physics Tubiana and Monasson [2017], Decelle and Furtlehner [2021b] and quantum physics Melko et al. [2019], where they successfully capture long-range and multibody correlations Yelmen et al. [2023], hierarchical organization Decelle et al. [2023], and collective interactions Decelle et al. [2024, 2025]. By learning directly from data, EBMs uncover hidden organizational principles and yield valuable insights into the mechanisms governing complex systems.

Training energy-based models (EBMs) is computationally demanding because accurate learning requires equilibrated Markov Chain Monte Carlo (MCMC) sampling to ensure that the dataset distribution is correctly encoded in the Boltzmann weight Decelle et al. [2021], Agoritsas et al. [2023]. Insufficient equilibration leads to distorted representations, poor generalization, and unstable dynamics, especially in high-dimensional, multimodal datasets where slow mixing induces mode collapse and memory effects. To make training practical, Hinton's contrastive divergence (CD) Hinton [2002] approximates equilibrium sampling through short MCMC runs initialized from data samples. Although widely used, CD produces models with poor equilibrium properties Salakhutdinov and Murray [2008], Desjardins et al. [2010], Decelle et al. [2021]. Persistent contrastive divergence (PCD) Tieleman [2008] improves stability by evolving persistent chains, yet fails on clustered data where chains drift from equilibrium Béreux et al. [2023]. More advanced approaches—such

Submitted to 39th Conference on Neural Information Processing Systems (NeurIPS 2025). Do not distribute.

as constrained MCMC Béreux et al. [2023], population annealing Krause et al. [2018], or non-equilibrium reweighting Carbone et al. [2024]—better capture multimodal structure but remain computationally expensive or inefficient for highly structured datasets Béreux et al. [2025].

Optimized MCMC schemes such as Parallel Tempering (PT) [Hukushima and Nemoto, 1996] have improved EBM training [Salakhutdinov, 2009, Desjardins et al., 2010] but remain too costly for practical use and become inefficient on clustered data due to first-order phase transitions along the temperature ladder [Decelle and Furtlehner, 2021a, Béreux et al., 2025]. Stacked Tempering, which trains progressively smaller RBMs using the previous hidden layer as input, accelerates sampling [Fernandez-de Cossio-Diaz et al., 2024] but is impractical since it requires simultaneous training of multiple networks. Recent theoretical analyses revealed cascades of second-order phase transitions during learning [Bachtis et al., 2024], motivating tempering strategies that avoid first-order transitions. Parallel Trajectory Tempering (PTT) implements this idea by exchanging model parameters along the training trajectory via a Metropolis criterion, akin to Hamiltonian Exchange Monte Carlo [Rosta et al., 2011], and achieves remarkable sampling speedups for clustered models [Béreux et al., 2025].

In this work, we show that PTT can be effectively used to train higher-quality models. To make it practical, we introduce a strategy that eliminates the need to simulate all models in the trajectory ladder simultaneously—avoiding prohibitive memory and computational costs—while enabling larger learning rates. This approach integrates PTT into training with no additional overhead compared to standard CD or PCD, making it suitable for real-world applications. The PTT algorithm and training procedure are detailed in Section 2, results in Section 3, and implementation aspects in Appendices B–B.2.

2 Parallel Trajectory Tempering

The PTT algorithm was recently introduced in Béreux et al. [2025] as a sampling method that dramatically accelerates RBM sample generation by exploiting the smooth evolution of the model's distribution during training Bachtis et al. [2024]. Unlike traditional Parallel Tempering, which employs a temperature ladder for replica exchange, PTT uses a set of model parameters saved along the training trajectory, making it analogous to the Hamiltonian Exchange Monte Carlo method Rosta et al. [2011] but using a training trajectory. Exchanges are thus proposed between replicas at two neighboring training steps, t and $t-1$, in the model ladder, and accepted with probability:

$$p_{\mathrm{acc}}(\boldsymbol{x}_t \leftrightarrow \boldsymbol{x}_{t-1}) = \min\left[1, \exp\left(\Delta\mathcal{H}_t(\boldsymbol{x}_t) - \Delta\mathcal{H}_t(\boldsymbol{x}_{t-1})\right)\right] \text{ with } \Delta\mathcal{H}_t(\boldsymbol{x}) \equiv \mathcal{H}_t(\boldsymbol{x}) - \mathcal{H}_{t-1}(\boldsymbol{x}). \quad (1)$$

We now describe how this algorithm can be used to efficiently compute the gradient online during training. At the start of EBM training, only a single model $\mathcal{H}_0$ is available. To initialize PTT, we keep a frozen copy of this initial model and propose configuration swaps between it and the evolving model. As training proceeds, the distribution of the trained model gradually diverges from that of the frozen one, leading to a decrease in the swap acceptance rate α. Whenever α falls below a threshold $\alpha^\star$, a frozen copy $\mathcal{H}_1$ of the current model is inserted between the last frozen model and the updated one. The process continues by sampling from the set $\{\mathcal{H}_t\}$ and proposing swaps with the model being trained, while new frozen models are added according to the acceptance-rate criterion. The procedure for adding models is detailed in Appendix B.2.

However, adding new models during training increases both computational and memory costs as the process advances. To address this, we introduce a *reservoir* sampling strategy that prevents the need to simulate a large number of models simultaneously. Whenever a new model is added to the replica set, a large collection of independent equilibrium samples (the *reservoir*) is generated for the antepenultimate model. Subsequently, during sampling, only the models following the one associated with the reservoir are simulated, while independent moves are proposed by uniformly drawing configurations from the reservoir. This reduces the computational cost of full PTT sampling to the few instances when new models are added to the ladder, thereby substantially lowering the overhead while preserving the accuracy of the sampling process.

An important advantage of the PTT algorithm is its ability to reliably compute the log-likelihood of the replicas online during training. This is achieved through the formal exact relation between the partition functions of successive models along the trajectory, $Z_{t+1} = \left\langle e^{\mathcal{H}_t - \mathcal{H}_{t+1}} \right\rangle_{\mathcal{H}_t} Z_t$, where $\langle\cdot\rangle_{\mathcal{H}_t}$ denotes the Boltzmann average with respect to model $\mathcal{H}_t$. This implies that the evolution of the log-partition function Z_t for $t = 1, \ldots, T$ can be readily estimated from the samples generated during training. The

log-partition function at the final step T is then given by $\log Z_T = \sum_{t=0}^{T-1} \log \langle \exp(\mathcal{H}_t - \mathcal{H}_{t+1}) \rangle_{\mathcal{H}_t} + \log Z_0$, where the average is estimated using the equilibrium configurations obtained with PTT. This can also be used a posteriori to compute the log-likelihood on models saved during training.

Finally, having access to more accurate gradient estimates enables the use of more advanced update rules than standard gradient ascent. To illustrate this, we compare simple gradient ascent with the Nesterov Accelerated Gradient (NAG) method, described in Appendix B.1. It is worth noting that this strategy typically performs poorly in standard training protocols such as PCD, where it often slows down or even halts training before convergence to well-trained models.

3 Results

To evaluate the method, we consider three different datasets: (i) a binarized version of the black-and-white MNIST dataset Deng [2012], consisting of 50,000 handwritten digit images (0–9); (ii) the Human Genome Dataset (HGD) Colonna et al. [2014], Consortium et al. [2015], which encodes whether each of 805 selected genes is mutated (1) or not (0) with respect to a reference individual; and (iii) the BKACE protein family (PFAM ID:PF05853), comprising 16,323 homologous amino acid sequences aligned on 272 sites. The HGD and BKACE datasets exhibit strong clustering, as revealed by their principal component projections. The first two datasets are binary, while the last one is categorical with 21 possible states. Using these three datasets, we perform three types of training: PTT with NAG, PCD with NAG, and PCD with a fixed learning rate γ, and compare their performance in terms of both model quality and wall-clock training time. Results are shown for MNIST Fig. 1, for HGD in Fig. 2 (more analysis in the appendix Fig. 5), and for BKACE in Fig. 3. Model quality is assessed through the evolution of the test log-likelihood as a function of training time (measured either in parameter updates or wall-clock time), when this quantity can be computed via PTT sampling estimates. This excludes the PCD runs for the BKACE proteins, where we are unable to construct a viable model ladder with non-vanishing swap acceptance to sample the configurations, because the training breaks quite early. We further evaluate the generated samples by comparing their first and second moments to those of the training data, and by projecting both datasets onto their first principal components to compare histograms along each direction, thereby assessing potential mode-collapse effects.

In Fig. 1, we show that for the non-clustered MNIST dataset, PTT with NAG $\gamma = 0.1$ slightly surpasses the performance of PCD-100 with a fixed $\gamma = 0.1$ in both log-likelihood and sampling quality, while requiring the same wall-clock training time. This demonstrates the reliability of the method. The improvement becomes much more dramatic when we consider clustered datasets. For HGD (see Fig. 2), PCD-100 is unable to properly reproduce all the clusters in the dataset even with $\alpha = 0.01$ while the PTT does is perfectly. The limitations are even more clear when one compares the first and two moments of the distributions in Fig 5 at the appendix, showing that PCD runs are unable to even reproduce properly the frequencies. Log-likelihoods are consistently higher for the PTT training.

Finally, we evaluate our method on a protein dataset (BKACE). This dataset exhibits a highly clustered distribution, posing significant challenges for PCD. In Fig. 3, we compare the projections of generated samples obtained for both PCD-RBM and PTT-RBM onto the first principal components of the dataset. PCD fails to adequately capture the full distribution, with certain regions of the dataset being underrepresented and spurious modes emerging. On the other side, samples generated after PTT training closely align with the empirical distribution of the dataset, demonstrating the method's superior ability to model complex, multimodal data structures.

4 Conclusion

In this work, we address the long-standing challenge of poor mixing across modes in a class of EBMs. By leveraging Parallel Trajectory Tempering (PTT) during training, we achieve a drastic reduction in mixing times, enabling the use of larger learning rates. Combined with efficient strategies—such as a sample reservoir and an adaptive learning-rate scheduler—our approach outperforms previous methods like PCD and makes it possible to train RBMs on highly clustered datasets with many well-separated modes that were previously inaccessible.

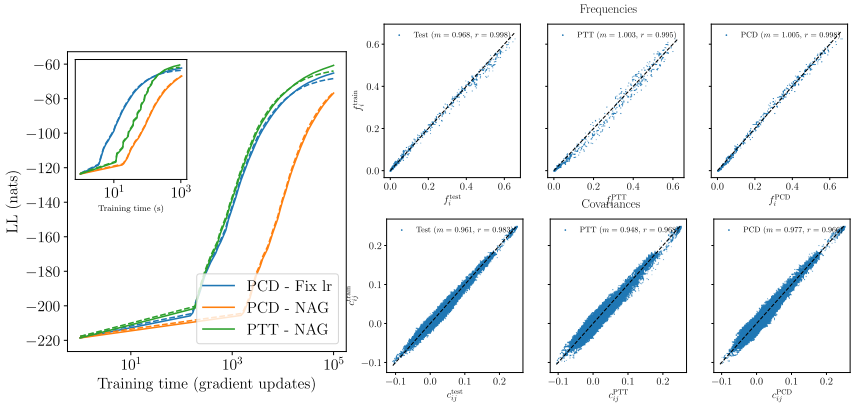

Figure 1: **Comparison of training methods on MNIST.** The left panel shows the train and test log-likelihood as a function of gradient updates and wall-clock time. The right panel compares the statistics of generated data against the training data: Top row shows the empirical pixel averages, and the bottom row shows the covariances. The first column corresponds to the test dataset, the second column to samples generated by PTT-RBM, and the third column to samples generated by PCD-RBM. For both methods, statistics are computed using the last saved model after training.

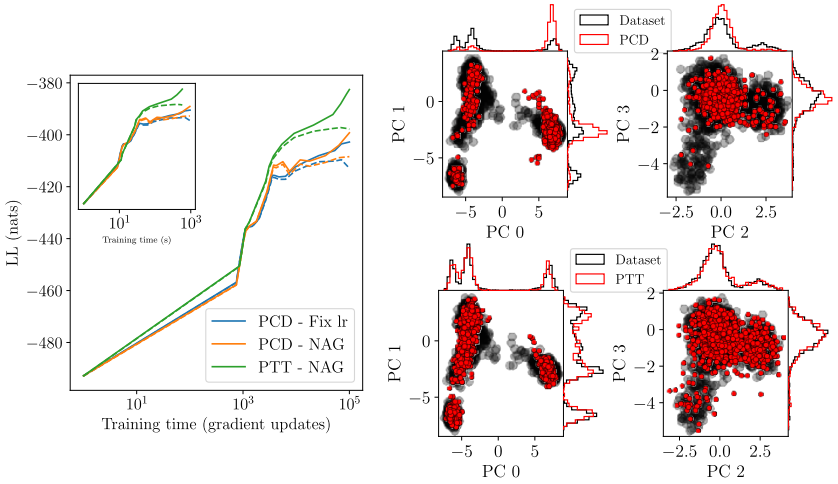

Figure 2: **Comparison of training methods on the HGD dataset**. The left panel shows the train and test log-likelihood as a function of gradient updates and wall-clock time. The right panel compares the distribution of generated samples after PCD training (top row) and PTT training (bottom row).

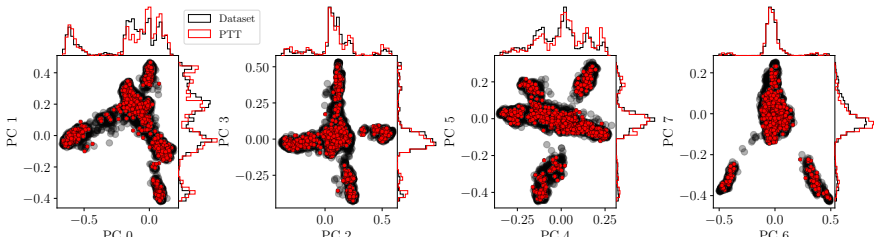

Figure 3: **Distribution of the generated samples on BKACE**. We plot projection of both the samples generated by the PTT trained RBM and the dataset. The generative samples match very well the data distribution at the contrary of PCD trained RBMs, see Fig 6.

References

Elisabeth Agoritsas, Giovanni Catania, Aurélien Decelle, and Beatriz Seoane. Explaining the effects of non-convergent sampling in the training of energy-based models. arXiv preprint arXiv:2301.09428, 2023.

Yves F Atchadé, Gareth O Roberts, and Jeffrey S Rosenthal. Towards optimal scaling of metropolis-coupled markov chain monte carlo. Statistics and Computing, 21(4):555–568, 2011.

Dimitrios Bachtis, Giulio Biroli, Aurélien Decelle, and Beatriz Seoane. Cascade of phase transitions in the training of energy-based models. NeurIPS (2024), arXiv:2405.14689, 2024.

Nicolas Béreux, Aurélien Decelle, Cyril Furtlehner, and Beatriz Seoane. Learning a restricted boltzmann machine using biased monte carlo sampling. SciPost Physics, 14(3):032, 2023.

Nicolas Béreux, Aurélien Decelle, Cyril Furtlehner, Lorenzo Rosset, and Beatriz Seoane. Fast training and sampling of restricted boltzmann machines. In 13th International Conference on Learning Representations-ICLR 2025, 2025.

Davide Carbone, Mengjian Hua, Simon Coste, and Eric Vanden-Eijnden. Efficient training of energy-based models using jarzynski equality. Advances in Neural Information Processing Systems, 36, 2024.

Vincenza Colonna, Qasim Ayub, Yuan Chen, Luca Pagani, Pierre Luisi, Marc Pybus, Erik Garrison, Yali Xue, Chris Tyler-Smith, and 1000 Genomes Project Consortium. Human genomic regions with exceptionally high levels of population differentiation identified from 911 whole-genome sequences. Genome biology, 15(6):R88, 2014.

1000 Genomes Project Consortium et al. A global reference for human genetic variation. Nature, 526(7571):68, 2015.

Aurélien Decelle and Cyril Furtlehner. Exact training of restricted boltzmann machines on intrinsically low dimensional data. Physical Review Letters, 127(15):158303, 2021a.

Aurélien Decelle and Cyril Furtlehner. Restricted boltzmann machine: Recent advances and mean-field theory. Chinese Physics B, 30(4):040202, 2021b.

Aurélien Decelle, Cyril Furtlehner, and Beatriz Seoane. Equilibrium and non-equilibrium regimes in the learning of restricted boltzmann machines. Advances in Neural Information Processing Systems, 34:5345–5359, 2021.

Aurélien Decelle, Beatriz Seoane, and Lorenzo Rosset. Unsupervised hierarchical clustering using the learning dynamics of restricted boltzmann machines. Phys. Rev. E, 108:014110, Jul 2023. doi: 10.1103/PhysRevE.108.014110. URL `https://link.aps.org/doi/10.1103/PhysRevE.108.014110`.

Aurélien Decelle, Cyril Furtlehner, Alfonso de Jesús Navas Gómez, and Beatriz Seoane. Inferring effective couplings with restricted boltzmann machines. SciPost Physics, 16(4):095, 2024.

Aurélien Decelle, Alfonso de Jesús Navas Gómez, and Beatriz Seoane. Inferring high-order couplings with neural networks. arXiv preprint arXiv:2501.06108, 2025.

Li Deng. The mnist database of handwritten digit images for machine learning research [best of the web]. IEEE signal processing magazine, 29(6):141–142, 2012.

Guillaume Desjardins, Aaron Courville, Yoshua Bengio, Pascal Vincent, and Olivier Delalleau. Tempered markov chain monte carlo for training of restricted boltzmann machines. In Proceedings of the thirteenth international conference on artificial intelligence and statistics, pages 145–152. JMLR Workshop and Conference Proceedings, 2010.

Giovanni di Sarra, Barbara Bravi, and Yasser Roudi. The unbearable lightness of restricted boltzmann machines: Theoretical insights and biological applications. Europhysics Letters, 149(2):21002, 2025.

Jorge Fernandez-de Cossio-Diaz, Clément Roussel, Simona Cocco, and Remi Monasson. Accelerated sampling with stacked restricted boltzmann machines. In The Twelfth International Conference on Learning Representations, 2024.

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

Koji Hukushima and Koji Nemoto. Exchange monte carlo method and application to spin glass simulations. Journal of the Physical Society of Japan, 65(6):1604–1608, 1996.

Oswin Krause, Asja Fischer, and Christian Igel. Population-contrastive-divergence: Does consistency help with rbm training? Pattern Recognition Letters, 102:1–7, 2018.

Roger G Melko, Giuseppe Carleo, Juan Carrasquilla, and J Ignacio Cirac. Restricted boltzmann machines in quantum physics. Nature Physics, 15(9):887–892, 2019.

Erik Nijkamp, Mitch Hill, Song-Chun Zhu, and Ying Nian Wu. Learning non-convergent non-persistent short-run mcmc toward energy-based model. Advances in Neural Information Processing Systems, 32, 2019.

Erik Nijkamp, Mitch Hill, Tian Han, Song-Chun Zhu, and Ying Nian Wu. On the anatomy of mcmc-based maximum likelihood learning of energy-based models. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 34, pages 5272–5280, 2020.

Edina Rosta, Marcin Nowotny, Wei Yang, and Gerhard Hummer. Catalytic mechanism of rna backbone cleavage by ribonuclease h from quantum mechanics/molecular mechanics simulations. Journal of the American Chemical Society, 133(23):8934–8941, 2011.

Ruslan Salakhutdinov and Iain Murray. On the quantitative analysis of deep belief networks. In Proceedings of the 25th international conference on Machine learning, pages 872–879, 2008.

Russ R Salakhutdinov. Learning in markov random fields using tempered transitions. Advances in neural information processing systems, 22, 2009.

Tijmen Tieleman. Training restricted boltzmann machines using approximations to the likelihood gradient. In Proceedings of the 25th international conference on Machine learning, pages 1064–1071, 2008.

Jérôme Tubiana and Rémi Monasson. Emergence of compositional representations in restricted boltzmann machines. Physical review letters, 118(13):138301, 2017.

Thijs L van der Plas, Jérôme Tubiana, Guillaume Le Goc, Geoffrey Migault, Michael Kunst, Herwig Baier, Volker Bormuth, Bernhard Englitz, and Georges Debrégeas. Neural assemblies uncovered by generative modeling explain whole-brain activity statistics and reflect structural connectivity. Elife, 12:e83139, 2023.

Burak Yelmen, Aurélien Decelle, Leila Lea Boulos, Antoine Szatkownik, Cyril Furtlehner, Guillaume Charpiat, and Flora Jay. Deep convolutional and conditional neural networks for large-scale genomic data generation. PLOS Computational Biology, 19(10):e1011584, 2023.

A Appendix

B The Restricted Boltzmann Machine and the PCD training

The RBM consists of $N_{\rm v}$ visible and $N_{\rm h}$ hidden nodes, both represented by binary variables $\{0,1\}$. The two layers interact through a weight matrix $\boldsymbol{W}$ without intra-layer couplings, and each unit is influenced by local biases $\boldsymbol{\theta}$ (visible) and $\boldsymbol{\eta}$ (hidden). The joint Gibbs–Boltzmann distribution reads

$$p(\boldsymbol{v},\boldsymbol{h}) = \frac{1}{Z}\exp[-\mathcal{H}(\boldsymbol{v},\boldsymbol{h})], \qquad \mathcal{H}(\boldsymbol{v},\boldsymbol{h}) = -\sum_{ia} v_i W_{ia} h_a - \sum_i \theta_i v_i - \sum_a \eta_a h_a, \qquad (2)$$

where Z is the partition function. As in other models with hidden variables, training aims to minimize the discrepancy between the empirical data distribution $p_{\mathcal{D}}(\boldsymbol{v}) = M^{-1}\sum_{m=1}^{M}\delta(\boldsymbol{v}-\boldsymbol{v}^{(m)})$ and the model's marginal distribution $p(\boldsymbol{v}) = \sum_{\boldsymbol{h}}\exp[-\mathcal{H}(\boldsymbol{v},\boldsymbol{h})]/Z = \exp[-H(\boldsymbol{v})]/Z$. The model is trained by maximizing the log-likelihood $\mathcal{L} = \langle -H(\boldsymbol{v})\rangle_{\mathcal{D}} - \log Z$, using standard stochastic gradient ascent. The corresponding gradients are

$$\frac{\partial\mathcal{L}}{\partial W_{ia}} = \langle v_i h_a\rangle_{\mathcal{D}} - \langle v_i h_a\rangle_{\mathrm{RBM}}, \quad \frac{\partial\mathcal{L}}{\partial\theta_i} = \langle v_i\rangle_{\mathcal{D}} - \langle v_i\rangle_{\mathrm{RBM}}, \quad \frac{\partial\mathcal{L}}{\partial\eta_a} = \langle h_a\rangle_{\mathcal{D}} - \langle h_a\rangle_{\mathrm{RBM}}, \quad (3)$$

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 the average over the dataset, and $\langle f(\boldsymbol{v},\boldsymbol{h})\rangle_{\mathrm{RBM}}$ the model expectation with respect to the model's $p(\boldsymbol{v},\boldsymbol{h})$.

Since Z is intractable, the model averages in the gradient are typically estimated using N_{s} independent MCMC processes, commonly referred to as *parallel chains*. Observable averages $\langle o(\boldsymbol{v},\boldsymbol{h})\rangle_{\mathrm{RBM}}$ are then approximated by the empirical mean

$$\frac{1}{R}\sum_{r=1}^{R} o(\boldsymbol{v}^{(r)},\boldsymbol{h}^{(r)}),$$

@articlecolonna2014human, title=Human genomic regions with exceptionally high levels of population differentiation identified from 911 whole-genome sequences, author=Colonna, Vincenza and Ayub, Qasim and Chen, Yuan and Pagani, Luca and Luisi, Pierre and Pybus, Marc and Garrison, Erik and Xue, Yali and Tyler-Smith, Chris and 1000 Genomes Project Consortium, journal=Genome biology, volume=15, number=6, pages=R88, year=2014, publisher=Springer

where $(\boldsymbol{v}^{(r)},\boldsymbol{h}^{(r)})$ denotes the final configuration reached by each of the R chains. Reliable gradient estimates require the Markov chains to equilibrate before each parameter update, but achieving convergence at every step is computationally unfeasible. Recent studies have shown that using non-convergent MCMC to estimate the gradient introduces strong memory effects in the trained models Nijkamp et al. [2020], Agoritsas et al. [2023]: the model no longer encodes the dataset distribution in its Boltzmann distribution, but instead reproduces it through a dynamical process that can only be controlled under carefully designed training protocols Decelle et al. [2021], Nijkamp et al. [2019].

To mitigate out-of-equilibrium effects, it is common to keep R permanent (or persistent) chains, meaning that the final configurations obtained from the MCMC process used to estimate the gradient at update t, $\boldsymbol{P}_t \equiv \{(\boldsymbol{v}_t^{(r)},\boldsymbol{h}_t^{(r)})\}_{r=1}^{R}$, are used to initialize the chains at the subsequent update $t+1$. This algorithm is known as PCD Tieleman [2008]. In this framework, training can be viewed as a slow cooling process in which, instead of varying a single parameter (e.g., temperature), the entire set of model parameters $\boldsymbol{\Theta}_t = (\boldsymbol{W}_t,\boldsymbol{\theta}_t,\boldsymbol{\eta}_t)$ is updated at each step according to $\boldsymbol{\Theta}_{t+1} = \boldsymbol{\Theta}_t + \gamma\boldsymbol{\nabla}\mathcal{L}_t$, where $\boldsymbol{\nabla}\mathcal{L}_t$ is the gradient in Eq. equation 3 estimated using the configurations in $\boldsymbol{P}_t$, and γ is the learning rate.

B.1 Nesterov Accelerated Gradient

The Nesterov Accelerated Gradient ascent is a gradient ascent with an additional momentum term. The update is decomposed in two stages. First a regular SGD step:

$$\phi_{t+1} = \theta_t + \eta\nabla_\theta\mathcal{L}(\theta_t) \quad (4)$$

and then a momentum step:

$$\theta_{t+1} = \phi_{t+1} + \mu(\phi_{t+1} - \phi_t) \quad (5)$$

In practice we rely on the PyTorch implementation of this optimizer.

B.2 Adding a new model to the set of replicas

Adding a new frozen model in the PTT scheme results in an acceptance rate close to one with the current model, since the updated model will need some time to diverge from the distribution of the last frozen model. In Parallel Tempering algorithms, high acceptance rate hinder the performance as it leads to a drop in expected squared jumping distance (ESJD) Atchadé et al. [2011]. The drop in ESJD means that the chains used to compute the gradient of the EBM are prevented to travel to higher

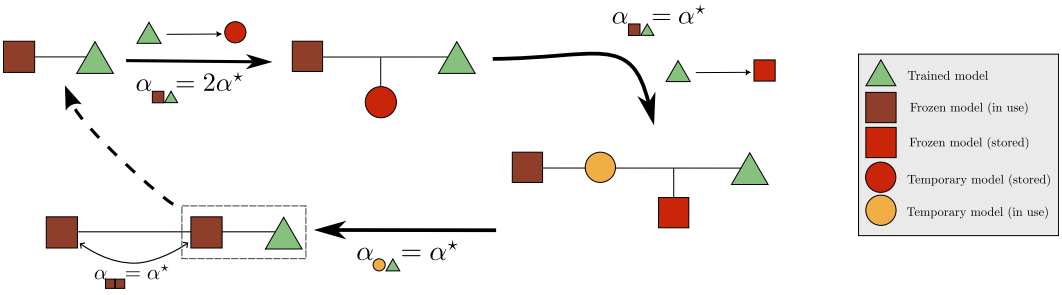

Figure 4: This panel illustrates the procedure to add a new model to the PTT scheme during training, as explained in Appendix B.2.

temperatures and mix between clusters, leading to a degradation in the estimation of the gradient. To circumvent this issue, we rely on temporary models that act as intermediate points until the updated model sufficiently diverges from the frozen weights as illustrated in Fig. 4.

Let $\mathcal{H}_t$ denote the most recent frozen model and $\mathcal{H}_{\text{curr}}$ the updated model. Let $\alpha(k, l)$ represent the acceptance rate between models k and l, and $\alpha^\star$ the target acceptance rate. The adaptive freezing process is defined as follows:

- **Temporary Model Creation:** When $\alpha(\mathcal{H}_t, \mathcal{H}_{\text{curr}}) \leq 2\alpha^\star$, a temporary frozen copy $\mathcal{H}_{\text{temp}}$ is created from $\mathcal{H}_{\text{curr}}$ and stored outside the sampling scheme.
- **Permanent Model Freezing:** When $\alpha(\mathcal{H}_t, \mathcal{H}_{\text{curr}}) \leq \alpha^\star$, a new frozen copy $\mathcal{H}_{t+1}$ is created from $\mathcal{H}_{\text{curr}}$ and stored outside the sampling scheme. Simultaneously, $\mathcal{H}_{\text{temp}}$ is introduced into the sampling scheme between $\mathcal{H}_t$ and $\mathcal{H}_{\text{curr}}$.
- **Temporary Model Replacement:** When $\alpha(\mathcal{H}_{\text{temp}}, \mathcal{H}_{\text{curr}}) \leq \alpha^\star$, $\mathcal{H}_{\text{temp}}$ is replaced by $\mathcal{H}_{t+1}$ in the sampling process. The procedure then repeats between $\mathcal{H}_{t+1}$ and $\mathcal{H}_{\text{curr}}$.

B.3 Hyperparameters

The hyperparameters used for the trainings are given in Table 1.

Table 1: Hyperparameters used for the training of RBMs.

Name	Batch size	#Chains	#Update	lr	#MCMC steps	Increment	#Hidden nodes	L2
HGD								
PCD	1000	1000	100 000	0.01	10	N/A	100	0
PTT - NAG	1000	1000	100 000	0.01	2	5	100	0
MNIST								
PCD - Fix lr	1000	1000	100 000	0.1	100	N/A	500	0
PCD - NAG	1000	1000	100 000	0.01	100	N/A	500	0
PTT - NAG	1000	1000	100 000	0.01	2	5	500	0
BKACE								
PCD - Fix lr	1000	1000	100 000	0.01	10	N/A	1000	0.001
PTT - NAG	1000	1000	100 000	0.01	2	5	1000	0.001

B.4 Additional plots

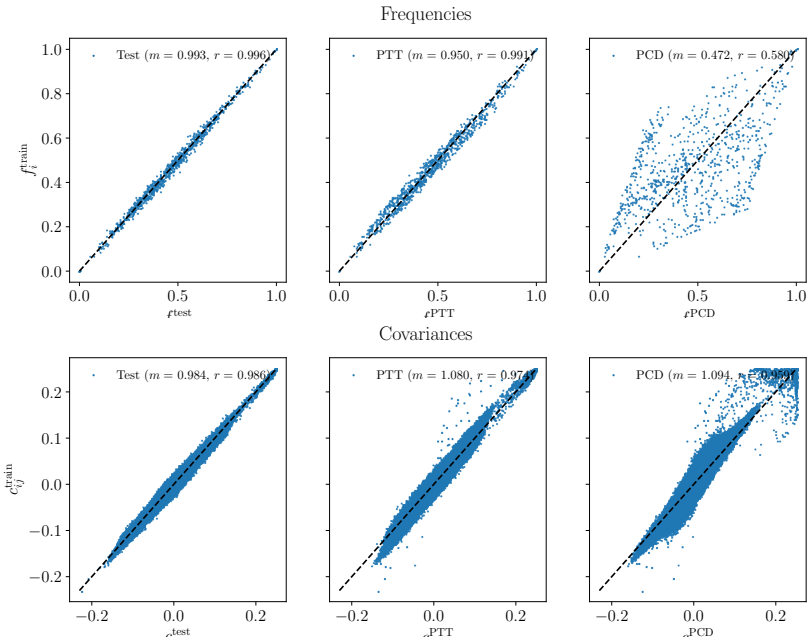

Figure 5: **Comparison of the generative accuracy of the PTT-RBM and PCD-RBM on the HGD dataset**. The left column compares the test dataset with the train dataset, the middle one the PTT-RBM with the train dataset and the right one the PCD-RBM with the train dataset. The top row shows pixel means, and the bottom row shows 2-body correlations.

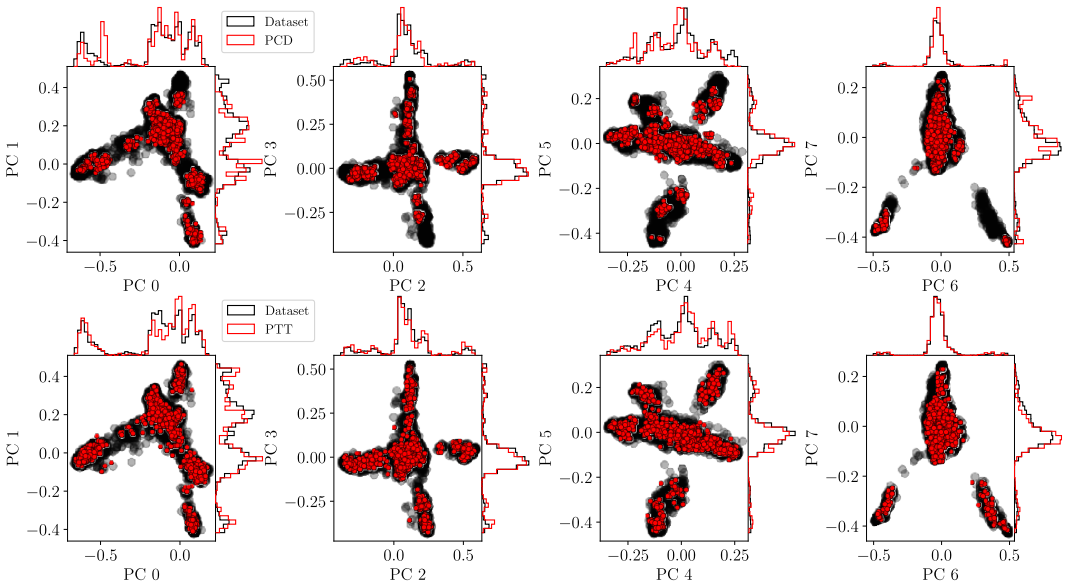

Figure 6: **Distribution of the generated samples on BKACE**. We show both the generated samples from the RBMs trained with PTT and with PCD. The samples generated by the PCD trained RBM is clearly not matching well the empirical data distribuion.