PROCEEDINGS OF SCIENCE

Improvement of Heatbath Algorithm in LFT using Generative models

Ali Faraz,[e] **Ankur Singha,**[a,b,*] **Dipankar Chakrabarti,**[c] **Shinichi Nakajima**[a,b,c] **and Vipul Arora**[e]

[a] *Probabilistic Modeling and Inference Group, BIFOLD, Germany*
[b] *Machine Learning Group, Technische Universität Berlin, Germany*
[c] *Department of Physics, Indian Institute of Technology Kanpur, India*
[d] *RIKEN Center for AIP, Japan*
[e] *Department of Electrical Engineering, Indian institute of Technology Kanpur, India*

E-mail: a.singha@tu-berlin.de, vipular@iitk.ac.in

The Heatbath Algorithm is commonly used for sampling in local lattice field theories, but performing exact updates or sampling from the local density is challenging when dealing with continuous variables. Heatbath methods rely on rejection-based sampling at each site, which can suffer from low acceptance rates if the proposal distribution is not optimally chosen—a non-trivial task. In this work, we propose a novel, straightforward approach for generating proposals at each lattice site for the ϕ^4 and XY models using generative AI models. This method learns a conditional local distribution, without requiring training samples from the target, conditioned on both neighboring sites and action parameter values.

The 41st International Symposium on Lattice Field Theory (LATTICE2024)
28 July-3 August 2024
Liverpool, UK

*Speaker

© Copyright owned by the author(s) under the terms of the Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License (CC BY-NC-ND 4.0).

PoS(LATTICE2024)036

1. Introduction

Lattice models in physics describe systems on a discrete grid, where configurations follow a Boltzmann distribution characterized by a Hamiltonian $H(\phi, \lambda)$ or action $s(\phi, \lambda)$. The statistical properties of these lattices vary with parameters λ, exhibiting significant changes near specific values of λ known as critical regions, where phenomena like critical slowing down occur. In these regions, traditional sampling methods like MCMC, while providing convergence guarantees, become inefficient due to high autocorrelation times. Specialized algorithms, such as Swendsen-Wang [1], Wolff [2, 3] address this but have limitations, particularly for continuous symmetries. The Hamiltonian Monte Carlo (HMC) [4], which makes global MCMC updates, is considered the state of the art for lattice QCD simulations. Recently, generative machine learning (ML) methods, including normalizing flows [5] and diffusion models [6] have shown promise in efficiently sampling lattice field theories, with successful applications in ϕ^4 [7–14] and gauge theory [15–25]. These generative methods however aim to model the joint distribution of the entire lattice, which poses a significant scalability challenge for learning-based approaches. In lattice community a well known approach for sampling local lattice theories is the Heat Bath algorithm, which factorizes the lattice distribution, allowing each lattice site to be sampled conditioned on its local neighboring sites. This approach is particularly efficient for discrete systems like the Ising model, where only two possible states exist. However, in continuous systems, this requires problem specific designs [26, 27] and often requiring rejection-based methods. The difficulty of finding effective proposal distributions for a generic continuous systems can lead to high rejection rates in Heatbath, increasing the simulation cost. Moreover, at different regime of the action parameter, one may need to fine tune the proposal distribution.

We present the Parallelizable Block Metropolis-within-Gibbs (PBMG) method for sampling in local lattice models, offering an efficient proposal distribution for rejection-based Heatbath algorithms. PBMG learns the local single-site distribution conditioned on neighboring sites and the action parameter values. Once trained, the model can be used to sample the entire lattice as in the Heatbath method. The target distribution is a simple one-dimensional distribution, thus significantly improves the learning efficiency. Notably, PBMG does not require any training samples from the target distribution. With a well-trained model, we can reduce the rejection rate commonly encountered in Heatbath sampling. The PBMG operates within a Metropolis-within-Gibbs framework, using ML model-generated proposals conditioned on neighboring sites as well as action parameters. Thus PBMG are conditional generative models, e.g., conditional GMMs and conditional NFs.

To validate the proposed approach, we apply it to 2-D lattices, namely, the XY model from statistical Physics and the scalar ϕ^4 model from Lattice Field Theory.

2. Heatbath as Parallelizable Block Metropolis-within-Gibbs (PBMG)

In this section, we introduce the Heat Bath methods within a generative model framework, which correspond to a Metropolis-within-Gibbs approach, using clear notation.

Consider an N-dimensional probability distribution $p(\phi_1, \phi_2, \ldots, \phi_N)$. For a lattice, N is the number of lattice sites and ϕ_i is the random variable at site i. We partition these sites into G

PoS(LATTICE2024)036

partitions such that the distribution of a site i in a partition g, conditioned on all sites $j \notin g$, is independent of all sites $i' \in g \setminus i$.

For implementing MCMC in general state-spaces, one requires to construct a Markov chain transition kernel $p(\phi_i|\phi_{j\neq i})$ that keeps the target distribution $p(\phi_1, \phi_2, \ldots, \phi_N)$ invariant, and is ergodic for this distribution. Such kernels can also be combined via composition. Keeping this in mind, let K_i be a transition kernel that updates the site $i \in g$, keeping all other sites of the lattice the same. Then the combined kernel that changes all the sites in the partition g is

$$K_g = \prod_{i \in g} K_i \tag{1}$$

and the overall kernel for updating all the sites in a lattice is

$$K = \prod_{g} K_g \,. \tag{2}$$

The advantage of partitioning is that all the sites in the same partition can be sampled simultaneously, thereby making the process faster. Moreover, each kernel K_i need not be conditioned on all the sites outside the partition g, but only a small number of sites in a local neighbourhood of the site i.

Every site-kernel K_i for each $i \in g$ and for every partition g is a Metropolis-within-Gibbs kernel, which means that each site-kernel K_i is a Gibbs kernel with a Metropolis-Hastings accept-reject step.

Let $\boldsymbol{\phi}_{-g}$ denote the set of random variables at all the lattice sites excluding the ones in the partition g and $\boldsymbol{\psi}$ denote given lattice parameters (e.g., temperature, coupling parameters). Let us define $q(\phi_i^{(t)}|\boldsymbol{\phi}_{-g}, \boldsymbol{\psi}; \boldsymbol{\theta})$ as the parametric proposal distribution, parameterized by $\boldsymbol{\theta}$ and conditioned on $\boldsymbol{\phi}_{-g}$ and $\boldsymbol{\psi}$. Then, the acceptance probability α_{K_i} for any site-kernel $K_i \; \forall \; i \in g$ is

$$\alpha_{K_i} = \frac{p(\phi_i^{(t+1)}|\boldsymbol{\phi}_{-g}, \boldsymbol{\psi})}{p(\phi_i^{(t)}|\boldsymbol{\phi}_{-g}, \boldsymbol{\psi})} \cdot \frac{q(\phi_i^{(t)}|\phi_i^{(t+1)}, \boldsymbol{\phi}_{-g}, \boldsymbol{\psi}; \boldsymbol{\theta})}{q(\phi_i^{(t+1)}|\phi_i^{(t)}, \boldsymbol{\phi}_{-g}, \boldsymbol{\psi}; \boldsymbol{\theta})} \tag{3}$$

We design a proposal such that,

$$q(\phi_i^{(t+1)}|\phi_i^{(t)}, \boldsymbol{\phi}_{-g}, \boldsymbol{\psi}; \boldsymbol{\theta}) = q(\phi_i^{(t+1)}|\boldsymbol{\phi}_{-g}, \boldsymbol{\psi}; \boldsymbol{\theta}) \tag{4}$$

The maximum acceptance rate possible i.e., an acceptance rate equal to 1 will be achieved when $q(\phi_i^{(t)}|\boldsymbol{\phi}_{-g}, \boldsymbol{\psi}; \boldsymbol{\theta})$ is exactly the same as $p(\phi_i^{(t)}|\boldsymbol{\phi}_{-g}, \boldsymbol{\psi})$. This reduces our goal to design (or learn) a proposal that could sample from the true conditional distribution as closely as possible. In order to achieve this goal, we use methods like Normalizing Flows and Gaussian Mixture Models in generative machine learning. In the next two sections, we apply the PBMG method to the XY model and the Scalar ϕ^4 theory in 2D.

3. Application to the XY Model

The local Hamiltonian of the the XY model for (i, j)th component $\phi_{i,j}$ of the lattice vector $\boldsymbol{\phi}$ is

$$\begin{aligned} H(\phi_{i,j}) = -\big[&\cos(\phi_{i,j} - \phi_{i+1,j}) + \cos(\phi_{i,j} - \phi_{i,j+1}) \\ &+ \cos(\phi_{i,j} - \phi_{i-1,j}) + \cos(\phi_{i,j} - \phi_{i,j-1}) \big] \end{aligned} \tag{5}$$

PoS(LATTICE2024)036

We see that the Hamiltonian of the (i, j)th component depends only on the components of the four nearest neighbours denoted by $n(i, j) = \{(i + 1, j), (i, j + 1), (i - 1, j), (i, j - 1)\}$. Therefore the conditional distribution of $\phi_{i,j}$ given the four nearest neighbour components and temperature is

$$p\left(\phi_{i,j}|\{\phi_{l,m} : (l, m) \in n(i, j)\}, T\right) = p(\phi_{i,j}|\mathbf{v}_{i,j}) \propto e^{-\frac{H(\phi_{i,j})}{T}} \tag{6}$$

The above conditional distribution is our target distribution. Here, $\mathbf{v}_{i,j} = (\phi_{i+1,j}, \phi_{i,j+1}, \phi_{i-1,j}, \phi_{i,j-1}, T)$ is the 5x1 condition vector corresponding to the site (i, j) which consists of the four nearest neighbour components and the temperature. For this model, we have divided the lattice into two partitions g_0 and g_1.

$$g_k = \{(i, j) : (i + j)\%2 = k\}; k = 0, 1 \tag{7}$$

We use Normalizing Flows to model the proposal distribution $q(\phi_{i,j}|\mathbf{v}_{i,j}; \boldsymbol{\theta})$. Using the change of variables formula,

$$q(\phi_{i,j}|\mathbf{v}_{i,j}; \boldsymbol{\theta}) = p_Z\left(f^{-1}(\phi_{i,j}; \boldsymbol{\theta}_R)|\mathbf{v}_{i,j}; \boldsymbol{\theta}_B\right)\left|\det\left(\frac{\partial f^{-1}(\phi_{i,j}; \boldsymbol{\theta}_R)}{\partial \phi_{i,j}}\right)\right| \tag{8}$$

where, $p_Z(z|\mathbf{v}_{i,j}; \boldsymbol{\theta}_B)$ is the base distribution and $f(z; \boldsymbol{\theta}_R)$ is the invertible transformation used in the Normalizing Flow. Here, $\boldsymbol{\theta} = \{\boldsymbol{\theta_B}, \boldsymbol{\theta_R}\}$. We use Rational Quadratic Splines (RQS) as the transform f.

The loss function used in the training procedure is the expected value of the KL divergence between the proposal $q(\phi_{i,j}|\mathbf{v}_{i,j}; \boldsymbol{\theta})$ and the target $p(\phi_{i,j}|\mathbf{v}_{i,j})$. The Monte Carlo approximation can be used to estimate the KL divergence as follows

$$\mathcal{L} \approx \frac{1}{n} \cdot \frac{1}{N} \sum_{r=1}^{n} \sum_{k=1}^{N} [\log p_Z(z_k|(\mathbf{v}_{i,j})_r; \boldsymbol{\theta}_B) + \log|\det J_f(z_k|(\mathbf{v}_{i,j})_r; \boldsymbol{\theta}_R)|^{-1} - \log p\left(f(z_k; \boldsymbol{\theta}_R)|(\mathbf{v}_{i,j})_r\right)] \tag{9}$$

For training this model we generate training data $\mathbf{v}_{i,j}$ from $p_v(\mathbf{v}_{i,j})$ i.e. $\mathbf{v}_{i,j}$ from $p_v(\mathbf{v}_{i,j}) = \text{Unif}([0, 2\pi]^4 \times [T_1, T_2])$, where $T_2 - T_1$ is the training range for temperatures. For further details on the proposal model, model architecture and the training/inference process for XY model, please refer to Appendix 6.

4. Application to the ϕ^4 Theory

The local action for ϕ^4 theory for lattice site (i, j) can be written as

$$S_{loc}(\phi_{i,j}, \lambda, m^2, \{\phi_{l,m} : (l, m) \in n(i, j)\}) = S_{loc}(\phi_{i,j}, \lambda, m^2, \kappa_{i,j}) \tag{10}$$

$$= \left(m^2 + 4\right)\phi_{i,j}^2 + \lambda\phi_{i,j}^4 - 2\phi_{i,j}\kappa_{i,j} \tag{11}$$

where $\kappa_{i,j} = \phi_{i+1,j} + \phi_{i,j+1} + \phi_{i-1,j} + \phi_{i,j-1}$.

The conditional distribution of the lattice site (i, j) can be written as

$$p\left(\phi_{i,j}|\lambda, m^2, \kappa_{i,j}\right) = p\left(\phi_{i,j}|\mathbf{v}_{i,j}\right) \propto e^{-S_{loc}(\phi_{i,j}, \mathbf{v}_{i,j})} \tag{12}$$

PoS(LATTICE2024)036

where, $\mathbf{v}_{i,j} = (\lambda, m^2, \kappa_{i,j})$ is the condition vector for the distribution. For ϕ^4 theory as well, we have divided the lattice into the same two partitions g_0 and g_1, where

$$g_k = \{(i, j) : (i + j)\%2 = k\}, k = 0, 1 \tag{13}$$

We construct the proposal distribution for the scalar ϕ^4 theory by using a Gaussian Mixture Model with six Gaussian components. The proposal distribution parameterized by $\boldsymbol{\theta}$, can be written as

$$q(\phi_{i,j}|\mathbf{v}_{i,j};\boldsymbol{\theta}) = \sum_{k=1}^{6} \pi_k(\mathbf{v}_{i,j};\boldsymbol{\theta}_k)\mathcal{N}(\phi_{i,j}|\mu_k(\mathbf{v}_{i,j};\boldsymbol{\theta}_k), \sigma_k(\mathbf{v}_{i,j};\boldsymbol{\theta}_k)) \tag{14}$$

where μ_k, σ_k π_k are the mean, standard deviation and mixing coefficients of the k^{th} Gaussian distribution, and $\boldsymbol{\theta} = \{\boldsymbol{\theta}_k\}_{k=1}^{6}$.

The training procedure of PBMG-ϕ^4 is similar to that of PBMG-XY. The loss function used in the training procedure is the expected value of the KL divergence between the proposal $q(\phi_{i,j}|\mathbf{v}_{i,j};\boldsymbol{\theta})$ and the target $p(\phi_{i,j}|\mathbf{v}_{i,j})$

$$\mathcal{L} \approx \frac{1}{n}\sum_{r=1}^{n}\left[\frac{1}{N}\sum_{k=1}^{N}\left[\log q((\phi_{i,j})_k|(\mathbf{v}_{i,j})_r;\boldsymbol{\theta}) - \log p\big((\phi_{i,j})_k|(\mathbf{v}_{i,j})_r\big)\right] + \|\boldsymbol{\pi}\left((\mathbf{v}_{i,j})_r;\boldsymbol{\theta}\right)\|\right] \tag{15}$$

For further details on the architecture and the training/inference process for ϕ^4 theory, please refer to Appendix 6.

5. Results for PBMG

In this section, we assess the performance of the PBMG model in comparison to the heatbath method for both the ϕ^4 and XY models in 2D. For the ϕ^4 model, we use the standard heatbath method, where samples are drawn from a Gaussian distribution, followed by a rejection step to account for the ϕ^4 interaction terms. In the case of the XY model, we use a uniform distribution as the proposal for the rejection step. We compute and compare the acceptance rates between the PBMG and heatbath algorithms. Note that the acceptance rate with rejection sampling differs from that in Metropolis-within-Gibbs sampling. To establish equivalency, we define the acceptance rate for Heatbath algorithm in the following manner.

If $n_i^{(k)}$ represents the number of trials needed to obtain one successful update at site i for the kth configuration in the ensemble, then the total number of attempts A for the entire configuration is given by

$$A_k = \sum_{i=1}^{N} n_i^{(k)}$$

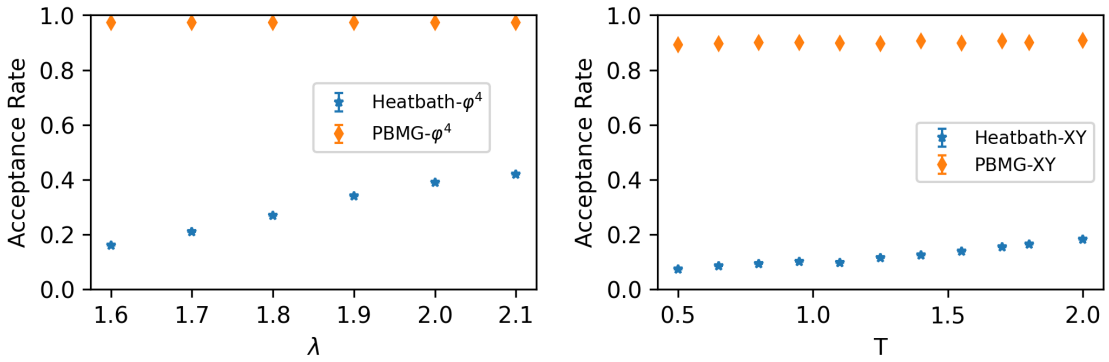
where N is the total number of lattice sites. The total number of successful updates for a configurations is equal to N.

We define the acceptance rate R for the kth configuration as: $R_k = \frac{\text{success updates}}{\text{total attempts}} = \frac{N}{A_k}$

PoS(LATTICE2024)036

For PBMG, the acceptance rate is straightforward to compute, as it simply corresponds to the Metropolis-Hastings acceptance rate.

The acceptance rate directly reflects the computational cost of our simulation algorithm. In Figure 1, we compare the acceptance rates of both methods for the XY model and ϕ^4 theory.



(a) **(b)**

Figure 1: The acceptance rate for both PBMG and Heatbath Algorithm, for lattice size of 64×64; a) Phi4 Theory, b) XY Model.

For the ϕ^4 theory, we generate samples using both the PBMG and heatbath algorithms for λ values in the range $(1.6, 2.1)$, covering both sides of the phase transition. As shown in Figure 1a, the acceptance rate is approximately 98%. For the XY model, we compare the acceptance rate across different temperature values in the range $(0.5, 2.0)$, as illustrated in Figure 1b. We observe that the acceptance rate for the PBMG model is close to 90%. In both the XY and ϕ^4 models, the acceptance rate remains nearly constant across the parameter (T/λ) values.

6. Conclusion

We have proposed a generative-based heatbath sampler for local lattice systems with continuous degrees of freedom. Our model, PBMG, is straightforward to train as it learns a one-dimensional distribution and can serve as a proposal for a heatbath sampler. A key advantage of this approach is its conditioning on neighboring sites and the action or Hamiltonian parameters, without requiring any training samples from the target distribution. Unlike traditional methods that may need different proposals for different ranges of the action parameter, the PBMG model offers a flexible, single proposal mechanism for updating all lattice sites while covering a broad range of action parameter values. Extending this improved heatbath approach to gauge theory could be an exciting direction for future research.

Acknowledgments

This work was supported by Core Research Grant of SERB, Govt. of India (project no. 003466). AS and SN were supported by the German Federal Ministry of Education and Research under the grant BIFOLD24B.

PoS(LATTICE2024)036

Appendix A: PBMG for XY model

Modeling the Proposal distribution

We use Normalizing Flows to model the proposal distribution $q(\phi_{i,j}|\mathbf{v}_{i,j};\boldsymbol{\theta})$. $p_Z(z|\mathbf{v}_{i,j};\boldsymbol{\theta}_B)$ is the base distribution and $f(z;\boldsymbol{\theta}_R)$ is the invertible transformation used in the Normalizing Flow. Here, $\boldsymbol{\theta} = \{\boldsymbol{\theta_B}, \boldsymbol{\theta_R}\}$. The condition vector that is input to all the neural networks is $\mathbf{v}_{i,j}$.

Using the change of variables formula,

$$q(\phi_{i,j}|\mathbf{v}_{i,j};\boldsymbol{\theta}) = p_Z\left(f^{-1}(\phi_{i,j};\boldsymbol{\theta}_R)|\mathbf{v}_{i,j};\boldsymbol{\theta}_B\right)\left|\det\left(\frac{\partial f^{-1}(\phi_{i,j};\boldsymbol{\theta}_R)}{\partial\phi_{i,j}}\right)\right| \tag{16}$$

We use Rational Quadratic Splines (RQS) as the transform f.

Training and Inference Procedure

The loss function used in the training procedure is the expected value of the KL divergence between the proposal $q(\phi_{i,j}|\mathbf{v}_{i,j};\boldsymbol{\theta})$ and the target $p(\phi_{i,j}|\mathbf{v}_{i,j})$ i.e., the true conditional distribution over all possible values of the condition vector $\mathbf{v}_{i,j}$ during training. The first four components of $\mathbf{v}_{i,j}$ lie in the interval $[0, 2\pi]$ and the last component $T \in [0.13, 2.05]$. We will, therefore, sample $\mathbf{v}_{i,j}$ from $p_v(\mathbf{v}_{i,j}) = \text{Unif}([0, 2\pi]^4 \times [0.13, 2.05])$ to calculate the expectation.

$$\mathcal{L} = \mathbb{E}_{\mathbf{v}_{i,j}\sim p_v(\mathbf{v}_{i,j})}\left[\mathbb{E}_{z\sim p_Z(z|\mathbf{v}_{i,j};\boldsymbol{\theta}_B)}\left[\log q(\phi_{i,j}|\mathbf{v}_{i,j};\boldsymbol{\theta}) - \log p(\phi_{i,j}|\mathbf{v}_{i,j})\right]\right] \tag{17}$$

The Monte Carlo approximation can be used to estimate the above expectation as follows

$$\mathcal{L} \approx \frac{1}{n}\cdot\frac{1}{N}\sum_{r=1}^{n}\sum_{k=1}^{N}[\log p_Z(z_k|(\mathbf{v}_{i,j})_r;\boldsymbol{\theta}_B) + \log|\det J_f(z_k|(\mathbf{v}_{i,j})_r;\boldsymbol{\theta}_R)|^{-1} - \log p\left(f(z_k;\boldsymbol{\theta}_R)|(\mathbf{v}_{i,j})_r\right)] \tag{18}$$

We sample from $p_v(\mathbf{v}_{i,j})$ to generate 10,000 samples of $\mathbf{v}_{i,j}$ and use this as the training set. We use the Adam optimizer with default hyperparameters and a cosine decay schedule for the learning rate with an initial learning rate of 0.0005 and 20,000 decay steps. We perform validation by calculating the average acceptance rate for 24 random temperatures in the range [0.13, 2.05] and stop training when we achieve an average acceptance rate of around 85%. Our model gets trained quickly, taking less than an hour to train on a low-end single GPU machine.

The procedure for MCMC sampling using PBMG-XY is briefed in the algorithm given below. Here, $\mathbf{V}_g = [\mathbf{v}_{i,j}]_{(i,j)\in g}$.

Appendix B: PBMG for ϕ^4 model

Modeling the Proposal Distribution: PBMG-ϕ^4

We construct the proposal distribution for the scalar ϕ^4 theory by using a Gaussian Mixture Model with six Gaussian components. The proposal distribution parameterized by $\boldsymbol{\theta}$, can be written as

$$q(\phi_{i,j}|\mathbf{v}_{i,j};\boldsymbol{\theta}) = \sum_{k=1}^{6}\pi_k(\mathbf{v}_{i,j};\boldsymbol{\theta}_k)\mathcal{N}(\phi_{i,j}|\mu_k(\mathbf{v}_{i,j};\boldsymbol{\theta}_k), \sigma_k(\mathbf{v}_{i,j};\boldsymbol{\theta}_k)) \tag{19}$$

PoS(LATTICE2024)036

where μ_k, σ_k π_k are the mean, standard deviation and mixing coefficients of the k^{th} Gaussian distribution, and $\boldsymbol{\theta} = \{\boldsymbol{\theta}_k\}_{k=1}^{6}$. The parameters μ_k, σ_k π_k are a function of the condition vector $\mathbf{v}_{i,j} = (\lambda, m^2, \{\phi_{l,m} : (l,m) \in n(i,j)\})$ through a neural network parametrized by $\boldsymbol{\theta}_k$ which are learnt using a suitable loss function. The input to the k^{th} neural network is the condition vector $\mathbf{v}_{i,j}$, and the outputs are the parameters $(\mu_k, \log(\sigma_k), \pi_k)$.

The architectures of all six neural networks are the same, with the only difference lying in the initialization of the network parameters. Each neural network consists of one hidden layer with 500 neurons and a ReLU activation function. The neurons in the final layer use linear activation. The value of $\log(\sigma_k) > 1$ is clipped to 1. Since $\sum_k \pi_k = 1$, the networks output logit values that are converted to π_k by applying softmax.

Training and Inference Procedure

The training procedure of PBMG-ϕ^4 is similar to that of PBMG-XY. The loss function used in the training procedure is the expected value of the KL divergence between the proposal $q(\phi_{i,j}|\mathbf{v}_{i,j};\boldsymbol{\theta})$ and the target $p(\phi_{i,j}|\mathbf{v}_{i,j})$ i.e., the true conditional distribution over all possible values of the condition vector $\mathbf{v}_{i,j}$ during training, along with an L_2 regularization term. The effect of the regularization term is that the mixing coefficients remain close to each other, which, in turn, facilitates accurate training. We train our model for the following range of the parameters: $\lambda \in [2.5, 15]$, $m^2 \in [-8, 0]$ and $\kappa_{i,j} \in [0, 3]$. We will, therefore, sample $\mathbf{v}_{i,j}$ from $p_v(\mathbf{v}_{i,j}) = \text{Unif}([2.5, 15] \times [-8, 0] \times [0, 3])$ to calculate the expectation. Here, $\boldsymbol{\pi}(\mathbf{v}_{i,j};\boldsymbol{\theta}) = [\pi_k(\mathbf{v}_{i,j};\boldsymbol{\theta}_k)]_{k=1}^{6}$ and $\|.\|$ represents the L_2 norm.

$$\mathcal{L} = \mathbb{E}_{\mathbf{v}_{i,j}\sim p_v(\mathbf{v}_{i,j})}\left[\mathbb{E}_{\phi_{i,j}\sim q(\phi_{i,j}|\mathbf{v}_{i,j};\boldsymbol{\theta})}\left[\log\frac{q(\phi_{i,j}|\mathbf{v}_{i,j};\boldsymbol{\theta})}{p(\phi_{i,j}|\mathbf{v}_{i,j})}\right] + \|\boldsymbol{\pi}(\mathbf{v}_{i,j};\boldsymbol{\theta})\|\right] \tag{20}$$

And the Monte Carlo approximation to the above expression is

$$\mathcal{L} \approx \frac{1}{n}\sum_{r=1}^{n}\left[\frac{1}{N}\sum_{k=1}^{N}\left[\log q((\phi_{i,j})_k|(\mathbf{v}_{i,j})_r;\boldsymbol{\theta}) - \log p\big((\phi_{i,j})_k|(\mathbf{v}_{i,j})_r\big)\right] + \|\boldsymbol{\pi}\left((\mathbf{v}_{i,j})_r;\boldsymbol{\theta}\right)\|\right] \tag{21}$$

We sample from $p_v(\mathbf{v}_{i,j})$ to generate 17,500 samples of $\mathbf{v}_{i,j}$ and use this as the training set. We use the Adam optimizer with a learning rate of 0.0001 and default hyperparameters. We perform validation by calculating the average acceptance rate for 50 random sets of parameters (λ, m^2) where $\lambda \in [2.5, 15]$ and $m^2 \in [-8, 0]$, and stop training when we achieve an average acceptance rate of around 98%. PBMG-ϕ^4 also gets trained very quickly taking only a few minutes to train on a low-end single GPU machine.

The procedure for MCMC sampling using PBMG-ϕ^4 is exactly the same as that of PBMG-XY.

References

[1] R.H. Swendsen and J.-S. Wang, *Nonuniversal critical dynamics in Monte Carlo simulations*, *Phys. Rev. Lett.* **58** (1987) 86.

[2] U. Wolff, *Comparison between cluster Monte Carlo algorithms in the ising model*, *Phys. Lett. B* **228** (1989) 379.

PoS(LATTICE2024)036

[3] U. Wolff, *Critical slowing down*, *Nucl. Phys. B Proc. Suppl* **17** (1990) 93.

[4] R.M. Neal et al., *Mcmc using hamiltonian dynamics*, *Handbook of Markov chain Monte Carlo* **2** (2011) 2.

[5] D.J. Rezende and S. Mohamed, *Variational inference with normalizing flows*, 1505.05770.

[6] J. Ho, A. Jain and P. Abbeel, *Denoising diffusion probabilistic models*, 2006.11239.

[7] M.S. Albergo, G. Kanwar, S. Racanière, D.J. Rezende, J.M. Urban, D. Boyda et al., *Flow-based sampling for fermionic lattice field theories*, *Phys. Rev. D* **104** (2021) 114507 [2106.05934].

[8] M.S. Albergo, D. Boyda, K. Cranmer, D.C. Hackett, G. Kanwar, S. Racanière et al., *Flow-based sampling in the lattice Schwinger model at criticality*, *Phys. Rev. D* **106** (2022) 014514 [2202.11712].

[9] D.C. Hackett, C.-C. Hsieh, M.S. Albergo, D. Boyda, J.-W. Chen, K.-F. Chen et al., *Flow-based sampling for multimodal distributions in lattice field theory*, 2107.00734.

[10] K.A. Nicoli, C.J. Anders, L. Funcke, T. Hartung, K. Jansen, P. Kessel et al., *Estimation of thermodynamic observables in lattice field theories with deep generative models*, *Phys. Rev. Lett.* **126** (2021) 032001 [2007.07115].

[11] M. Caselle, E. Cellini and A. Nada, *Sampling the lattice Nambu-Goto string using Continuous Normalizing Flows*, 2307.01107.

[12] A. Singha, D. Chakrabarti and V. Arora, *Conditional normalizing flow for Markov chain Monte Carlo sampling in the critical region of lattice field theory*, *Phys. Rev. D* **107** (2023) 014512 [2207.00980].

[13] A. Singha, D. Chakrabarti and V. Arora, *Generative learning for the problem of critical slowing down in lattice Gross-Neveu model*, *SciPost Phys. Core* **5** (2022) 052 [2111.00574].

[14] J.M. Pawlowski and J.M. Urban, *Reducing autocorrelation times in lattice simulations with generative adversarial networks*, *Mach. Learn. Sci. Tech* **1** (2020) 045011 [1811.03533].

[15] G. Kanwar, M.S. Albergo, D. Boyda, K. Cranmer, D.C. Hackett, S. Racanière et al., *Equivariant flow-based sampling for lattice gauge theory*, *Phys. Rev. Lett.* **125** (2020) 121601 [2003.06413].

[16] T. Eichhorn, G. Fuwa, C. Hoelbling and L. Varnhorst, *Parallel Tempered Metadynamics: Overcoming potential barriers without surfing or tunneling*, 2307.04742.

[17] R. Abbott et al., *Gauge-equivariant flow models for sampling in lattice field theories with pseudofermions*, *Phys. Rev. D* **106** (2022) 074506 [2207.08945].

[18] R. Abbott et al., *Sampling QCD field configurations with gauge-equivariant flow models*, *PoS* **LATTICE2022** (2023) 036 [2208.03832].

PoS(LATTICE2024)036

[19] R. Abbott et al., *Normalizing flows for lattice gauge theory in arbitrary space-time dimension*, 2305.02402.

[20] L. Wang, G. Aarts and K. Zhou, *Diffusion models as stochastic quantization in lattice field theory*, *JHEP* **2024** (2024) 060 [2309.17082].

[21] A. Singha, D. Chakrabarti and V. Arora, *Sampling U(1) gauge theory using a retrainable conditional flow-based model*, *Phys. Rev. D* **108** (2023) 074518 [2306.00581].

[22] J. Finkenrath, *Fine grinding localized updates via gauge equivariant flows in the 2D Schwinger model*, *PoS* **LATTICE2023** (2024) 022 [2402.12176].

[23] Q. Zhu, G. Aarts, W. Wang, K. Zhou and L. Wang, *Diffusion models for lattice gauge field simulations*, 2410.19602.

[24] M. Caselle, E. Cellini and A. Nada, *Stochastic normalizing flows for Effective String Theory*, *PoS* **LATTICE2024** (2025) 027 [2412.19109].

[25] A. Bulgarelli, E. Cellini and A. Nada, *Scaling of Stochastic Normalizing Flows in* SU(3) *lattice gauge theory*, 2412.00200.

[26] M. Creutz, *Monte Carlo study of quantized su(2) gauge theory*, *Phys. Rev. D* **21** (1980) 2308.

[27] A. Kennedy and B. Pendleton, *Improved heatbath method for Monte Carlo calculations in lattice gauge theories*, *Phys. Lett. B* **156** (1985) 393.

PoS(LATTICE2024)036