PAPER • **OPEN ACCESS**

Neural importance resampling: a practical sampling strategy for neural quantum states

IOP Publishing Mach. Learn.: Sci. Technol. 7 (2026) 015020 https://doi.org/10.1088/2632-2153/ae37c9

MACHINE LEARNING
Science and Technology

PAPER

OPEN ACCESS

RECEIVED
25 July 2025

REVISED
5 November 2025

ACCEPTED FOR PUBLICATION
13 January 2026

PUBLISHED
29 January 2026

Neural importance resampling: a practical sampling strategy for neural quantum states

Eimantas Ledinauskas[1,2,*] **and Egidijus Anisimovas**[1]

[1] Institute of Theoretical Physics and Astronomy, Vilnius University Saulėtekio al. 3, LT-10257 Vilnius, Lithuania
[2] Baltic Institute of Advanced Technology, Pilies St. 16-8, LT-01403 Vilnius, Lithuania
[*] Author to whom any correspondence should be addressed.

E-mail: eimantas.ledinauskas@ff.vu.lt and egidijus.anisimovas@ff.vu.lt

Keywords: variational Monte Carlo, neural quantum states, many-body quantum systems, importance resampling, autoregressive neural networks

Abstract

Neural quantum states (NQS) have emerged as powerful tools for simulating many-body quantum systems, but their practical use is often hindered by limitations of current sampling techniques. Markov chain Monte Carlo (MCMC) methods suffer from slow mixing and require manual tuning, while autoregressive NQS impose restrictive architectural constraints that complicate the enforcement of symmetries and the construction of determinant-based multi-state wave functions. In this work, we introduce neural importance resampling (NIR), a new sampling algorithm that combines importance resampling with a separately trained autoregressive proposal network. This approach enables efficient and unbiased sampling without constraining the NQS architecture. We demonstrate that NIR supports stable and scalable training, including for multi-state NQS, and mitigates issues faced by MCMC and autoregressive approaches. Numerical experiments on the 2D transverse-field Ising and J_1–J_2 Heisenberg models show that NIR outperforms MCMC in challenging regimes and yields results competitive with state of the art methods. Our results establish NIR as a robust alternative for sampling in variational NQS algorithms.

1. Introduction

Neural quantum states (NQS), originally introduced by [5], have seen rapid methodological advancements, enabling their successful application to an increasingly diverse range of many-body quantum systems [6, 19, 23, 40]. Today, NQS are considered state-of-the-art tools for tackling the complexities of many-body quantum physics.

Methods that utilize NQS typically require sampling configurations according to the probability distribution defined by the NQS wave function. Given the exponentially large dimensionality of many-body quantum systems, this sampling is commonly performed using Markov chain Monte Carlo (MCMC) techniques, particularly the Metropolis-Hastings (MH) algorithm. However, this approach has several well-known drawbacks, including slow mixing, autocorrelation, difficulties in equilibration, and susceptibility to local traps. Additionally, effective sampling often relies on careful tuning of the proposal distribution.

These issues can be addressed using autoregressive NQS, which enable exact and efficient sampling of basis states [16, 38]. However, this approach imposes architectural constraints on the neural networks, introducing new limitations. For instance, enforcing global constraints and symmetries of the wave function becomes more challenging and requires special approaches which can be suboptimal [32]. Another limitation of autoregressive NQS arises in the context of a recent method proposed by [29], which aims to approximate multiple lowest energy states. This method involves variational Monte Carlo (VMC) with a multi-state wave function constructed as a determinant of multiple single-state NQS. It remains unclear how to perform exact sampling from such a determinant wave function, even when each constituent NQS is autoregressive.

In this work, we introduce an alternative sampling algorithm based on importance resampling (IR), with proposals generated by a separate autoregressive neural network. This approach employs a dedicated sampling neural network (SNN) that automatically learns the probability distribution defined by the NQS, eliminating the need for manually designed proposal distributions. The learned proposal distribution is efficient enough for IR to be viable in the high-dimensional spaces typical of many-body quantum systems. We argue that this algorithm is practical for NQS applications and offers advantages over both MCMC and autoregressive NQS sampling. Compared to MCMC, IR is simpler, makes fewer assumptions, and is therefore more robust. Additionally, thanks to the trainable SNN, it requires less manual tuning for specific problems. In contrast to autoregressive NQS, our method does not impose architectural constraints on the NQS itself, since sampling is delegated to a separate network. This allows for straightforward enforcement of constraints and facilitates the use of multi-state determinant NQS.

The remainder of this paper is organized as follows. In section 2, we provide a brief overview of the necessary background and context. Section 3 introduces the proposed sampling method in detail and discusses the related work. In section 4, we present numerical experiments that demonstrate the practicality and effectiveness of our approach. Finally, section 5 offers concluding remarks.

2. Context

2.1. Single-state NQS

To streamline presentation, we focus in this work on systems with discrete degrees of freedom, such as spin lattices or tight-binding models. These systems can be described by a wave function, which is a complex vector of the form:

$$|\psi\rangle = \sum_s \psi(s)\,|s\rangle. \tag{1}$$

Here, $|s\rangle$ are the basis vectors in the computational basis defined as the tensor product of the states at a single site and $\psi(s) = \langle s|\psi\rangle$ are the corresponding components of the state vector. In many-body systems, the dimension of this vector grows exponentially with system size, quickly becoming intractable to handle directly.

NQS use a neural network with parameters θ to represent the wave function by mapping $|s\rangle$ to the corresponding components $\psi_\theta(s)$. This approach compresses the otherwise intractable wave function vector into a more manageable neural network representation. The parameters θ can be optimized for specific tasks. NQS have been widely applied in various contexts, including: ground state approximation via VMC, quantum state tomography, and time evolution [6, 19, 23, 40].

2.2. Multi-state NQS

[29] proposed a generalization of VMC that enables the simultaneous search for multiple low-energy eigenstates. This is achieved by transforming the problem of finding excited states of a given system into the problem of finding the ground state of an expanded system. While the method can be used with various types of variational ansatzes, it is particularly well-suited to NQS. The approach requires the variational wave function to take the following form:

$$\mathbf{\Psi}\left(s^1,\ldots,s^K\right) = \det\begin{pmatrix} \psi_1\left(s^1\right) & \cdots & \psi_K\left(s^1\right) \\ \vdots & \ddots & \vdots \\ \psi_1\left(s^K\right) & \cdots & \psi_K\left(s^K\right) \end{pmatrix}. \tag{2}$$

Here, $s^1,\ldots,s^K$ is a tuple of K basis states of the original system, and $\psi_j(s^k)$ are single state ansatzes (neural networks in our case). The energy is generalized from a scalar to a matrix as follows:

$$\mathbf{E}_\Psi = \left\langle \begin{pmatrix} \psi_1\left(s^1\right) & \cdots & \psi_K\left(s^1\right) \\ \vdots & \ddots & \vdots \\ \psi_1\left(s^K\right) & \cdots & \psi_K\left(s^K\right) \end{pmatrix}^{-1} \begin{pmatrix} \hat{H}\psi_1\left(s^1\right) & \cdots & \hat{H}\psi_K\left(s^1\right) \\ \vdots & \ddots & \vdots \\ \hat{H}\psi_1\left(s^K\right) & \cdots & \hat{H}\psi_K\left(s^K\right) \end{pmatrix} \right\rangle_{s^1,\ldots,s^K\sim\Psi^2}. \tag{3}$$

Here, $\langle\ldots\rangle_{s^1,\ldots,s^K\sim\Psi^2}$ denotes the expectation value over tuples of basis states sampled according to the probability distribution defined by $\mathbf{\Psi}^2$ and $\hat{H}$ is the Hamiltonian operator of the original system. The trace of the energy matrix, which is minimized during VMC, represents the total energy of the K states. Estimates of the individual lowest-energy eigenvalues can be obtained by diagonalizing the matrix after VMC convergence.

2.3. Observable and gradient estimation

Because the full wave function vector is intractable, the expectation values of observables must be estimated stochastically. This is accomplished using the following well-known identities:

$$O_{\mathrm{loc}}(s) = \sum_{s'} \frac{\psi(s')}{\psi(s)} O_{s,s'} \tag{4}$$

$$\left\langle \hat{O} \right\rangle = \langle O_{\mathrm{loc}}(s) \rangle \approx \frac{1}{N} \sum_{s \sim \psi^2} O_{\mathrm{loc}}(s). \tag{5}$$

Here, $O_{\mathrm{loc}}(s)$ denotes the local estimator of the operator $\hat{O}$, and $O_{s,s'} = \langle s|\hat{O}|s'\rangle$ are the matrix elements of the operator. This method is feasible only when $O_{s,s'}$ is sparse, allowing efficient computation of $O_{\mathrm{loc}}(s)$. The approximation relies on sampling basis states s according to the probability distribution defined by the squared amplitude of the NQS wave function.

One particularly important observable is the energy, which is minimized during VMC to obtain the ground state. The gradient of the energy with respect to the NQS parameters—necessary for NQS optimization—can be computed using the following identity:

$$\frac{\partial E}{\partial \theta_j} = 2\Re \left\{ \left\langle H_{\mathrm{loc}}(s) \frac{\partial}{\partial \theta_j} \log \psi^*(s) \right\rangle - \langle H_{loc}(s) \rangle \left\langle \frac{\partial}{\partial \theta_j} \log \psi^*(s) \right\rangle \right\}. \tag{6}$$

As with the expectation values of the operators, evaluating this gradient requires sampling from the NQS distribution.

2.4. MCMC sampling

One commonly used method to sample basis states from NQS is the MH algorithm [12, 24]. MH constructs a Markov chain with transition probabilities defined by a proposal distribution $T(s \to s')$ and an acceptance ratio:

$$A(s \to s') = \min\left(1, \frac{|\psi(s')|^2 T(s' \to s)}{|\psi(s)|^2 T(s \to s')}\right). \tag{7}$$

Under the conditions of irreducibility, aperiodicity, and detailed balance, this procedure guarantees convergence to the desired stationary distribution.

The choice of proposal distribution T strongly influences sampling efficiency. While simple local proposals are often sufficient, some specific systems require nontrivial or learned proposals. Dynamic proposals, such as those parameterized by neural networks, can adapt to the structure of the target distribution and improve mixing. However, ensuring proper MCMC convergence with dynamic proposal distributions imposes some nontrivial constraints. In particular, dynamically learned or adaptive proposals must be designed to preserve the ergodicity and detailed balance (or at least stationarity) of the Markov chain. One essential constraint is diminishing adaptation, which requires that the magnitude of changes to the proposal mechanism decrease over time, ensuring that the chain stabilizes and avoids persistent non-stationarity [34]. Another is bounded convergence, which requires that the adapted proposals remain within a controlled family that maintains uniform ergodicity [1].

2.5. Autoregressive NQS

Another common approach to sampling from NQS is to use autoregressive neural networks [16, 38]. Any probability distribution over a discrete set of variables can be factorized into a product of conditional probabilities. Analogously, wave functions $\psi(s)$ defined on discrete configurations $s = (s_1, \ldots, s_N)$—where each s_j may represent, for example, a quantum number associated with a lattice site—can be written in the form:

$$\psi(s) = \prod_{j=1}^{N} \psi_j\left(s_j \,|\, s_1, \ldots, s_{j-1}\right). \tag{8}$$

These conditional amplitudes can be modeled using an autoregressive neural network, which enables exact sampling from $|\psi(s)|^2$ by sequentially generating each s_j from the corresponding conditional distribution. This eliminates the need for MCMC.

Autoregressive NQS offer efficient and unbiased sampling but have certain limitations. The autoregressive structure makes it more difficult to enforce global constraints and symmetries. For example,

[32] found that feed-forward NQS outperform autoregressive NQS in modeling the J_1–J_2 model, primarily due to a suboptimal symmetrization procedure imposed by the autoregressive structure. Moreover, it is unclear how to extend the autoregressive framework to support the multi-state NQS construction described in section 2.2. Even when single-state wave functions are modeled autoregressively, it remains unresolved how this structure can be leveraged to sample from the full determinant-based ansatz.

3. Neural importance resampling (NIR)

3.1. Importance resampling (IR)

IR, also known as sampling-IR [35], is an algorithm used to obtain samples from a target distribution $p(s)$ by leveraging samples from a proposal distribution $q(s)$. First, a batch of N_q samples s^j is drawn from $q(s)$. Then, importance weights are computed for each sample as $w_j = p(s^j)/q(s^j)$. Finally, a new batch of N_p samples is drawn from the original batch, with selection probabilities proportional to the importance weights. This method is simple to implement and easy to understand. However, its main limitation is that, in high-dimensional settings, sampling efficiency requires the proposal distribution to overlap significantly with the target distribution. Designing such a proposal distribution is often challenging, which usually restricts the method's applicability to low-dimensional problems.

We note that if sampling is used solely to compute expectations under the target distribution, then importance sampling can be applied by simply reweighting samples drawn from the proposal distribution when computing averages. This approach should perform similarly to resampling. In this work, we chose to use resampling purely for implementation reasons: It allows us to replace only the sampler component of an existing computation pipeline without modifying the rest of the code.

3.2. Autoregressive proposal network

As mentioned in the previous section, IR requires substantial overlap between the proposal and target distributions. We propose that, in high-dimensional settings, this overlap can be achieved by using an autoregressive neural network as the proposal distribution. To ensure that the proposal remains close to the target, the network is trained prior to sampling. In this work, we use a Transformer architecture [39] (with reordered layer normalization [41]), motivated by the success of Transformers as multi-domain generative models [31], which demonstrates their ability to learn highly complex distributions. The approach for sampling in this case is identical to the autoregressive NQS described in section 2.5, except that the neural network now outputs the probability directly instead of the complex amplitude. Each site configuration is passed as a separate input token, encoded as a learnable embedding vector of dimension d_{embd}. Learnable positional embeddings are added to these token embeddings. The resulting vectors are then processed by standard transformer layers using a causal attention mask. Each output vector is subsequently linearly projected to a scalar value, which represents the conditional probability logits. After applying a softmax, these values are interpreted as $p(s_j|s_1,\ldots,s_{j-1})$. To ensure that the prediction for s_j depends only on the previous site configurations $s_1,\ldots,s_{j-1}$, the input token sequence is shifted. This is achieved by prepending a learnable token to the beginning of the input sequence and discarding the final token.

3.3. Forward Kullback–Leibler divergence (KLD) loss

Maximizing the overlap between the proposal and the target distributions can be achieved by minimizing the KLD. There are two variants of KLD: the forward (mean-seeking) form,

$$L_{\text{f}} = \sum_s p(s) \log \frac{p(s)}{q(s)}, \tag{9}$$

and the backward (mode-seeking) form,

$$L_{\text{b}} = \sum_s q(s) \log \frac{q(s)}{p(s)}. \tag{10}$$

The gradients of these objectives can be estimated using the following expressions:

$$\frac{\partial L_{\text{f}}}{\partial \theta_j} = -\left\langle \frac{\partial}{\partial \theta_j} \log q(s) \right\rangle_{s\sim p}, \tag{11}$$

$$\frac{\partial L_{\text{b}}}{\partial \theta_j} = \left\langle (\log q(s) - \log p(s)) \frac{\partial \log q(s)}{\partial \theta_j} \right\rangle_{s\sim q}. \tag{12}$$

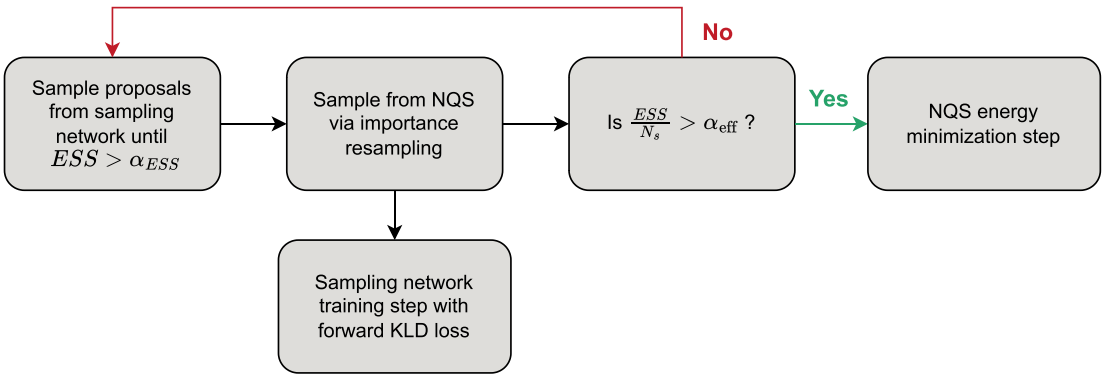

Figure 1. Scheme for the proposed NQS training algorithm.

The gradient of the backward KLD is easier to estimate, as it only requires sampling from $q(s)$, which can be done efficiently and without bias using the autoregressive sampling network. However, the forward KLD is better aligned with the requirement of IR that $q(s) > 0$ whenever $p(s) > 0$ due to its mean-seeking property. The downside of forward KLD is that it requires samples drawn from $p(s)$ (the same resampled configurations that are used for NQS optimization), which can be biased if the overlap between the NQS and sampling network distributions is not yet sufficient. In our numerical experiments, we found that both loss functions are effective, but forward KLD results in more robust training of NQS, with reduced sensitivity to hyperparameters. Training with the backward KLD often results in mode collapse, where certain subsets of configurations with high $p(s)$ have very low $q(s)$ and are rarely proposed, preventing the IR step from correcting this bias. This behavior is visible in the effective sample size (see section 3.4, which occasionally spikes when a configuration with very small $q(s)$ is nevertheless sampled, producing very large importance weights. We observed these spikes when using the backward KLD and switched to the forward KLD, after which the issue disappeared. To further enforce the requirement that $q(s) > 0$ whenever $p(s) > 0$, we add a small floor p_{floor} to the single-site probabilities, ensuring that all configurations satisfy $q(s) > 0$.

In this work, we focus on the multi-state NQS described in section 2.2, which is permutation-invariant with respect to the order of the input configurations $s^1, \ldots, s^K$ by construction. To encourage the proposal network to learn this property, we randomly permute the order of the states before using them for training. This approach could be extended to incorporate other types of symmetries, depending on the Hamiltonian under study.

3.4. Adaptive proposal network retraining

The effectiveness of NIR critically depends on the overlap between the target distribution, given by the NQS, and the proposal distribution produced by the sampling network. However, during training, this overlap can degrade rapidly if the NQS and the sampling network begin to diverge. In such cases, the NIR resampling procedure can fail to adequately correct for the mismatch between distributions. To mitigate this issue, we monitor the effective sample size $ESS = \left(\sum_j w_j\right)^2 / \sum_k w_k^2$. We sample configurations from the proposal network until ESS exceeds a predefined threshold α_{ESS}. If the overlap with the target distribution is poor, more proposals must be generated. This compensates for low overlap and results in stable training with high-quality sample batches. After that, IR is used to obtain samples from the NQS. These samples are then used to train the proposal network with the forward KLD loss.

However, if the overlap between the proposal network and NQS distributions is low, sampling can become very slow. To prevent divergence between the two distributions, we introduce a second safeguard: the obtained samples are used to train the NQS only if the sampling efficiency, defined as the ratio between ESS and the total number of proposed configurations, exceeds a predefined threshold α_{eff}. If the efficiency is too low, we repeat the sampling and proposal-network-training steps. This provides a simple mechanism for adaptive retraining and helps maintain alignment between the distributions during optimization. A block diagram for this algorithm is illustrated in figure 1.

3.5. Related work

Recent work across machine learning and computational science communities has shown that IR becomes practical in high-dimensional spaces once the proposal distribution is learned by a neural

network. Neural Adaptive Sequential Monte Carlo introduced gradient-based adaptation of recurrent proposal networks, minimizing the forward-KLD to the target and repeatedly resampling particles to maintain diversity [11]. Building on this idea, Auto-Encoding SMC integrates proposal learning with variational inference, treating resampling weights as part of an evidence lower bound that is optimized jointly with generative and proposal networks [20]. In computer graphics, Neural Importance Sampling employs normalizing flows to learn globally coherent proposals that drastically reduce Monte Carlo variance when integrating high-dimensional light-transport integrands [25]. Flow-based proposals have also been used for event generators in particle physics, where exhaustive neural importance sampling yields unbiased estimates with competitive efficiency [30].

At the same time, generative neural samplers have been applied in statistical physics. For instance, [26] propose a framework for asymptotically unbiased estimation of observables (including partition-function-dependent ones) using generative neural samplers. Autoregressive neural networks have also been used as adaptive proposals in MCMC for spin-glass models, yielding improved efficiency [22]. More broadly, adaptive MCMC methods augmented with normalizing-flow transitions have been shown to learn nonlocal proposal kernels and accelerate sampling in high-dimensional spaces [9]. Recent work further explores this interface between flow-based proposals and energy-based modeling: adaptive flow samplers have been coupled with energy-based models to maintain balanced learning dynamics between data-driven and model-driven components [10], while systematic benchmarks in statistical-physics settings have clarified when machine-learning-assisted Monte Carlo methods truly outperform conventional algorithms [8].

Collectively, these studies demonstrate that neural proposal models trained with KL-based objectives can achieve substantial overlap with the target distribution required by IR, motivating our adoption of a dedicated SNN in the NIR algorithm.

4. Numerical experiments

4.1. Transverse-field Ising (TFI) model

In this section, we present the results of numerical experiments performed on the paradigmatic two-dimensional TFI model. The Hamiltonian for this system is given by:

$$\hat{H} = -\sum_{\langle j,k\rangle} \hat{\sigma}_z^j \hat{\sigma}_z^k - g \sum_j \hat{\sigma}_x^j, \tag{13}$$

where $\langle j,k\rangle$ denotes pairs of nearest-neighbor sites on the lattice, $\hat{\sigma}_x^j$ and $\hat{\sigma}_z^j$ are Pauli matrices acting on site j, and g is the strength of the transverse field. The first term favors spin alignment along the z-axis, promoting the ferromagnetic order, while the second term controls quantum fluctuations that compete with this order and drive the system towards the paramagnetic phase. On an infinite square lattice the phase transitions between these regimes occurs at $g \approx 3.0444$ [3], and due to the quantum-to-classical mapping the transition belongs to the 3D classical-Ising universality class. Thus, the TFI model provides a very suitable benchmark problem that combines conceptual simplicity with nontrivial behavior.

The largest TFI lattice considered in this work is 8×8, which may seem small for modern methods. However, to target multiple low-energy states the Hilbert space must be extended (see section 2.2). For three states, the system effectively has 192 sites, which is comparable in size to a 14×14 lattice.

4.2. Multi-state NQS details

We search for the three lowest energy states, which requires three separate neural networks to construct the multi-state NQS described in section 2.2. We use a multilayer perceptron (MLP) architecture for these networks. Each network consists of $n_{\text{layer}} = 4$ hidden layers, with each layer containing $d_{\text{mlp}} = 256$ neurons. Every linear projection is followed by layer normalization [21] and the GELU activation function [15]. The architecture of the sampling proposal network is described in section 3.2. We use ADAM optimizer [18] for both the NQS and the sampling network. For the NQS, we also employ the minSR method [7] method to improve convergence. A table of hyperparameters can be found in section A. The NQS code was implemented using the JAX [4], Equinox [17], and Optax [2] libraries.

4.3. Comparing NIR with MCMC

As previously noted in [38] and [36], the TFI poses challenges for NQS with an MCMC sampler in the ferromagnetic phase. The primary difficulty lies in the probability distribution of the degenerate ground state, which is sharply peaked, with the two most probable configurations (corresponding to all spins up and all spins down) being far apart in configuration space.

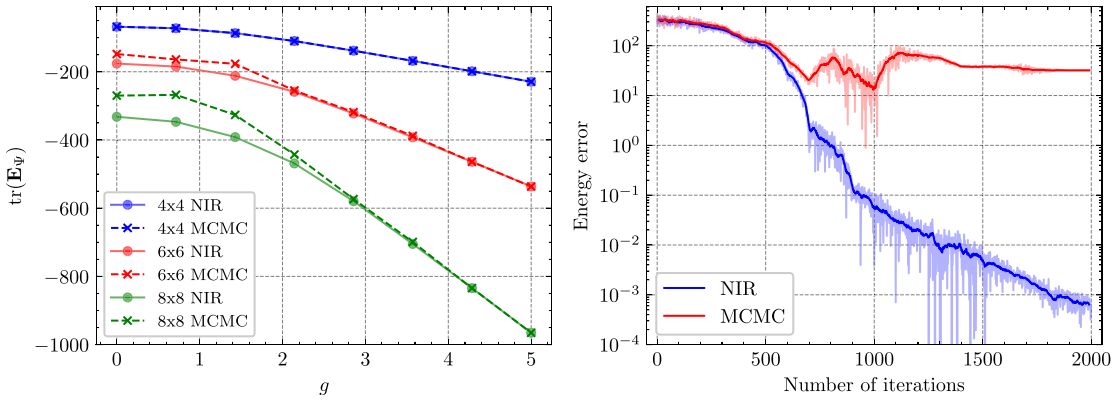

Figure 2. Left: Dependence of the trace of the final variational energy matrix on transverse field strength g for three lattice sizes (4×4, 6×6, 8×8). For every combination, two data points are shown, differing only in the sampler used. Right: Example of the error evolution in the sum of the three lowest energies during a training run of NQS using NIR (blue) and MCMC (red) for TFI on an 8×8 lattice and $g = 0.1$. The transparent curves show the values obtained at each iteration from the batch used in the optimization step, and the solid lines represent their moving averages.

We performed multiple pairs of identical runs: one using the NIR sampler and the other using the MCMC sampler, keeping all other hyperparameters fixed. The comparison of the final variational energy (the trace of the energy matrix defined in equation (3)) across different transverse-field strengths and lattice sizes is shown in the left panel of figure 2. The data points correspond to averages over three runs with different random seeds. On small lattices, such as 4×4, both methods perform similarly. However, in the ferromagnetic regime, a performance gap appears already on the 6×6 lattice and becomes more pronounced on the 8×8 lattice. With the MCMC sampler, training often becomes unstable, and the NQS frequently gets trapped in an excited state with energy significantly above the lowest three. In contrast, NIR maintains stable training. The right panel of figure 2 shows the typical evolution of the energy error during training for both methods with 8×8 lattice and $g = 0.1$.

Since the only difference between the runs is the sampling method, and the divergence appears in the ferromagnetic regime, this strongly suggests that the issue arises because of MCMC sampling failure. To further support this, we compare the samplers against exact sampling on a small 3×2 lattice with $g = 0.01$. We note that with the multi-state formulation used to target the three lowest energy states, this system effectively has 18 sites and is near the practical limit for exact sampling. We perform the comparison by sampling 25 600 configurations using either the NIR or MCMC sampler, estimating the empirical configuration probabilities from these samples, and then computing the Jensen–Shannon divergence (JSD) relative to the exact probabilities obtained by evaluating the NQS over all configurations. The evolution of JSD during NQS optimization is shown in figure 3. Early in training, the values are similar, but as the NQS approaches the lowest energy states, the JSD for MCMC increases because the distribution becomes difficult to sample locally, while the JSD for NIR decreases as the proposal network learns to approximate the NQS distribution. In this experiment, we disabled the adaptive proposal-network training described in section 3.4. Instead, we used a fixed number of proposal samples and performed one proposal network update per NQS update to simplify the JSD dynamics. For such a small system, the overlap between the MCMC and true distributions remains sufficient for successful NQS training. However, as lattice size increases, the locality of the MCMC sampler becomes increasingly problematic, and the advantage of NIR becomes clearly visible in the NQS training dynamics.

4.4. Comparing NIR with density matrix renormalization group (DMRG)

To demonstrate that our method performs competitively in the critical regime, we compute the dependence of the three lowest energy levels on g in the range from 0 to 6 for 8×8 lattice. Figure 4 presents the results and compares them with those obtained using the DMRG [28, 37] method, with the virtual bond dimension set to 100. We used TeNPY [13] package to implement DMRG. The critical regime poses a significant general challenge for numerical studies as the correlation length diverges and the system becomes gapless, making the convergence slower. In particular, DMRG techniques rely on the low-entanglement approximation, and in large systems, they would need exponentially larger bond dimensions. On the absolute scale, both methods show good agreement and capture the closing and reopening of the energy gap for $g \in (2, 3)$.

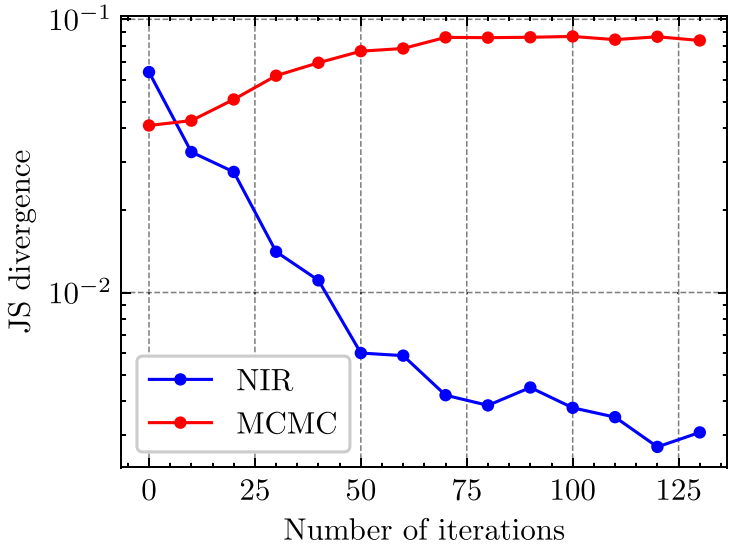

Figure 3. Evolution of the JSD between empirical configuration probabilities from the MCMC and NIR samplers and the exact probabilities during NQS optimization for a 2×3 lattice and $g = 0.01$.

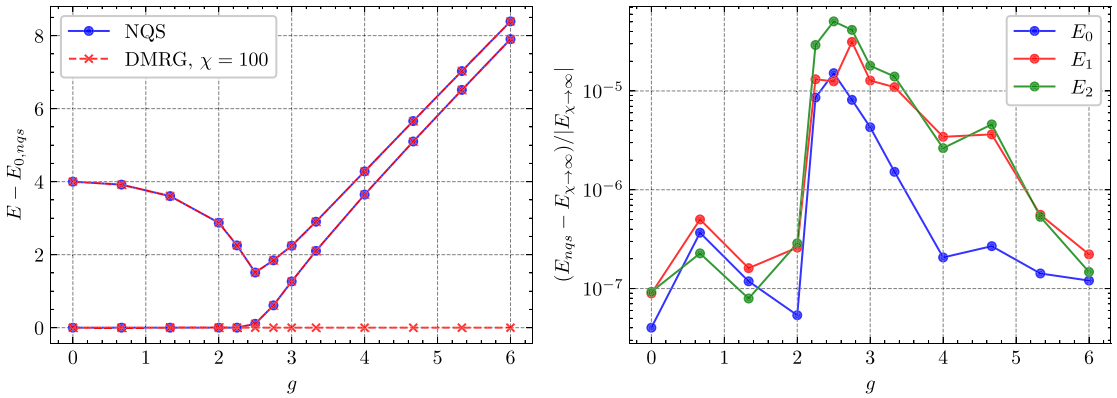

Figure 4. Left: Dependence of the three lowest energy levels (relative to the ground state found with NQS) on g for TFI on an 8×8 lattice. Computed using NQS (blue) and DMRG (red). **Right:** Dependence of the difference between energies obtained with NQS and DMRG on g.

To benchmark our results more precisely, we computed energies using four DMRG bond dimensions (25, 50, 100, and 200) and extrapolated to infinite bond dimension by fitting $E(\chi) = a + b(\chi + c)^d$, where a, b, c, and d are fitted parameters. The right panel of figure 4 shows the energy error relative to these extrapolated values. The agreement is on the order of 10^{-7} at low and high g, and on the order of 10^{-5} near the critical region.

4.5. Benchmarking on Heisenberg J_1–J_2 model

In this section, we benchmark the NIR sampler by searching for the ground state of the J_1–J_2 Heisenberg model at the point of maximum frustration, $J_2 = J_1/2$. This problem has become a standard benchmark in the NQS literature. The Hamiltonian is

$$\hat{H} = J_1 \sum_{\langle i,j \rangle} \vec{S}_i \cdot \vec{S}_j + J_2 \sum_{\langle\langle i,j \rangle\rangle} \vec{S}_i \cdot \vec{S}_j, \tag{14}$$

where $\vec{S}_j$ are spin-$\frac{1}{2}$ operators defined on the sites of 2D square lattice with periodic boundary conditions. The model includes competing antiferromagnetic interactions between nearest neighbors $\langle i,j \rangle$ and next-nearest neighbors $\langle\langle i,j \rangle\rangle$, with interaction strengths J_1 and J_2, respectively.

For this experiment, we use a residual convolutional NQS architecture [14]. The input is first passed through a single convolutional layer to increase its channel dimension to 16. It is then processed by four residual blocks, each consisting of two sequential convolutional layers with an identity skip connection.

Table 1. table for final variational energies and sampling efficiencies for different sizes of proposal network.

d_{embd}	n_{layer}	n_{param}	E/N	ESS/n_{proposed}
16	2	7938	—	0.01
16	3	11 154	—	0.03
16	4	14 370	−0.4969	0.10
32	2	30 210	−0.4966	0.15
32	3	42 786	−0.4968	0.21
32	4	55 362	−0.4969	0.28

Afterward, global average pooling is applied, and a final linear layer maps the result to two values representing the real and imaginary parts of $\log\psi(s)$. All convolutional layers use 3×3 kernels with circular padding to preserve spatial dimensions, leave the channel dimension unchanged. Each convolutional layer is followed by layer normalization [21] and a GELU activation function [15].

Due to circular padding and global pooling, the network is translationally invariant. We enforce spin-inversion symmetry by flipping all spins in the input if the first spin is down. Rotation and reflection symmetries of the Hamiltonian are enforced by applying the corresponding transformations to the input configurations and averaging the outputs. We also restrict the Hilbert space to the zero-magnetization sector by subtracting 30 from the real part of $\log\psi(s)$ when a configuration has non-zero magnetization. This prevents NIR from sampling those states, even though the proposal network does not explicitly enforce this constraint.

The proposal network architecture and NIR algorithm are the same as in the TFI experiments. We do not enforce symmetries or magnetization constraints in the proposal network. In principle, one could apply random symmetry-group transformations to sampled configurations to make NIR respect the symmetries, but we found this unnecessary and did not add it for simplicity.

We performed multiple runs on the 10×10 using the same NQS architecture but different proposal network sizes. The results are summarized in table 1, which lists the proposal network embedding dimension, number of transformer layers, total number of learnable parameters, final variational energy per site, and the average sampling efficiency over the run. The runs in the first two rows were not completed because their sampling efficiency was too low, making them computationally impractical. In these cases, the proposal network failed to approximate the distribution well enough. The remaining runs give nearly identical final results, despite the proposal networks achieving different approximation accuracies. This is enabled by the adaptive scheme described in section 3.4, which compensates for lower overlap between distributions by increasing the number of proposal samples.

To our knowledge, some of the lowest energies per site reported for the J_1–J_2 model on the 10×10 square lattice at maximum frustration point are $-0.497\,627$ [7], $-0.497\,629$ [27], and $-0.497\,634$ [33] (see table 1 in [33] for a broader comparison). Our result is slightly higher, but the goal here is not to set a new state of the art. Rather, it is to demonstrate that the NIR sampler does not break down when applied to a highly frustrated system and therefore serves as a practical alternative for general NQS applications.

5. Conclusions

We have introduced NIR, a sampling approach for NQS that overcomes some key limitations of both MCMC and autoregressive NQS methods. By decoupling the sampling process from the NQS architecture and employing a dedicated, trainable autoregressive neural network, NIR enables efficient and robust sampling via IR. This approach avoids the architectural constraints of autoregressive NQS while mitigating the mixing and equilibration issues associated with MCMC. Through numerical experiments on the TFI and J_1–J_2 Heisenberg models, we have demonstrated that NIR achieves stable training even in regimes where MCMC fails and produces competitive results with DMRG across a wide parameter range. Our work suggests that NIR is a practical and scalable alternative for sampling in NQS-based variational algorithms, particularly in the context of multi-state NQS where traditional sampling approaches may fall short.

Data availability statement

Our implementation includes several unpublished speed-optimization techniques. We intend to release these in a subsequent paper, so we are not open-sourcing the codebase at this stage to protect the novelty and priority of our contributions. The data that support the findings of this study are available upon reasonable request from the authors.

Acknowledgments

This project has received funding from the Research Council of Lithuania (LMTLT), Agreement No. S-ITP-24-6.

Appendix. Hyperparameters

The following table contains a list of hyperparameters used in numerical experiments with TFI model.

Table A1. Hyperparameters table.

Hyperparameter	Symbol	Value
Number of MLP hidden layers	n_{layer}	4
Number of neurons in MLP hidden layers	d_{mlp}	128
Proposal network embedding dimension	d_{embd}	32
Proposal network attention heads	—	4
Proposal network number of transformer layers	—	4
NQS learning rate	—	10^{-3}
Proposal network learning rate	—	10^{-3}
NQS training steps	—	10 000
Batch size (both NQS and proposal network)	—	512
ESS threshold	α_{ESS}	2.0
Sampling efficiency threshold	α_{eff}	0.1
MCMC warmup steps	—	640
MCMC thinning interval	—	10

ORCID iDs

Eimantas Ledinauskas 0000-0001-7802-4334
Egidijus Anisimovas 0000-0001-8919-7265

References

[1] Andrieu C and Thoms J 2008 A tutorial on adaptive MCMC *Stat. Comput.* **18** 343–73
[2] Babuschkin I *et al* 2020 The DeepMind JAX Ecosystem (available at: http://github.com/deepmind)
[3] Blöte H W J and Deng Y 2002 Cluster Monte Carlo simulation of the transverse Ising model *Phys. Rev. E* **66** 066110
[4] Bradbury J *et al* 2018 JAX: composable transformations of Python+NumPy programs (available at: http://github.com/google/jax)
[5] Carleo G and Troyer M 2017 Solving the quantum many-body problem with artificial neural networks *Science* **355** 602–6
[6] Carrasquilla J 2020 Machine learning for quantum matter *Adv. Phys. X* **5** 1797528
[7] Chen A and Heyl M 2024 Empowering deep neural quantum states through efficient optimization *Nat. Phys.* **20** 1476–81
[8] Bono L M D, Ricci-Tersenghi F and Zamponi F 2025 Performance of machine-learning-assisted Monte Carlo in sampling from simple statistical physics models *Phys. Rev. E* **112** 045307
[9] Gabrié M, Rotskoff G M and Vanden-Eijnden E 2022 Adaptive Monte Carlo augmented with normalizing flows *Proc. Natl Acad. Sci.* **119** e2109420119
[10] Grenioux L, Moulines Eric and Gabrié M 2023 Balanced training of energy-based models with adaptive flow sampling *Workshop on Structured Probabilistic Inference & Generative Modeling* (arXiv:2306.00684)
[11] Shixiang Shane G, Ghahramani Z and Turner R E 2015 Neural adaptive sequential Monte Carlo *Advances in Neural Information Processing Systems* p 28
[12] Hastings W K 1970 Monte Carlo sampling methods using Markov chains and their applications *Biometrika* **57** 97–109
[13] Hauschild J and Pollmann F 2018 Efficient numerical simulations with tensor networks: tensor network python (TeNPy) *SciPost Physics Lecture Notes* p 5 (available at: https://github.com/tenpy/tenpy)
[14] Kaiming H, Zhang X, Ren S and Sun J 2016 Deep residual learning for image recognition *Proc. IEEE Conf. on Computer Vision and Pattern Recognition* pp 770–8
[15] Hendrycks D and Gimpel K 2016 Gaussian error linear units (GELUs) (arXiv:1606.08415)

[16] Hibat-Allah M, Ganahl M, Hayward L E, Melko R G and Carrasquilla J 2020 Recurrent neural network wave functions *Phys. Rev. Res.* **2** 023358
[17] Kidger P and Garcia C 2021 Equinox: neural networks in JAX via callable PyTrees and filtered transformations *Differentiable Programming Workshop at Neural Information Processing Systems 2021* (arXiv:2111.00254)
[18] Kingma D P and Jimmy B 2014 Adam: a method for stochastic optimization (arXiv:1412.6980)
[19] Lange H, de Walle A V Abedinnia A and B Annabelle 2024 From architectures to applications: a review of neural quantum states *Quantum Sci. Technol.* **9** 040501
[20] Tuan Anh L, Igl M, Rainforth T Jin T, and Wood F 2017 Auto-encoding sequential Monte Carlo (arXiv:1705.10306)
[21] Jimmy Lei B, Kiros J R and Hinton G E 2016 Layer normalization (arXiv:1607.06450)
[22] McNaughton B, Milošević M V, Perali A and Pilati S 2020 Boosting Monte Carlo simulations of spin glasses using autoregressive neural networks *Phys. Rev. E* **101** 053312
[23] Medvidović M and Moreno J R 2024 Neural-network quantum states for many-body physics (arXiv:2402.11014)
[24] Metropolis N, Rosenbluth A W, Rosenbluth M N, Teller A H and Teller E 1953 Equation of state calculations by fast computing machines *J. Chem. Phys.* **21** 1087–92
[25] Müller T, McWilliams B, Rousselle F, Gross M and Novák J 2019 Neural importance sampling *ACM Trans. on Graphics (ToG)* **38** 1–19
[26] Nicoli K A, Nakajima S, Strodthoff N, Samek W, Müller K-R and Kessel P 2020 Asymptotically unbiased estimation of physical observables with neural samplers *Phys. Rev. E* **101** 023304
[27] Nomura Y and Imada M 2021 Dirac-type nodal spin liquid revealed by refined quantum many-body solver using neural-network wave function, correlation ratio and level spectroscopy *Phys. Rev. X* **11** 031034
[28] Orús R 2019 Tensor networks for complex quantum systems *Nat. Rev. Phys.* **1** 538–50
[29] Pfau D, Axelrod S, Sutterud H, von Glehn I and Spencer J S 2024 Accurate computation of quantum excited states with neural networks *Science* **385** eadn0137
[30] Pina-Otey S, Sánchez F, Lux T and Gaitan V 2020 Exhaustive neural importance sampling applied to Monte Carlo event generation *Phys. Rev. D* **102** 013003
[31] Raiaan M A K, Md S H M, Fatema K, Fahad N M, Sakib S J, Most M J M, Ahmad J, Ali M E and Azam S 2024 A review on large language models: architectures, applications, taxonomies, open issues and challenges *IEEE Access* **12** 26839–74
[32] Reh M, Schmitt M and Gärttner M 2023 Optimizing design choices for neural quantum states *Phys. Rev. B* **107** 195115
[33] Rende R, Viteritti L L, Bardone L, Becca F and Goldt S 2024 A simple linear algebra identity to optimize large-scale neural network quantum states *Commun. Phys.* **7** 260
[34] Roberts G O and Rosenthal J S 2007 Coupling and ergodicity of adaptive Markov chain Monte Carlo algorithms *J. Appl. Probab.* **44** 458–75
[35] Rubin D B 1987 The calculation of posterior distributions by data augmentation: comment: a noniterative sampling/importance resampling alternative to the data augmentation algorithm for creating a few imputations when fractions of missing information are modest: the sir algorithm *J. Am. Stat. Assoc.* **82** 543–6
[36] Schmitt M, Rams M M, Dziarmaga J, Heyl M and Zurek W H 2022 Quantum phase transition dynamics in the two-dimensional transverse-field Ising model *Sci. Adv.* **8** eabl6850
[37] Schollwöck U 2011 The density-matrix renormalization group in the age of matrix product states *Ann. Phys., NY* **326** 96–192
[38] Sharir O, Levine Y, Wies N, Carleo G and Shashua A 2020 Deep autoregressive models for the efficient variational simulation of many-body quantum systems *Phys. Rev. Lett.* **124** 020503
[39] Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez A N, Kaiser Łukasz and Polosukhin I 2017 Attention is all you need *Advances in Neural Information Processing Systems* p 30
[40] Vivas D R, Madroñero J, Bucheli V, Gómez L O and Reina J H 2022 Neural-network quantum states: a systematic review (arXiv:2204.12966)
[41] Xiong R, Yang Y, He Di, Zheng K, Zheng S, Xing C, Zhang H, Lan Y, Wang L and Liu T 2020 On layer normalization in the transformer architecture *Int. Conf. on Machine Learning* (PMLR) pp 10524–33