Hybrid ground-state quantum algorithms based on neural Schrödinger forging

Paulin de Schoulepnikoff,[1,2] Oriel Kiss [2,3,*] Sofia Vallecorsa [2] Giuseppe Carleo,[1] and Michele Grossi [2,†]

[1]*Institute of Physics, Ecole Polytechnique Fédérale de Lausanne (EPFL), CH-1015 Lausanne, Switzerland*
[2]*European Organization for Nuclear Research (CERN), Geneva 1211, Switzerland*
[3]*Department of Nuclear and Particle Physics, University of Geneva, Geneva 1211, Switzerland*

Entanglement forging based variational algorithms leverage the bipartition of quantum systems for addressing ground-state problems. The primary limitation of these approaches lies in the exponential summation required over the numerous potential basis states, or bitstrings, when performing the Schmidt decomposition of the whole system. To overcome this challenge, we propose a method for entanglement forging employing generative neural networks to identify the most pertinent bitstrings, eliminating the need for the exponential sum. Through empirical demonstrations on systems of increasing complexity, we show that the proposed algorithm achieves comparable or superior performance compared to the existing standard implementation of entanglement forging. Moreover, by controlling the amount of required resources, this scheme can be applied to larger as well as non-permutation-invariant systems, where the latter constraint is associated with the Heisenberg forging procedure. We substantiate our findings through numerical simulations conducted on spin models exhibiting one-dimensional rings, two-dimensional triangular lattice topologies, and nuclear shell model configurations.

I. INTRODUCTION

In recent years, significant advances have been made in simulating the static and dynamical properties of many-body quantum systems using variational algorithms. For instance, density matrix renormalization group methods based on matrix-product states [1–3], neural networks quantum states [4], equivariant neural networks [5], or kernels methods [6] have accurately computed the ground-state energy of spin systems [7], or also fermi systems, such as molecules [8] or nuclei [9]. While neural network quantum states represent wave functions using classical representations, we can also consider their quantum counterpart, where the wave function *Ansatz* takes the form of a parametrized quantum circuit [10], such as in the popular variational quantum eigensolver (VQE) [11]. Even if VQE has been successfully applied in various areas, such as chemistry [11–13], spin chains [14–17], or nuclei [18–21], it is still unclear whether VQE is a scalable algorithm. Hence, the optimization procedure becomes increasingly difficult [22] with the system size because of the presence of barren plateaus in the loss landscape [23]. It is consequently desirable to conceive variational quantum algorithms acting on a minimal number of qubits.

Although the VQE is already a hybrid algorithm, in the sense that it relies on classical resources to perform the optimization, we take a step forward and design an algorithm relying on both neural networks and quantum circuits. More specifically, we consider VQE based on entanglement forging (EF) [24], a circuit-knitting strategy that effectively performs a Schmidt decomposition of the variational quantum state and optimizes the two sub systems separately before reconstructing the entanglement classically. This procedure has the desirable properties of reducing the number of qubits while still reproducing the ground-state energy with high accuracy. It is similar in spirit to quantum-embedded density functional theory [25], where quantum resources are only used for the most challenging parts.

Besides computing ground-state energies, EF also allows practical heuristic simulations, notably in analyzing bipartite entanglement. This concept is fundamental in quantum mechanics, as its measurement provides an understanding of the behavior of strongly correlated systems [26]. For instance, bipartite entanglement has been used in condensed matter physics to study phenomena such as quantum phase transitions, topological order, and many-body localization [27,28]. Advances in experimental techniques have made it possible to measure entanglement entropy in a variety of condensed matter systems over the past few years, revealing insights into their underlying quantum properties [29].

The main contribution of this paper is a Schrödinger forging procedure using an autoregressive neural network (ARNN) [30,31]. This method combines the versatility of Schrödinger forging with controlling the computational resources required via the introduction of a cutoff. Generative neural networks have already been proposed for EF [32], but only in the context of Heisenberg forging, which requires

*Corresponding author: oriel.kiss@cern.ch
†michele.grossi@cern.ch

permutation symmetry of the two subsystems. Our method, however, does not require permutation symmetry between the two sub systems, making it a more versatile approach to solving ground-state problems using quantum computers. Moreover, our algorithm naturally includes a cutoff in the number of basis states, limiting the required number of quantum circuits.

This paper is structured as follows. We first introduce EF in Sec. II A, as well as two ways to tackle its scalability issue based on Monte Carlo sampling and neural networks. The main contribution of this paper is then proposed in Sec. II B as a third option. We conclude our work with numerical simulations in Sec. III, testing our hybrid architecture on various physical models, such as one-dimensional spin chains, spins on a triangular lattice with a random external field, and the nuclear shell model.

II. METHODS

The general strategy of variational algorithms for ground-state problems is to prepare a wave function *Ansatz* $|\psi\rangle$, using the variational principle

$$E_0 \leqslant \frac{\langle \psi | H | \psi \rangle}{\langle \psi \rangle} \qquad (1)$$

to approximate the ground-state energy E_0 of the Hamiltonian of interest H. The *Ansatz* can take the form of, e.g., a neural network [4] or a quantum circuit [11], while the variational parameters are usually optimized with, e.g., gradient-based methods. In the following, we will explore hybrid classical-quantum models aiming at describing a bipartite system with quantum circuits, while the entanglement between the partitions is forged classically.

A. Entanglement forging

The starting point of the EF procedure is to employ a Schmidt decomposition, a direct application of a singular value decomposition (SVD), to write a quantum state $|\psi\rangle$ of a bipartite $H = H_A \otimes H_B$ quantum system, with dimensions N_A and N_B, as

$$|\psi\rangle = U \otimes V \sum_\sigma \lambda_\sigma |\sigma\rangle_A |\sigma\rangle_B. \qquad (2)$$

In the above, U and V are unitaries, and $|\sigma\rangle_X \in \{0, 1\}^{N_X}$ and λ_σ are the corresponding Schmidt coefficient. The latter are positive, normalized $\sum_\sigma |\lambda_\sigma|^2 = 1$, and the number of Schmidt coefficients is called the Schmidt rank. We recall that the distribution of the Schmidt coefficients is related to the level of entanglement between the two subsystems, with the von Neumann entropy being calculated by

$$S_{vN} = -2 \sum_\sigma \lambda_\sigma^2 \log(|\lambda_\sigma|). \qquad (3)$$

Therefore, maximal entanglement is characterized by a uniform distribution, while minimal entanglement is characterized by a dirac delta.

The variational state is obtained by parametrizing U and V with two quantum circuits and considering the Schmidt coefficients as additional variational parameters. Following Eddins

et al. [24], the most direct way to compute the expectation values, called *Schrödinger forging*, is to directly insert the Schmidt decomposition, e.g., Eq. (2), into $\langle O \rangle = \langle \psi | O | \psi \rangle$. Assuming that the observable O admits a bipartition $O = O_A \otimes O_B$, the expectation value can then be expressed as

$$\langle \psi | O | \psi \rangle = \sum_{n=1}^{2^{N/2}} \lambda_n^2 \langle \sigma_n | U^\dagger O_A U | \sigma_n \rangle \langle \sigma_n | V^\dagger O_B V | \sigma_n \rangle$$
$$+ \sum_{n=1}^{2^{N/2}} \sum_{m=1}^{n-1} \lambda_n \lambda_m \sum_{p \in \mathbb{Z}_4} (-1)^p$$
$$\times \langle \phi_{\sigma_n, \sigma_m}^p | U^\dagger O_A U | \phi_{\sigma_n, \sigma_m}^p \rangle$$
$$\times \langle \phi_{\sigma_n, \sigma_m}^p | V^\dagger O_B V | \phi_{\sigma_n, \sigma_m}^p \rangle, \qquad (4)$$

where $|\phi_{\sigma_n, \sigma_m}^p\rangle = |\sigma_n\rangle + i^p |\sigma_m\rangle$, $\mathbb{Z}_4 = \{0, 1, 2, 3\}$, and all the bitstrings σ have been labeled with a number n (or m). Decompositions with equally sized subsystems are considered: $N_A = N_B =: N/2$. We note that this is not a strict requirement, but we use it to simplify the notations.

We remark that this involves an exponential sum in the system size. As such, two methods have been suggested to solve this scalability issue [24]. The first uses an unbiased estimator of $\langle \psi | O_A \otimes O_B | \psi \rangle$ that can be evaluated by importance sampling according to $\sim \lambda_n \lambda_m$. The second approach is to leverage permutation symmetry between the two subsystems, producing another EF scheme. Since it is defined at the operator level, we refer to it as *Heisenberg forging*. This approach has been further developed by Huembeli *et al.* [32] using ARNN. More details about Heisenberg forging can be found in Appendix A.

B. Schrödinger forging with generative neural networks

In this section, we present an approach to Schrödinger forging. The starting point is to remark that the Schmidt coefficients decay exponentially if the two subsystems are weakly entangled, as is the case in low-energy eigenstates of chemical and spin-lattice model Hamiltonians. By introducing a cutoff in the sum, it is therefore possible to improve the efficiency of the estimation while keeping a sufficiently low additive error. However, it requires a selection of a set of bitstrings among the $2^{N/2}$ total possibilities, which represents an open problem for EF. To this end, we propose to use generative models (more specifically, ARNN [33]) to select the best candidates. ARNN is a type of neural network architecture commonly used in time-series forecasting and sequence modeling tasks. The autoregressive property means that the output at a given time step is regressed on its own past values. In fact, autoregressive models predict the next value in a sequence based on the previous values in that sequence. The use of an ARNN is motivated by the fact that the Schmidt coefficients are normalized and can thus be interpreted as a probability density. Following Ref. [34], we propose an algorithm, which is summarized in Algorithm 1. We note that this approach shares some similarities with quantum-inspired genetic algorithms, see, e.g., Ref. [35]. The parametrized unitaries and the Schmidt coefficients are finally optimized with a gradient-descent-based algorithm. A summary of the entire algorithm is shown in Fig. 1.

ALGORITHM 1. Generation of the set of bitstrings.

Inputs:
    Cutoff k
Outputs:
    Set of k bitstrings
Initialize:
    Start with a random set A of k bitstrings
while the algorithm has not converged **do**
    1. Generate a set of bitstrings G with
        the ARNN
    2. Using the bitstrings σ in the set $A \cup G$,
        find their Schmidt coefficient λ_σ by solving the
        system of equations

$$\frac{\partial \langle H \rangle}{\partial \lambda_\sigma} = 0 \quad \text{with} \quad \sum_\sigma |\lambda_\sigma|^2 = 1$$

    3. Create the set A' composed of the bitstrings
        from $A \cup G$ with the k biggest λ_σ
    4. Train the ARNN such that it models $p(\sigma) \sim |\lambda_\sigma|^2$
    5. Update $A \leftarrow A'$
end while
return the set A of k bitstrings

First, we explain how to use an autoregressive neural network to efficiently identify the relevant bitstrings. Since the Schmidt coefficients are normalized as $\sum_\sigma |\lambda_\sigma|^2 = 1$, they can be interpreted as a probability density. The chain rule from probability theory can be used to write

$$|\lambda_\sigma|^2 \sim p(\sigma_A, \sigma_B) = \prod_i p((\sigma)_i | \{(\sigma)_j, \ j < i\}), \quad (5)$$

and the bitstring pairs, associated with λ_σ can be encoded by stacking the bitstrings of subsystem B at the end of the bitstrings of subsystem A,

$$\sigma = |\sigma_A, \sigma_B\rangle$$
$$= |(\sigma)_1, \dots, (\sigma)_{N/2}, (\sigma)_{N/2+1}, \dots, (\sigma)_N\rangle. \quad (6)$$

Note that here $(\sigma)_i$ denotes the ith bit of the bitstring σ.

Neural networks, and more particularly, autoregressive methods, are powerful tools to model such conditional densities [36] by generating elements sequentially conditioned on the previous ones. To build the autoregressive model, we consider a dense ARNN whose architecture is very similar to a dense feedforward neural network. The notable difference is that the weights are tridiagonal matrices, ensuring the autoregressive nature of the model. From the ARNN, the bitstrings can then be sampled directly and efficiently, as detailed in Appendix B.

Exploring the full space of bitstrings is exponentially difficult, motivating the use of machine learning techniques to select the basis states that contribute the most to the wave function. Inspired by the work of Herzog *et al.* [34], we introduce an algorithm whose primary objective is to bypass exploring the extremely large space of basis states. Starting from a random set of bitstrings A_0, the strategy consists of adding bitstrings generated according to the approximation of the $|\lambda_\sigma|^2$ modeled by the ARNN. Since the variational energy is quadratic with respect to the Schmidt coefficient λ_σ, at each iteration, they can be determined by solving the constrained linear equation system

$$\frac{\partial \langle H \rangle}{\partial \lambda_\sigma} = 0$$
$$\sum_\sigma |\lambda_\sigma|^2 = 1, \quad (7)$$

where the sum runs over the set $A \cup G$, with A being the current set of bitstrings while G is the set of bitstrings sampled by the ARNN. The first equation ensures that the forged wave function has minimal energy, while the second guarantees its normalization. In a second step, the current set A is updated by taking the k bitstrings with the highest Schmidt coefficients. The ARNN is finally trained to model $p(\sigma_A, \sigma_B) \sim |\lambda_\sigma|^2$ in a supervised way. These steps are iterated until convergence, which is reached when the current set A is stable and the loss of the ARNN is close to zero. The choice of the
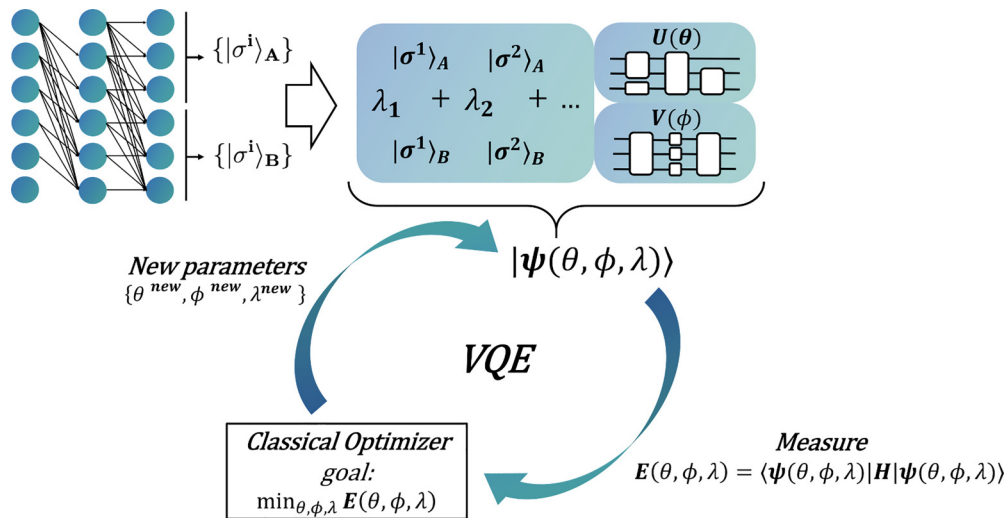

FIG. 1. Schema of the end-to-end algorithm. The set of bitstrings is first generated by the ARNN, and is then used to perform the Schrödinger forging VQE. This involves iteratively computing the variational energy on the quantum processing unit and classically optimizing the variational parameters until convergence.

TABLE I. Small models: number of bitstrings generated which are among the ones with the eight biggest Schmidt coefficients in the exact decomposition. The proposed algorithm is evaluated with $A \cup G$ or A' as training set $\mathcal{T}$, with or without MA. Furthermore, the ARNN if trained with the reversed KL, logcosh, and MMD loss. The final column shows the sum of the truncated Schmidt coefficients. The best results are highlighted in bold.

| | Standard approach | Proposed algorithm | | | | |
| | | logcosh | | | MMD | |
Models	reversed KL	$A \cup G$	A'	A' and MA	A'	$\sum_{k=0}^{7} \|\lambda_k\|^2$
TFIM 1D 14s	0/8	4/8	**7/8**	**7/8**	**7/8**	0.9641
Heis. 1D 14 spins	0/8	5/8	5/8	**7/8**	**7/8**	0.9605
J1J2 1D 14 spins	2/8	6/8	**7/8**	**7/8**	**7/8**	0.9163
TFIM 2D 12 spins	2/8	6/8	3/8	6/8	**7/8**	0.9842
tV (4 × 3)	2/8	5/8	5/8	4/8	**7/8**	0.9549

cutoff k is usually optimized by trial and error. Here, we start with a small cutoff, and slowly increase it until no further improvement is observed. We note that small cutoffs are often preferable, since they require less expensive calculations but also make the ARNN easier to train. This is why the ARNN plays an important role in choosing the optimal set of bitstrings. In summary, the training is composed of two stages: the training of the ARNN using some random initialization for U and V, followed by the optimization of the two unitaries in order to tailor them to the set of bitstrings. The ARNN is trained to model the distribution $p(\theta)$ on the target q obtained by solving the system of linear equations in Algorithm 1.

The choice of the loss function $\mathcal{L}$ and training set $\mathcal{T}$ play an important role in the training of the ARNN. For the training set, two possibilities are investigated: the model is either trained on the current set A and the generated bistrings G or, following Ref. [34], only on the non pruned bitstrings, i.e., the new set A'. Concerning the loss functions, we consider the explicit logcosh loss [37] and the implicit maximum mean discrepancy (MMD) loss [38]

$$
\mathcal{L}_{\text{MMD}}(\boldsymbol{\theta}) = \sum_{\sigma_1, \sigma_2 \in \mathcal{T}} q(\sigma_1) q(\sigma_2) K(\sigma_1, \sigma_2)
$$
$$
- 2 \sum_{\sigma_1, \sigma_2 \in \mathcal{T}} q(\sigma_1) p_{\boldsymbol{\theta}}(\sigma_2) K(\sigma_1, \sigma_2)
$$
$$
+ \sum_{\sigma_1, \sigma_2 \in \mathcal{T}} p_{\boldsymbol{\theta}}(\sigma_1) p_{\boldsymbol{\theta}}(\sigma_2) K(\sigma_1, \sigma_2), \quad (8)
$$

where $K(\sigma_1, \sigma_2) = e^{-\frac{\|\sigma_1 - \sigma_2\|_2^2}{2\Delta}}$ is chosen to be a Gaussian kernel, with $\| \cdot \|_2$ the 2-norm and Δ the bandwidth parameter. The latter determines the width of the kernel and controls the sensitivity of the MMD measurement. A larger bandwidth allows more global comparisons, while a smaller bandwidth focuses on local details. The MMD loss function effectively minimizes the difference between the mean embedding of the two distributions. It involves a pairwise comparison of every bitstring in the training set with their contribution being controlled by the kernel.

As a benchmark, we also consider a more standard approach for modeling probability distributions using the reversed Kullback-Leibler (KL) divergence for the loss of the

ARNN. A detailed description of this method is presented in Appendix D.

III. NUMERICAL SIMULATIONS

In this section, we present numerical experiments. We begin with the performance of the bitstrings selection algorithm on small models. Then, we proceed to expound upon the bitstrings selection and subsequent energy minimization process on various models of increasing complexity.

A. Identify the relevant bitstrings

We investigate the performance of the generative algorithm on small symmetric models: the transverse field Ising model (TFIM), the Heisenberg and J_1-J_2 model on a one-dimensional (1D) chain of 14 spins with periodic boundary condition, the two-dimensional (2D) TFIM on a 4×3 triangular lattice with a diagonal cut and open boundary condition, and the t-V model on a 4×3 grid. These models, further detailed in Appendix C, allow for an exact Schmidt decomposition, enabling us to assess the algorithm's performance by examining how many bitstrings associated with high Schmidt coefficients can be identified.

The results, for a cutoff dimension of $k = 8$ bitstrings, are presented in Table I. More specifically, we can find the performance of Algorithm 1 in terms of the number of correctly identified bitstrings using logcosh and MMD loss. Two training sets are considered for the former: the union $\mathcal{T} = A \cup G$ and the pruned set $\mathcal{T} = A'$. Furthermore, the impact of the parameters' initialization is attenuated by model averaging (MA). The ensemble technique consists of training four ARNNs with different initial weights and taking their average as the starting point of a final ARNN. Results obtained with the more standard reversed KL approach (see Appendix D) are also presented. Finally, the last column of the table contains the sum of the eight highest Schmidt coefficients squared from the exact decomposition. It indicates the amount of entanglement, the cutoff's accuracy, and the probability distribution's sharpness.

The algorithm proposed in this paper is able to find the majority of the most important bitstrings. Moreover, the MMD and logcosh loss function are superior to the standard approach based on the reversed KL divergence. Indeed, with the
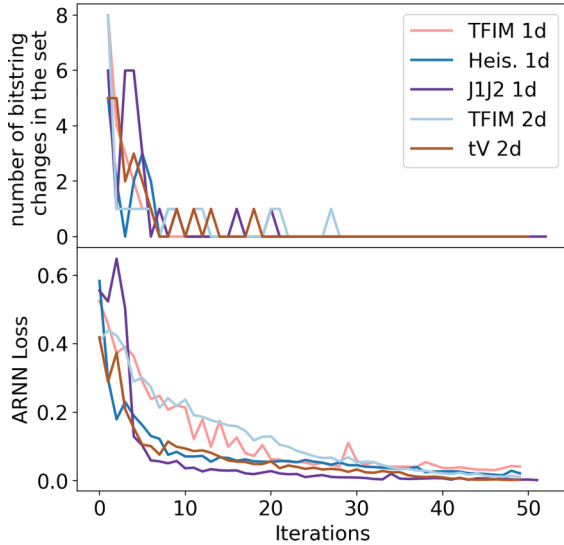
FIG. 2. Small models. Training of the generative algorithm for different physical systems. Top: the number of bitstrings updates between two consecutive iterations. Bottom: value of the MMD loss at each iteration.

latter, relevant bitstrings can only be found if the system is small or when the level of entanglement is high (leading to a wide probability distribution). This is not suitable in most applications, since low entanglement is important to guarantee a low additive error with a cutoff dimension. The best results are highlighted in bold, and are in general obtained with the MMD loss. More precisely, with the MMD loss, it is always able to find the four bitstrings with the highest Schmidt coefficient. This loss enables the ARNN to generalize well and make the algorithms converge quickly, as it can be further appreciated for the 2D TFIM with 12 spins in Appendix G, Fig. 17. In that case, the ARNN has only seen 24 bitstrings in total during the training and it is able to find the seven bitstrings with the highest Schmidt coefficients, containing the five most important ones, in only two iterations.

To gain a better understanding of the dynamics of the generative algorithm, the loss of the ARNN and the number of bitstring updates between two iterations are presented. Figure 2 shows the results with the MMD loss on the five different Hamiltonians. In all cases, we observe that the ARNN loss converges to zero and that the generated set of bitstrings is stable.

In general, the dynamics can be divided into two phases: an initial phase where the loss is high and the model explores a diverse range of bitstrings, followed by a second phase where the model attempts to exploit its approximation of the probability distribution to converge and generate bitstrings with high Schmidt coefficients. This exploration-exploitation trade-off can be modified by adjusting the number of bitstrings sampled at each iteration and the learning rate of the ARNN.

B. Complete entanglement forging scheme

In the last section, we shown that the ARNN is able to identify the bitstrings with the highest Schmidt coefficients.
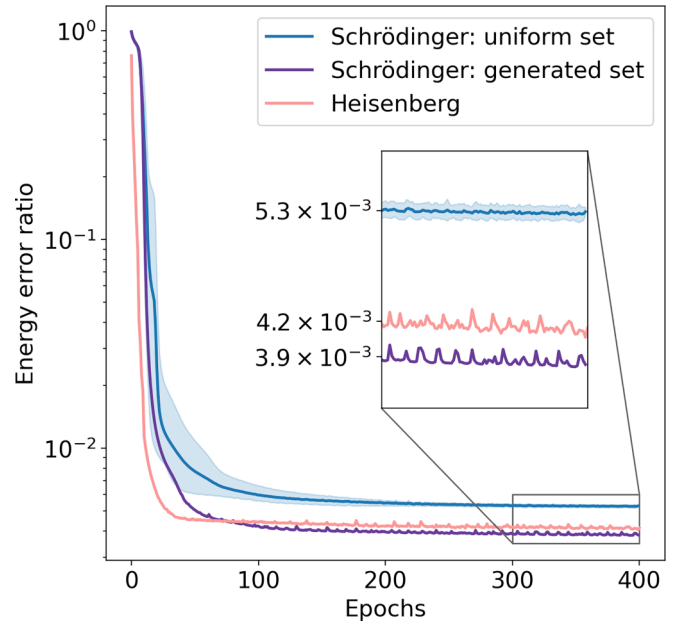

FIG. 3. One-dimensional TFIM 20 spins. Convergence of the variational energy of forged quantum states. The blue curve represents the mean energy over ten sets of $k = 8$ random bitstrings, with the shaded area displaying the standard deviation. The purple one is instead showing the training using the set generated by the ARNN. In addition, the simulation with the Heisenberg forging algorithm is shown in pink.

We now test the complete EF scheme. First, spin systems on a ring are considered, before going to a two-dimensional lattice, and finally to the nuclear shell model.

1. Spins in one dimension

We begin by considering the one-dimensional TFIM. More precisely, we consider a spin chain with periodic boundary conditions, an even number $N = 20$ of spins, and set the coupling and the external field coefficient to one. The Hamiltonian of the model can be written as

$$H = \sum_{i=1}^{N} Z^i Z^{i+1} + X^i. \tag{9}$$

Since the system is invariant under permutation symmetry, we can compare our approach to the Heisenberg forging with ARNN. Because of the symmetry, we can choose $\sigma_A = \sigma_B$, which reduces the number of possible bitstrings. As above, a cutoff dimension of $k = 8$ is chosen for the number of bitstrings, which was chosen by trial and error. Figure 3 shows the energy error ratio

$$\Delta = \left| \frac{E - E_{\text{exact}}}{E_{\text{exact}}} \right| \tag{10}$$

for the three forging schemes, i.e., Schrödinger with a random uniform set of bitstrings, Schrödinger with the generated set, and Heisenberg forging with the generated set. Following Ref. [32], a pretraining of the quantum circuit over 1000 iterations is performed. In both cases, the unitaries take the
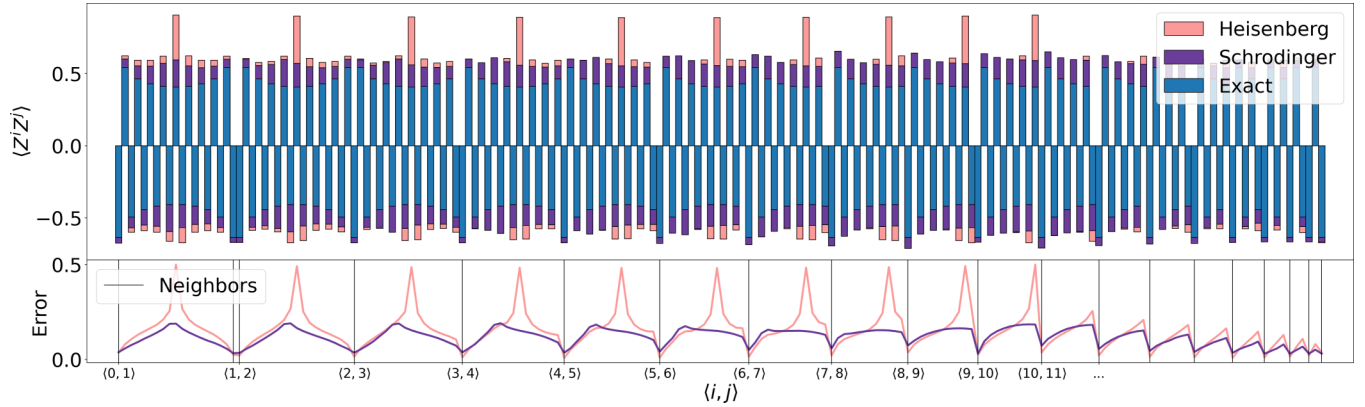
FIG. 4. Correlators in 1D. Correlators $\langle Z^i Z^j \rangle$ of the Schrödinger and Heisenberg forged states on the TFIM 20 spins in 1D. The pairs $\langle i, j \rangle$ are ordered as follows: $[[\langle i, j \rangle \text{ for } i < j] \text{ for } 0 \leqslant i < N]$. The neighboring cases, with $j = i + 1$, are highlighted with a black vertical line.

form of hardware efficient *Ansatz*

$$\mathcal{U}(\Theta) = \prod_{d=0}^{D-1} \left[U\left(\theta_d^0\right) \prod_{i=0}^{N/2} \mathrm{CX}_{i,i+2} \, U\left(\theta_d^1\right) \prod_{i=1}^{N/2} \mathrm{CX}_{i,i+2} \right] U(\theta_D),$$
(11)

where $D = 15$ is the number of layers, $\mathrm{CX}_{i,j}$ is a CNOT gate with control qubit i and target j, while $U(x)$ is N-fold tensor product of arbitrary single-qubit rotation parametrized with $3N$ parameters. We denote with Θ the set containing all indexed θ_i^j. Details on the training procedure, such as values for the hyperparameters and the optimization algorithm, can be found in Appendix E.

We observe that the choice of the random set has little impact on the performance of the Schrödinger forging procedure. Moreover, the models enhanced with the ARNN display better results, both being quite similar.

To ensure that specific physical properties of the ground state, outside of its energy, are correctly reproduced, the spin-spin correlators $\langle Z^i Z^j \rangle$ of the forged states have been calculated. They are shown in Fig. 4. We observe that the accuracy is not degrading over the overlap, suggesting that the error can be explained mainly by the training of the circuits rather than the EF procedure. The error on the correlators $\langle Z^i Z^j \rangle$ is minimal when $j = i + 1$ and maximal if the two spins are far apart in the chain. This can be explained by the locality of the *Ansatz*, built using gates acting on neighboring qubits.

2. Spins in two dimensions

We now move toward two-dimensional spin lattices, which are more challenging due to local operators being mapped to nonlocal ones when projected onto a line. We consider the TFIM on a 2D topology described by a triangular lattice, as shown in Fig. 13, see Appendix C. We break the permutation symmetry of the two subsystems by applying a random external field $h_i \sim U[-1, 1]$. Setting the coupling constant to one, the Hamiltonian is given by

$$H = \sum_{\langle i,j \rangle} Z^i Z^j + \sum_{i=0}^{N-1} h_i X^i,$$
(12)

where $\langle i, j \rangle$ are neighbors according to the triangular topology. The triangular lattice has a high coordination number, leading to strong magnetic susceptibility [39], meaning that the system is more sensitive to external magnetic fields and can therefore exhibit stronger magnetic order and complex physical phenomena, such as, e.g., disorder, localization, and heterogeneity.

The two-dimensional lattice is divided with a cut along the diagonal axis. We consider open boundary conditions (OBC), cylindrical boundary conditions (CBC), and toroidal boundary conditions (TBC). Since the boundary conditions can lead to different levels of entanglement [40,41], they play an essential role in the EF procedure, which is why different configurations are considered.

The convergence of the variational energies for the three boundary conditions are shown in Fig. 5. Like in the one-
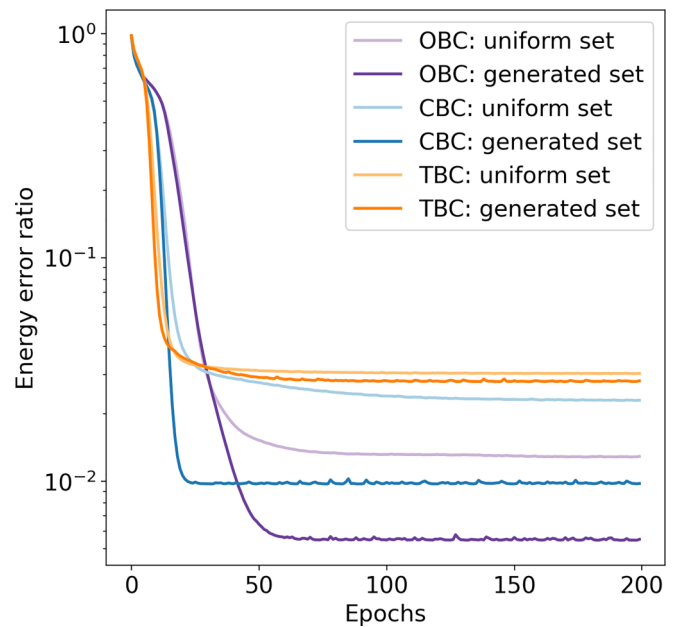

FIG. 5. Two-dimensional TFIM 12 spins. Convergence of the variational energy of forged quantum states. The colors indicate different boundary conditions, while the shaded curves show the mean energy over ten sets of $k = 8$ random uniform bitstrings.

dimensional case, a cutoff of $k = 8$ is chosen in the Schmidt decomposition. We observe that the bitstrings generated by the ARNN lead to an improvement of approximately 10^{-2} in the error energy ratio with respect to taking a random set. The most striking result, though, is that the gap between the random and generated methods is increasing with respect to the one-dimensional case, suggesting that sampling with the ARNN is becoming more effective when considering systems of increased complexity. On the other hand, no advantage can be noted in the context of TBC. It seems that the parametrization of the unitaries is the limiting factor in improving the energy error.

3. Nuclear shell model

Finally, we consider light nuclei in the shell model with Cohen-Kurath [42] interactions, where the Hamiltonian can be written in second quantization as

$$H = \sum_i \epsilon_i \hat{a}_i^\dagger \hat{a}_i + \frac{1}{2} \sum_{ijkl} V_{ijkl} \hat{a}_i^\dagger \hat{a}_j^\dagger \hat{a}_k \hat{a}_l. \qquad (13)$$

Here, $\hat{a}_i^\dagger$ and $\hat{a}_i$ are the creation and annihilation operators, respectively, for a nucleon in the state $|i\rangle$. Single-particle energies are denoted as ϵ_i and two-body matrix elements as V_{ijkl}. The orbitals $|i\rangle = |n = 0, l = 1, j, j_z, t_z\rangle$ are described as functions of the radial n and orbital angular momentum l, the total spin j, its projection on the z axis j_z its projection, and the z projection of the isospin t_z.

We consider nucleons in the p shell model space, which includes six orbitals for the protons and six orbitals for the neutrons, while each energy is computed with respect to an inert ^{4}He core. The shell-model Hamiltonian [see Eq. (13)] is converted into a qubit Hamiltonian via the Jordan-Wigner [43] transformation. Each single-particle state is represented by a qubit where $|0\rangle$ and $|1\rangle$ refer to an empty and an occupied state, respectively. Therefore, each nucleus can be distinguished by the number of excited orbitals, representing the protons and neutrons on top of the ^{4}He core.

The partition is made at the isospin level, meaning that subsystem A consists entirely of protons while subsystem B consists of neutrons. Therefore, the Schrödinger forging is the only possible choice since the system is not symmetric under proton-nucleon exchange. To build a chosen nuclei, we start from an appropriate initial state, with the desired number of nucleons, and act with an excitation preserving (EP) *Ansatz*. EP *Ansätze* can be built as a product of two-qubit excitation preserving blocks $U(\theta, \phi)$, also known as hop gates [44,45], of the form

$$U(\theta) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\theta) & \sin(\theta) & 0 \\ 0 & \sin(\theta) & -\cos(\theta) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}. \qquad (14)$$

This set can be extended with four-qubit excitation-preserving gates [46], defined as

$$G_{i,j,k,l}(\omega) |0011\rangle = \cos(\omega/2) |0011\rangle + \sin(\omega/2) |1100\rangle$$
$$G_{i,j,k,l}(\omega) |1100\rangle = \cos(\omega/2) |1100\rangle - \sin(\omega/2) |0011\rangle . \qquad (15)$$
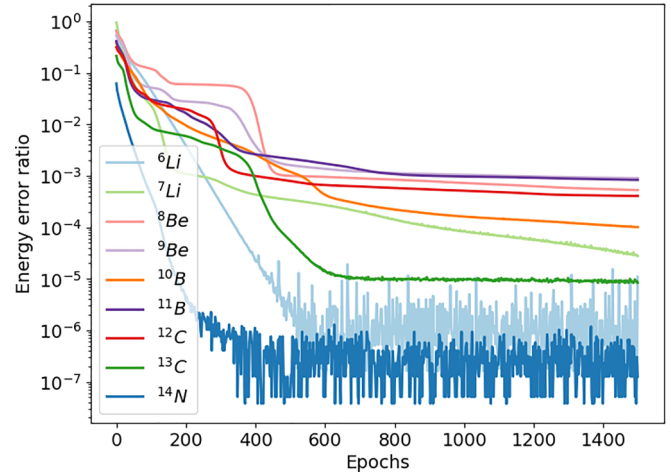

FIG. 6. Convergence of the variational energy of the Schrödinger forged states corresponding to the various nuclei of the nuclear p shell model. The Schmidt rank being at most 20, all bitstrings have been used in the Schmidt decomposition.

The parametrized circuit then takes the form of a layered *Ansatz* composed of a product of excitation-preserving gates, where the dth layer is described by

$$\mathcal{U}(\Theta_d) = \left(\bigotimes_{i=0}^{N-1} \mathrm{RZ}_i(\phi_d^i) \right) \prod_{i=0}^{N/2-2} U_{2i,2i+1}(\theta_d^i)$$
$$\times \prod_{i=1}^{N/2-3} U_{2i,2i+1}(\theta_d^i) \prod_{i=0}^{N-4} G_{i:i+3}(\omega_d^i). \qquad (16)$$

We denote by $\mathrm{RZ}_i(\phi)$ a rotation of the ith qubit around the z axis, and the subscript of the U and G gates indicate the qubit the gate is acting upon ($i : j$ is a slice from i to j). The large Θ_d parameters regroup all parameters in the dth layer, i.e., $\Theta_d = \{\phi_d^i, \theta_d^i, \omega_d^i\}$. A sketch of the quantum circuit is depicted in Appendix F.

Since the Schmidt rank of the p nuclear shell model is at most 20, the generative algorithm is unnecessary, as all bitstrings in the Schmidt decomposition can be used. The energy minimization for the various nuclei is presented in Fig. 6. We observe that every ground-state energy in the p shell can be reproduced with an error ratio of at most 10^{-3}, even for the difficult nuclei such as $12C$. Moreover, having access to the Schmidt decomposition allows us to evaluate the von Neumann entropy, whose evolution is presented in Fig. 7, which can be of broader interest. Figure 7(a) shows the evolution of the von Neumann entropy during the training, while Fig. 7(b) displays a visualization of the von Neumann entropy in the parameter space. To this end, a principal component analysis (PCA) is performed on the entire history of the Schmidt coefficient, and a scan of the entropy along the two main components is presented. In addition, the entropy value is shown for each training epoch (in gray) and the final value (in red).

In the final experiment, nucleons in the sd shell model space, including 12 orbitals for the protons and 12 orbitals for the neutrons, are considered. For the latter, each energy is computed with respect to an inert 16O core. Using the Jordan-
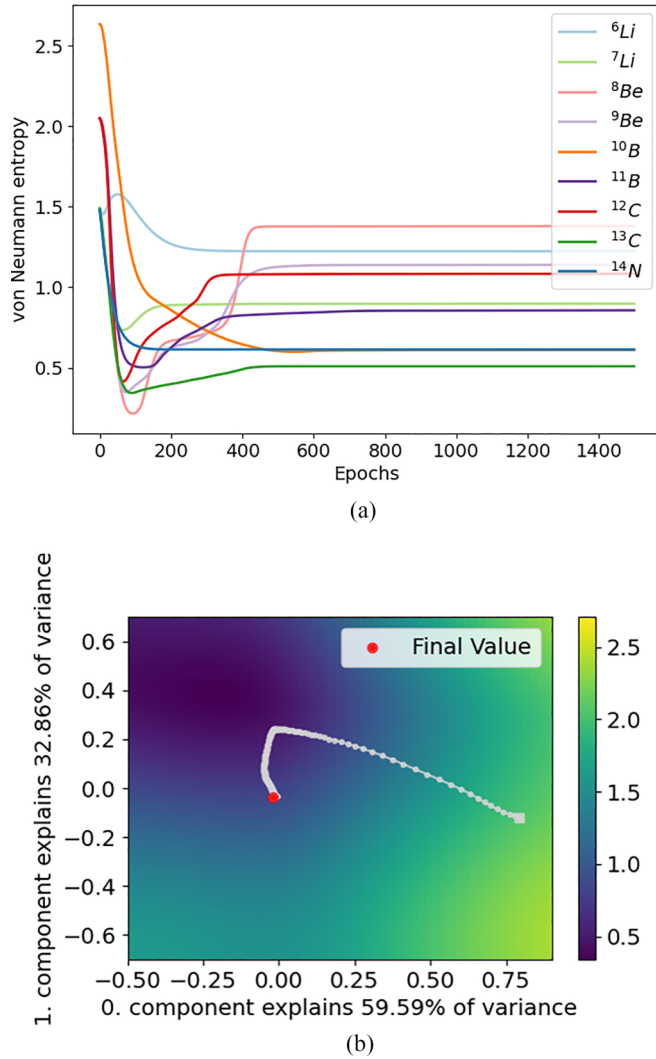
(a)

(b)

FIG. 7. (a) Von Neumann entropy of the various nuclei during the training. (b) Visualization of the two main components of the Von Neumann entropy of the ^{11}B in the variational space. In addition, the value of the entropy at each training epoch is shown (in gray) as well as the final value (in red).

Wigner mapping, this model leads to a 24-qubit Hamiltonian and is composed of a total of 11 210 overlapping terms. This high number can make EF particularly expensive, as it scales linearly with it. However, since most of their coefficients are close to zeros, an approximate Hamiltonian, consisting of the 38 overlapping terms with the most significant coefficients, is instead considered. Despite this approximation, the Hamiltonian can still reproduce 97% of the ground-state energy of the ^{23}Na nucleus, which is the focus of this experiment.

The ^{23}Na nucleus is composed of three protons and four neutrons on top of a 16O inert core. The ARNN sampler has therefore been modified to generate bitstrings with three ones in subsystem A (protons) and four ones in subsystem B (neutrons). The energy minimization and the final Schmidt decomposition of the ^{23}Na are presented in Figs. 8 and 9, respectively. Once again, a higher accuracy is obtained with the generated set. Multiple states from the generated set are contributing to the VQE, meaning that the ARNN is useful
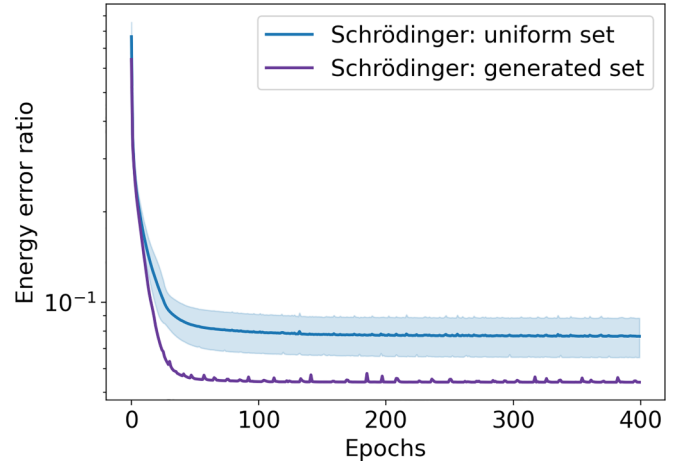

FIG. 8. Convergence of the variational energy of the Schrödinger forged states corresponding to the ^{23}Na nucleus of the nuclear *sd* shell model.

in selecting appropriate bitstrings. On the contrary, when the random set is used, the variational circuit prefers to adapt to one state, and set the contribution from the others to zero.

IV. DISCUSSION AND CONCLUSION

This paper proposes an alternative way to perform Schrödinger forging using autoregressive neural networks. We build on the work from Eddins *et al.* [24], which introduced the EF-based VQE, and on Huembeli *et al.* [32], which efficiently compute quantum expectation values as statistical expectation values over bitstrings sampled by a generative neural network. While their work leverages the additional permutation symmetry, our work is fully general and computationally efficient due to the introduction of a cutoff dimension. Moreover, the latter is giving us additional control over the amount of quantum resources required. This is not the case in the Heisenberg forging scenario, as shown in Appendix B, where the ARNN begins by sampling many
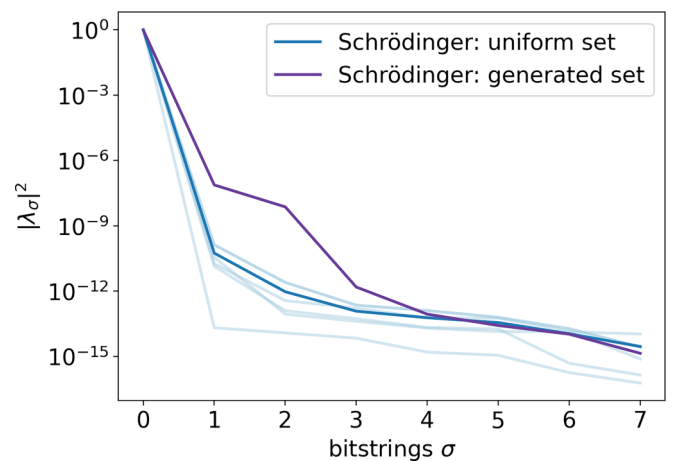

FIG. 9. Final Schmidt decomposition of the variational energy of the Schrödinger forged states corresponding to the ^{23}Na nucleus of the nuclear *sd* shell model.

bitstrings and finishes by using only one. Therefore, in this specific case, the Heisenberg forging with neural networks is expensive at the beginning of the training and loses its expressive power at the end. On the other hand, Schrödinger forging enables better control on the trade-off between expressiveness and computational expensiveness of the variational model without having the assumption of symmetric permutation of the two subsystems. When the additional permutation symmetry is present, we still recommend using Heisenberg forging, since it requires less epochs to be trained. However, we stress that many systems, such as molecules or nuclei, do not exhibit this symmetry, providing important use cases for Schrödinger forging.

Numerical simulations have been performed on ring and triangular lattice spin systems. Schrödinger forging with the ARNN consistently achieves better performance for the computation of the ground-state energy and correlators, compared with random sampling and Heisenberg forging with neural networks. In the case of the triangular lattice, different boundary conditions are considered, directly affecting the performance. The parametrization of unitaries is a limiting factor when complex boundary conditions are considered.

The most striking result is that the performance gap between random sampling and using the ARNN increases with the system's complexity, thus suggesting that our approach will be more profitable for larger systems. Finally, the nuclear shell model is also solved using the Schrödinger forging case up to the 10^{-3} error ratio for the most complex nucleus. The ARNN is unnecessary since the maximum number of possible bitstrings is 20, as all bitstrings can be used. The approach is then tested on a larger nucleus in the *sd* shell model, ^{23}Na. Once again, the generated set results in better accuracy than a random one.

Autoregressive models are easily interpreted and can naturally generate bitstrings with a certain number of excitations. They are also well suited for addressing the task at hand, owing to their robustness as density estimators. They do exhibit certain limitations, specifically in terms of sampling speed and the requirement for fixed-order input decomposition [47]. Nevertheless, the limited number of samples in this algorithm renders the issue of sampling speed inconsequential. Furthermore, experiments were conducted by varying the decomposition orders, and it was determined that such alterations did not yield any substantial changes in the obtained results. Masked multilayer perceptrons are a straightforward choice for building the autoregressive model. However, other architectures could be more suitable in some cases. In particular, transformers [48] provide a strong alternative since they are highly parallelizable and efficient in capturing the global context and long-range dependencies due to their attention mechanism.

At the beginning of this work, simulations were carried out on small models. In these cases, all bitstrings could be taken into account during the VQE. It was observed that the order of the bitstrings, with respect to their coefficient (in absolute value), did not significantly change during the VQE. Therefore, choosing the bitstrings at the beginning of the circuit training enables us to perform well. However, it could be suitable in some cases to train the quantum circuits and ARNN simultaneously, taking advantage of parameter sharing.

Finally, we note that there is not necessarily a correlation between having sets of bitstrings associated with high Schmidt coefficients and the trainability of the corresponding variational state. Indeed, in some cases, taking a set of bitstrings with lower Schmidt coefficients might be favorable to make the variational circuits easier to train. Therefore, it may be possible to include this feature in the algorithm by choosing bitstrings that maximize the gradients. Alternatively, an algorithm, adapting the form of each variational circuit, an approach close to the ADAPT-VQE [49], could be investigated.

The numerical simulations of the quantum circuits have been performed with PENNYLANE [50], powered by a JAX backend [51], while the NETKET library [52] has been used for the ARNN. Solving the constraint systems of equations in the generative algorithm involves a projected gradient descent algorithm available in the JAXOPT library [53]. Moreover, the Heisenberg forging code is available on GITHUB [54]. Visualization of the evolution of the von Neumann entropy is performed using ORQVIZ [55]. The PYTHON code of this project is accessible on GITHUB [56].

APPENDIX A: HEISENBERG FORGING

In the section, we briefly cover the basis of Heisenberg forging, covered in more detail in Refs. [24,32]. In this scenario, we assume a symmetric bipartition, i.e., with $U_A = V_B \equiv U$, and find a more efficient way to compute the expectation value. We first need to decompose O as

$$O_A \otimes O_B + O_B \otimes O_A = \frac{a_0}{2}(\{O_A, O_B\} \otimes \mathbb{1} + \mathbb{1} \otimes \{O_A, O_B\})$$
$$+ \sum_{\alpha, \beta \in \{0,1\}} a_{\alpha,\beta} C^*_{\alpha,\beta} \otimes C_{\alpha,\beta}, \quad (A1)$$

where $\{\cdot, \cdot\}$ denotes the anticommutator, $|a_{\alpha,\beta}| \leqslant 1$ are real coefficients, and $C_{\alpha,\beta}$ n-qubit Clifford operators that are defined below. Combining this with the Schmidt decomposition, and by symmetrizing the observable, we obtain

$$\langle \psi | O | \psi \rangle = a_0 \sum_n \lambda^2_{\sigma_n} \Re(\langle \sigma_n | U^\dagger O_A O_B U | \sigma_n \rangle)$$
$$+ \sum_{\alpha, \beta \in \{0,1\}} \frac{a_{\alpha,\beta}}{2} \sum_{n,m} \lambda_{\sigma_n} \lambda_{\sigma_m} |\langle \sigma_m | U^\dagger C_{\alpha,\beta} U | \sigma_n \rangle|^2.$$
$$(A2)$$

Since $O_A, O_B \in \{\mathbb{1}, X, Y, Z\}^{\otimes N/2}$, we either have $[O_A, O_B] = 0$ or $\{O_A, O_B\} = 0$. In the first case, we can find a Clifford circuit V such that $O_A = V Z_p V^\dagger$ and $O_B = V Z_q V^\dagger$. We can then define

$$C_{\alpha,\beta} = \tfrac{1}{2} V(\mathbb{1} + (-1)^\alpha Z_p + (-1)^\beta Z_q - (-1)^{\alpha+\beta} Z_p Z_q)V^\dagger.$$
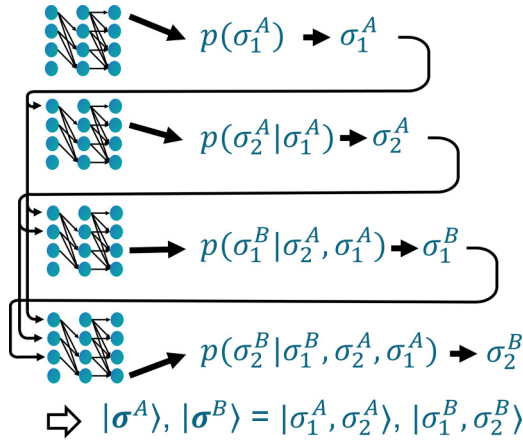$$(A3)$$

FIG. 10. Description of the sampling procedure of a bitstring with the ARNN. Each bit of the bitstring is generated sequentially and randomly, according to the probability modeled by the network. The latter is the output of the ARNN with the inputs being the values of the previous bit, already generated. The illustrated situation corresponds to a system of four qubits divided equally into two subsystems of two qubits. Note that the picture describes in fact one ARNN used in parallel to sample the different bits, and not multiples ARNN used in a sequential manner, as could be induced by the arrows in the diagram.

In the remaining case, we can simply use $C_{0,0} = (O_A + O_B)/\sqrt{2}$, $C_{0,1} = (O_A - O_B)/\sqrt{2}$ and $a_{1,0} = a_{1,1} = 0$.

The estimation of the sums can be performed nontrivially, using Monte Carlo sampling, where the number of samples grows as $1/\epsilon^2$, with ϵ the additive error. However, the sampling step is not obviously scalable in the Schrödinger case, and it will be discussed below. We also point out that, despite a sampling overhead, the individual quantum circuits are easier to implement than without the Schmidt decomposition because they are shallower and require fewer qubits.

APPENDIX B: SAMPLING FROM THE ARNN

In this section, we provide more details on how to efficiently and directly sample the bitstrings with the ARNN. We proceed recursively, as shown in Fig. 10. We begin by sampling the first bit of the string, which is then given as an input to sample the second bit, and so on up to the last bit.

In the nuclear shell model, it is important to control the number of value-one bits appearing in the string, since each nucleus is defined by a certain amount k of excited orbitals. Thus, the same procedure can be slightly modified to generate bitstrings with a fixed number of ones. Indeed, we just need to change the conditional probability in the sampling procedure, which can be done by setting $p(\sigma_i|\{(\sigma)_j, \ j < i\}) = 0$ if $\sum_{j<i}(\sigma)_j = k$. This ensures a maximum of k excitation. If, on the other hand, there is only $l < k$ excitation at the end of the string, the last $k - l$ bits are turned into one to correct for it. While this leads to nonuniform sampling at the beginning of the training, we expect the ARNN to overcome this issue by incorporating it through the learning stage.

A notable difference between the Schrödinger and Heisenberg forging schemes is that for the latter, it is impossible to control how many states one has to prepare on the quantum
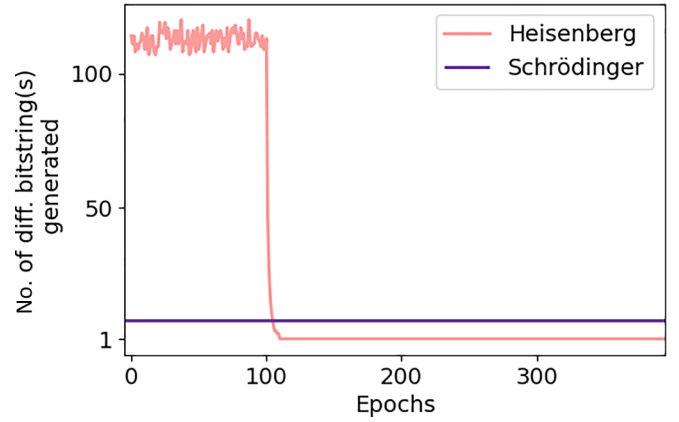

FIG. 11. Comparison between the number of different bitstrings sampled for Heisenberg and Schrödinger forging. In the Heisenberg case, at the beginning, a large number of states are required to be prepared on the quantum computer, while at the end, only one state remains. For Schrödinger forging instead, we have full control over the number of states we want to prepare.

hardware. Indeed, in this case, all bitstrings sampled by the ARNN must be taken into account. In practice, as shown in Fig. 11, many states must be prepared at the beginning of the training and only one at the end. In the case of Schrödinger forging, since the cut-off can be fixed at the beginning, the number of states to be prepared on the hardware is constant. Following Ref. [32], a 1000-epoch pretraining on the unitaries has been performed as proposed in Ref. [32]. However, other optimization strategies could be considered.

APPENDIX C: OVERVIEW OF THE MANY-BODY HAMILTONIANS OF THE SMALL MODELS

Here, we present the many-body quantum Hamiltonians used for the numerical simulations. First, we consider spin models: the TFIM, Heisenberg, and J_1-J_2 model on a 1D chain and the 1D TFIM on a triangular lattice. We also consider fermionic models, such as the t-V model on a 4×3 grid and the nuclear shell model.

The Hamiltonians of the 1D TFIM is

$$H = J \sum_{i=0}^{N} Z^i Z^{i+1} + X^i. \tag{C1}$$

The Hamiltonian of the 1D Heisenberg model is

$$H = J \sum_{i=0}^{N} X^i X^{i+1} + Y^i Y^{i+1} + Z^i Z^{i+1}, \tag{C2}$$

while for the 1D J_1-J_2 model we have

$$H = J_1 \sum_{i=0}^{N} X^i X^{i+1} + Y^i Y^{i+1} + Z^i Z^{i+1}$$

$$+ J_2 \sum_{i=0}^{N} X^i X^{i+2} + Y^i Y^{i+2} + Z^i Z^{i+2}. \tag{C3}$$

For these models, $J = 1$, $J_1 = 1$, $J_2 = 0.2$, and periodic boundary condition (PBC), i.e., $N \equiv 0$, $N + 1 \equiv 1$, are used.
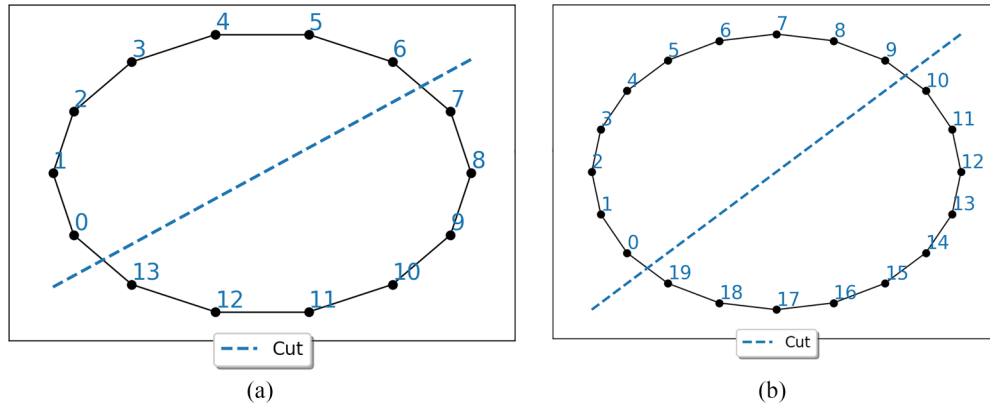
(a)   (b)

FIG. 12. One-dimensional spin chain with PBC, $N = 14$ spins (a) and $N = 20$ spins (b). The blue cut represents the separation between the two subsystems.

The topology of the spin chain with the separation between the two subsystems is presented in Figs. 12(a) and 12(b) for 14 and 20 spins, respectively.

The Hamiltonian of the 2D TFIM is given by

$$H = \sum_{\langle i,j \rangle} Z^i Z^j + \sum_{i=0}^{N-1} X^i, \qquad (C4)$$

where $\langle i, j \rangle$ are neighbors according to the triangular topology, see Fig. 13, which also shows the different cuts and boundary conditions. This model is more challenging due to local operators being mapped to nonlocal ones when projected onto a line. Moreover, it has a high coordination number, which leads to a strong magnetic susceptibility [39], meaning that the system is more sensitive to external magnetic fields and can exhibit stronger magnetic order.

The Hamiltonian of the t-V model is

$$H = -t \sum_{\langle i,j \rangle} (a_i^\dagger a_j + a_j^\dagger a_i) + V \sum_{\langle i,j \rangle} a_i^\dagger a_i a_j^\dagger a_j, \qquad (C5)$$

with a_i and $a_i^\dagger$ being, respectively, the creation and annihilation operators on site i. A 4×3 system of spinless fermions with periodic boundaries and $t = V = 1$ is considered. It is mapped to a qubit Hamiltonian with the Jordan-Wigner transformation. In this model, fermions are allowed to move on the grid, modifying the energy of the system. In this spinless version, there is only one spin orbit per site, giving a final Hamiltonian of 12 qubits.

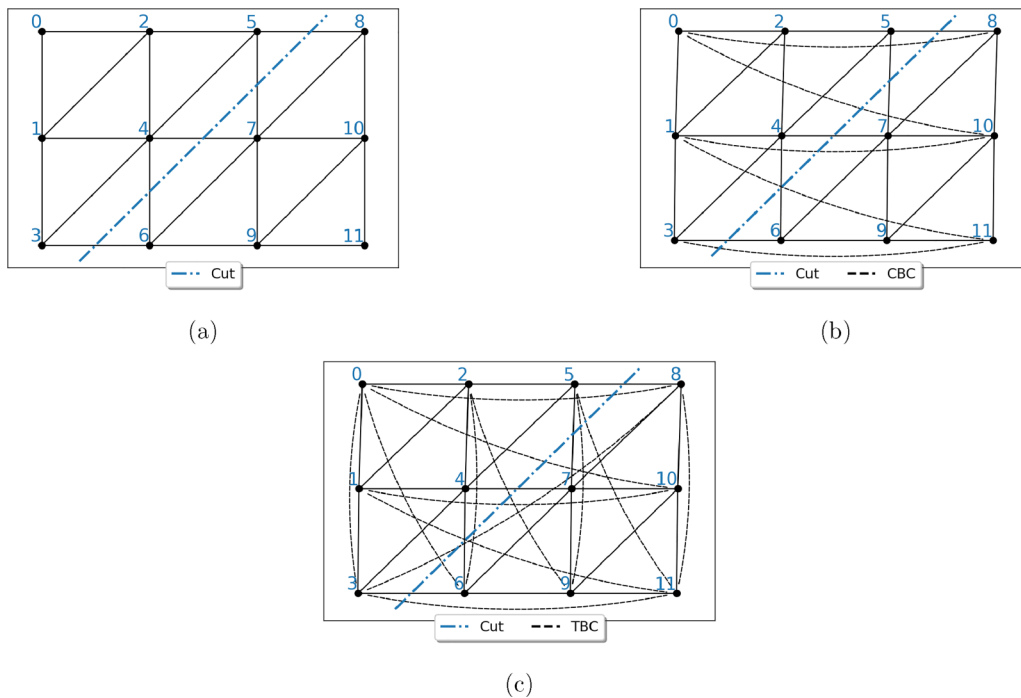(a)   (b)

(c)

FIG. 13. Triangular lattices used for the simulations. Lattices of 12 spins with OBC, CBC, and TBC are shown in (a), (b), and (c), respectively. The two subsystems are defined with a diagonal cut (blue).
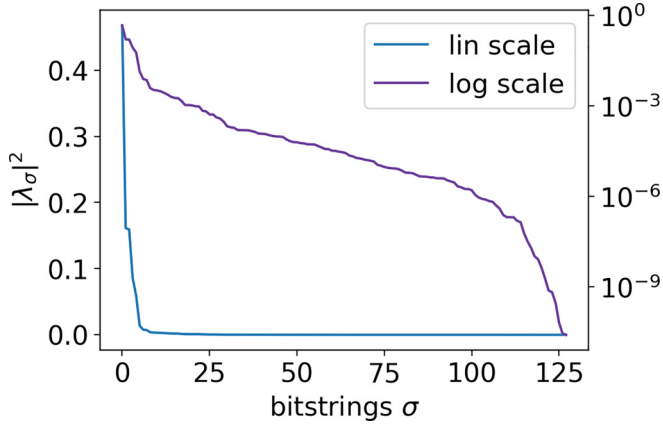
FIG. 14. Exact Schmidt decomposition on the TFIM 14 spins with periodic boundary conditions. The coefficients have been arranged in descending order and are presented with a linear (blue) and a logarithmic (purple) scale.

APPENDIX D: MODELING THE PROBABILITY DISTRIBUTION WITH A MORE STANDARD APPROACH

In this section, we present a more standard approach for modeling probability distributions using the reversed KL divergence for the loss of ARNN. We show why it is unsuitable for the considered problem situation.

This approach aims to model the full probability distribution $|\lambda_\sigma|^2$. Since we only have samples from the approximated probability distribution $p(\sigma_A, \sigma_B)$, the reversed KL can be used to learn the best representation of the distribution self-consistently. Thus, at each iteration, the training set comprises bitstrings sampled from the approximation distribution given by the ARNN. The latter is then trained in a supervised way to model the target distribution $|\lambda_\sigma|^2$ by minimizing the reversed KL divergence

$$\mathcal{L}_{\text{ARNN}}^{\text{rev-KLD}} = \mathbb{E}_{\sigma \sim p}\left[\log \frac{p(\sigma_A, \sigma_B)}{\lambda_\sigma^2} \right]. \qquad \text{(D1)}$$

With this choice, wherever $p(\sigma_A, \sigma_B)$ has a high probability, λ_σ^2 will also take a high value. This mode-seeking behavior is desired since the objective is mainly to sample bitstrings associated with a high Schmidt coefficient.

With weakly entangled systems, which is desirable to have a low additive error with the cut-off in the Schmidt decomposition, the target probability distribution is very sharp, as shown in Fig. 14. Such probability densities are very difficult to model with this approach. Indeed, with high probability, the training sets are composed of bitstrings associated with very small Schmidt coefficients. In this flat region (left of Fig. 14), the probability density appears uniform, and it is challenging to extrapolate the relevant bitstrings. Moreover, due to the normalization constraint, the Schmidt coefficients of a small set of bitstrings are not good estimators of the Schmidt coefficients of the ground truth distribution.

However, this defeats our purpose of identifying bitstrings with a high Schmidt coefficient rather than modeling the entire probability distribution. Hence, adopting a training strategy that keeps the bitstrings with high Schmidt coefficients through the iterations is convenient. With such a

training strategy, employing a loss composed of an average over the model data samples is impossible. The explicit form of the reversed KL divergence

$$\mathcal{L}_{\text{ARNN}}^{\text{expl-rev-KLD}} = \sum_\sigma p(\sigma_A, \sigma_B)\left[\log \frac{p(\sigma_A, \sigma_B)}{\lambda_\sigma^2} \right] \qquad \text{(D2)}$$

would be an alternative if only the target probability distribution is not very sharp. Hence, the reversed KL divergence is not a symmetric measure. Consequently, the gradients obtained from the reversed KL divergence may not provide stable and robust updates for the model when the predicted distribution diverges significantly from the target distribution. This lack of robustness makes it challenging to learn in highly uncertain situations or when the model needs to adapt to changes in the training set, which is the case here. The log-cosh and MMD loss were therefore used since they are more robust and suitable for modeling sharp distributions. Indeed, they do not suffer the same limitations since they focus on individual samples rather than the overall distribution and do not overemphasize outliers.

APPENDIX E: OPTIMIZATION DETAILS AND HYPERPARAMETERS

In this section, details on the optimization procedure are given. During the VQE stage of the training, the adabelief optimizer [57] is used to update the quantum circuit parameters, while Nesterov's accelerated gradient descent scheme [58] is performed for the Schmidt coefficient. One iteration of the Schmidt coefficient is done every ten iterations of the circuit parameters. The hyperparameters of the adabelief optimizers, following the convention of the original paper, are $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\epsilon = 10^{-16}$, while for Nesterov, a momentum coefficient of 0.6 is used. In both cases, we set the learning rate between 0.1 and 0.01.

The generative algorithm is trained using adabelief with the same hyperparameters and a learning rate of 0.001. The ARNN comprises five hidden layers and a hidden neuron density of $\alpha = 2$. At each iteration of the generative algorithm, the ARNN samples between 10 and 50 bitstrings to build the set G, while the exact value has been manually tuned for each simulation. This influences the performance since low values cause the ARNN to converge very quickly, leading to spikes caused by the lack of generalization and
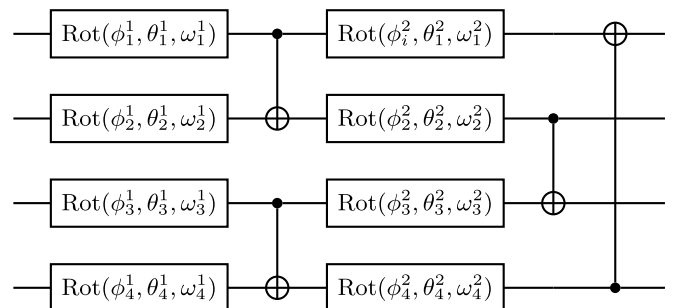

FIG. 15. Variational quantum circuit used for parametrizing the unitaries in the Schmidt decomposition for spin and fermionic models.
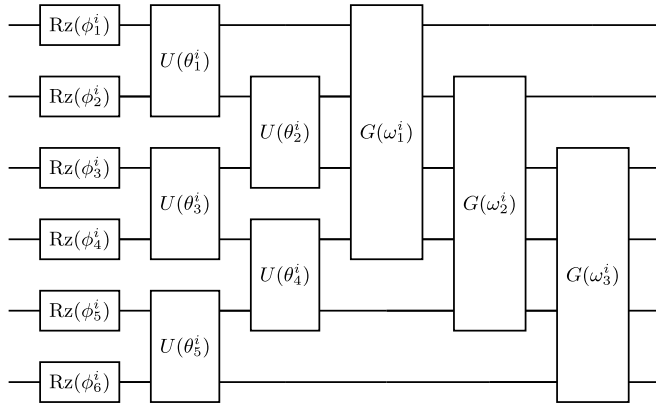
FIG. 16. Variational quantum circuit used for parametrizing the unitaries in the Schmidt decomposition for the shell models. The gates correspond to the ones described in the main text.

overfitting. On the other hand, high values deteriorate the algorithm's computational efficiency, convergence speed, and memory requirement in the same way as batches in stochastic gradient descent. For ARNN, the Lecun normal initializer for the weight, zero initial biases, and scaled exponential linear unit (SELU) activation function $\lambda = 1.0507$ $\alpha = 1.6733$ were used [59].

APPENDIX F: VARIATIONAL CIRCUITS

In this section, we provide a visual example of the quantum circuits used for VQE *Ansätze*. A layer of the variational circuit used for the spin systems is shown in Fig. 15 for $N = 4$ qubits, while Fig. 16 shows one layer of the circuits used for the nuclear shell model. We use the notation Rot to describe a generic rotation around the Bloch sphere, composed of *Ry*, *Rz*, and *Ry* rotation.

APPENDIX G: EVOLUTION OF THE GENERATED SET

A histogram was produced to visualize the ARNN training set evolution. Shown in Fig. 17, it illustrates the number of
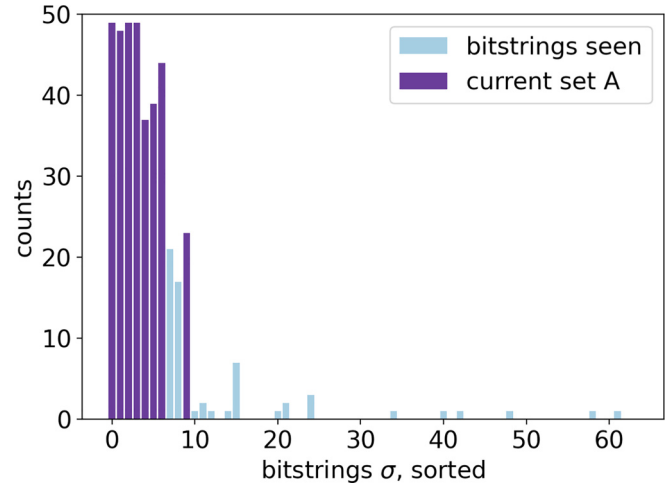

FIG. 17. TFIM 2D 12 spins, MMD loss: Histogram showing the bitstrings seen by the ARNN (present in the set A') with the respective number of times (counts). The results after 50 iterations are presented. The bitstrings contained in the final set A' are highlighted in purple and the bitstrings seen previously are illustrated in light blue.

times each bitstring has been present in the training set $\mathcal{T} = A'$ of the ARNN. The bitstrings present in the final set are shown in purple and those present during the algorithm are shown in light blue. The illustrated example is at the end of the algorithm on the TFIM 2D 12 spins with the MMD loss. Bitstrings are ordered in such a way that their associated Schmidt coefficient decreases (in absolute value). The bitstrings associated with the highest Schmidt coefficient are the most frequently viewed by the ARNN.

At the end, the seven bitstrings associated with the biggest Schmidt coefficient are present in the final set. The eighth bitstring in the set is bitstrings number ten. Given that bitstrings eight and nine were seen during the algorithm and that their associated Schmidt coefficients squared are very low and close to the one of bitstring ten, we can explain this lack by numerical errors when determining the Schmidt coefficients.

[1] N. Schuch, M. M. Wolf, F. Verstraete, and J. I. Cirac, Entropy scaling and simulability by matrix product states, Phys. Rev. Lett. **100**, 030504 (2008).

[2] J. Dukelsky, M. A. Martín-Delgado, T. Nishino, and G. Sierra, Equivalence of the variational matrix product method and the density matrix renormalization group applied to spin chains, Europhys. Lett. **43**, 457 (1998).

[3] S. Rommer and S. Östlund, Class of ansatz wave functions for one-dimensional spin systems and their relation to the density matrix renormalization group, Phys. Rev. B **55**, 2164 (1997).

[4] G. Carleo and M. Troyer, Solving the quantum many-body problem with artificial neural networks, Science **355**, 602 (2017).

[5] D. Pfau, J. S. Spencer, A. G. D. G. Matthews, and W. M. C. Foulkes, Ab initio solution of the many-electron Schrödinger equation with deep neural networks, Phys. Rev. Res. **2**, 033429 (2020).

[6] C. Giuliani, F. Vicentini, R. Rossi, and G. Carleo, Learning ground states of gapped quantum Hamiltonians with kernel methods, Quantum **7**, 1096 (2023).

[7] T. Westerhout, N. Astrakhantsev, K. S. Tikhonov, M. Katsnelson, and A. A. Bagrov, Generalization properties of neural network approximations to frustrated magnet ground states, Nat. Commun. **11**, 1593 (2020).

[8] J. Hermann, J. Spencer, K. Choo, A. Mezzacapo, W. Foulkes, D. Pfau, G. Carleo, and F. Noé, Ab initio quantum chemistry with neural-network wavefunctions, Nat. Rev. Chem. **7**, 692 (2023).

[9] A. Lovato, C. Adams, G. Carleo, and N. Rocco, Hidden-nucleons neural-network quantum states for the nuclear many-body problem, Phys. Rev. Res. **4**, 043178 (2022).

[10] J. R. McClean, J. Romero, R. Babbush, and A. Aspuru-Guzik, The theory of variational hybrid quantum-classical algorithms, New J. Phys. **18**, 023023 (2016).

[11] A. Peruzzo, J. McClean, P. Shadbolt, M.-H. Yung, X.-Q. Zhou, P. J. Love, A. Aspuru-Guzik, and J. L. O'Brien, A variational eigenvalue solver on a photonic quantum processor, Nat. Commun. **5**, 4213 (2014).

[12] A. Kandala, A. Mezzacapo, K. Temme, M. Takita, M. Brink, J. M. Chow, and J. M. Gambetta, Hardware-efficient variational quantum eigensolver for small molecules and quantum magnets, Nature (London) **549**, 242 (2017).

[13] J. Romero, R. Babbush, J. R. McClean, C. Hempel, P. J. Love, and A. Aspuru-Guzik, Strategies for quantum computing molecular energies using the unitary coupled cluster ansatz, Quantum Sci. Technol. **4**, 014008 (2019).

[14] A. Uvarov, J. D. Biamonte, and D. Yudin, Variational quantum eigensolver for frustrated quantum systems, Phys. Rev. B **102**, 075104 (2020).

[15] L. Crippa, F. Tacchino, M. Chizzini, A. Aita, M. Grossi, A. Chiesa, P. Santini, I. Tavernelli, and S. Carretta, Simulating static and dynamic properties of magnetic molecules with prototype quantum computers, Magnetochemistry **7**, 117 (2021).

[16] S. Monaco, O. Kiss, A. Mandarino, S. Vallecorsa, and M. Grossi, Quantum phase detection generalization from marginal quantum neural network models, Phys. Rev. B **107**, L081105 (2023).

[17] A. M. Romero, J. Engel, H. L. Tang, and S. E. Economou, Solving nuclear structure problems with the adaptive variational quantum algorithm, Phys. Rev. C **105**, 064317 (2022).

[18] E. F. Dumitrescu, A. J. McCaskey, G. Hagen, G. R. Jansen, T. D. Morris, T. Papenbrock, R. C. Pooser, D. J. Dean, and P. Lougovski, Cloud quantum computing of an atomic nucleus, Phys. Rev. Lett. **120**, 210501 (2018).

[19] I. Stetcu, A. Baroni, and J. Carlson, Variational approaches to constructing the many-body nuclear ground state for quantum computing, Phys. Rev. C **105**, 064308 (2022).

[20] O. Kiss, M. Grossi, P. Lougovski, F. Sanchez, S. Vallecorsa, and T. Papenbrock, Quantum computing of the ^{6}Li nucleus via ordered unitary coupled clusters, Phys. Rev. C **106**, 034325 (2022).

[21] A. Pérez-Obiol, A. M. Romero, J. Menéndez, A. Rios, A. García-Sáez, and B. Juliá-Díaz, Nuclear shell-model simulation in digital quantum computers, Sci. Rep. **13**, 12291 (2023).

[22] E. R. Anschuetz and B. T. Kiani, Quantum variational algorithms are swamped with traps, Nat. Commun. **13**, 7760 (2022).

[23] J. R. McClean, S. Boixo, V. N. Smelyanskiy, R. Babbush, and H. Neven, Barren plateaus in quantum neural network training landscapes, Nat. Commun. **9**, 4812 (2018).

[24] A. Eddins, M. Motta, T. P. Gujarati, S. Bravyi, A. Mezzacapo, C. Hadfield, and S. Sheldon, Doubling the size of quantum simulators by entanglement forging, PRX Quantum **3**, 010309 (2022).

[25] M. Rossmannek, P. K. Barkoutsos, P. J. Ollitrault, and I. Tavernelli, Quantum HF/DFT-embedding algorithms for electronic structure calculations: Scaling up to complex molecular systems, J. Chem. Phys. **154**, 114105 (2021).

[26] T. Serwatka, R. G. Melko, A. Burkov, and P.-N. Roy, Quantum phase transition in the one-dimensional water chain, Phys. Rev. Lett. **130**, 026201 (2023).

[27] A. Kitaev and J. Preskill, Topological entanglement entropy, Phys. Rev. Lett. **96**, 110404 (2006).

[28] M. Levin and X.-G. Wen, Detecting topological order in a ground state wave function, Phys. Rev. Lett. **96**, 110405 (2006).

[29] R. Islam, R. Ma, P. M. Preiss, M. Eric Tai, A. Lukin, M. Rispoli, and M. Greiner, Measuring entanglement entropy in a quantum many-body system, Nature (London) **528**, 77 (2015).

[30] M. Germain, K. Gregor, I. Murray, and H. Larochelle, Made: Masked autoencoder for distribution estimation, in *Proceedings of the 32nd International Conference on Machine Learning, JMLR W&CP, Vol. 32* (PMLR, 2015), pp. 881–889.

[31] A. van den Oord, N. Kalchbrenner, O. Vinyals, L. Espeholt, A. Graves, and K. Kavukcuoglu, Conditional image generation with PixelCNN decoders, in *Proceedings of the 30th International Conference on Neural Information Processing Systems* (Curran Associates Inc., Barcelona, Spain, 2016), pp. 4797–4805.

[32] P. Huembeli, G. Carleo, and A. Mezzacapo, Entanglement forging with generative neural network models, arXiv:2205.00933.

[33] T. D. Barrett, A. Malyshev, and A. I. Lvovsky, Autoregressive neural-network wavefunctions for ab initio quantum chemistry, Nat Mach Intell **4**, 351 (2022).

[34] B. Herzog, B. Casier, S. Lebègue, and D. Rocca, Solving the Schrödinger equation in the configuration space with generative machine learning, J. Chem. Theor. Comput. **19**, 2484 (2023).

[35] A. Narayanan and M. Moore, Quantum-inspired genetic algorithms, in *Proceedings of IEEE International Conference on Evolutionary Computation* (IEEE, Piscataway, NJ, 1996), pp. 61–66.

[36] H. Larochelle and I. Murray, The neural autoregressive distribution estimator, in *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, Proceedings of Machine Learning Research, edited by G. Gordon, D. Dunson, and M. Dudík (PMLR, Fort Lauderdale, 2011), Vol. 15, pp. 29–37.

[37] P. Chen, G. Chen, and S. Zhang, Log hyperbolic cosine loss improves variational auto-encoder (2019).

[38] A. Gretton, K. M. Borgwardt, M. J. Rasch, B. Schölkopf, and A. Smola, A kernel two-sample test, J. Mach. Learn. Res. **13**, 723 (2012).

[39] S. Blundell, *Magnetism in Condensed Matter*, Oxford Master Series in Condensed Matter Physics (Oxford, New York, 2001).

[40] A. Y. Kitaev, Fault-tolerant quantum computation by anyons, Ann. Phys. **303**, 2 (2003).

[41] X. Chen, Z.-C. Gu, and X.-G. Wen, Local unitary transformation, long-range quantum entanglement, wave function renormalization, and topological order, Phys. Rev. B **82**, 155138 (2010).

[42] S. Cohen and D. Kurath, Effective interactions for the 1p shell, Nucl. Phys. **73**, 1 (1965).

[43] P. Jordan and E. Wigner, Ìber das paulische Äquivalenzverbot, Z. Phys. **47**, 631 (1928).

[44] P. K. Barkoutsos, J. F. Gonthier, I. Sokolov, N. Moll, G. Salis, A. Fuhrer, M. Ganzhorn, D. J. Egger, M. Troyer, A. Mezzacapo, S. Filipp, and I. Tavernelli, Quantum algorithms for electronic structure calculations: Particle-hole Hamiltonian and optimized wave-function expansions, Phys. Rev. A **98**, 022322 (2018).

[45] B. T. Gard, L. Zhu, G. S. Barron, N. J. Mayhall, S. E. Economou, and E. Barnes, Efficient symmetry-preserving state preparation circuits for the variational quantum eigensolver algorithm, npj Quantum Inf. **6**, 10 (2020).

[46] J. M. Arrazola, O. Di Matteo, N. Quesada, A. Delgado, and N. Killoran, Universal quantum circuits for quantum chemistry, Quantum **6**, 742 (2022).

[47] S. Bond-Taylor, A. Leach, Y. Long, and C. G. Willcocks, Deep generative modelling: A comparative review of VAEs, GANs, normalizing flows, energy-based and autoregressive models, IEEE Trans. Pattern Anal. Mach. Intell. **44**, 7327 (2022).

[48] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, Attention is all you need, in *31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA*, edited by I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (Curran Associates, Inc., 2017), Vol. 30.

[49] H. R. Grimsley, S. E. Economou, E. Barnes, and N. J. Mayhall, An adaptive variational algorithm for exact molecular simulations on a quantum computer, Nat. Commun. **10**, 3007 (2019).

[50] V. Bergholm *et al.*, Pennylane: Automatic differentiation of hybrid quantum-classical computations, arXiv:1811.04968.

[51] J. Bradbury, R. Frostig, P. Hawkins, M. J. Johnson, C. Leary, D. Maclaurin, G. Necula, A. Paszke, J. VanderPlas, S. Wanderman-Milne, and Q. Zhang, JAX: composable transformations of Python+NumPy programs (2018), http://github.com/google/jax.

[52] F. Vicentini, D. Hofmann, A. Szabó, D. Wu, C. Roth, C. Giuliani, G. Pescia, J. Nys, V. Vargas-Calderón, N. Astrakhantsev, and G. Carleo, NetKet 3: Machine learning toolbox for many-body quantum systems, SciPost Phys. Codebases, 7 (2022).

[53] M. Blondel, Q. Berthet, M. Cuturi, R. Frostig, S. Hoyer, F. Llinares-López, F. Pedregosa, and J.-P. Vert, Efficient and modular implicit differentiation, in *36th Conference on Neural Information Processing Systems (NeurIPS 2022)*, edited by S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (Curran Associates, Inc., 2022), Vol. 35, pp. 5230–5242.

[54] G. C. Patrick Huembeli and A. Mezzacapo, CQSL entanglement forging with GNN models: Releases tag v0.2, (2022), https://github.com/cqsl/Entanglement-Forging-with-GNN-models.

[55] M. S. Rudolph, S. Sim, A. Raza, M. Stechly, J. R. McClean, E. R. Anschuetz, L. Serrano, and A. Perdomo-Ortiz, Orqviz: Visualizing high-dimensional landscapes in variational quantum algorithms, arXiv:2111.04695 [quant-ph].

[56] P. de Schoulepnikoff, K. Oriel, G. Michele, V. Sofia, and C. Giuseppe, GitHub: Learning Schmidt decompositions with artificial neural networks and quantum circuits, https://github.com/PaulinDS/Learning-Schmidt-decompositions-with-NN-and-QC.

[57] J. Zhuang, T. Tang, Y. Ding, S. Tatikonda, N. Dvornek, X. Papademetris, and J. S. Duncan, Adabelief optimizer: Adapting stepsizes by the belief in observed gradients, Adv. Neural Info. Proc. Syst. **33**, 18795 (2020).

[58] I. Sutskever, J. Martens, G. E. Dahl, and G. E. Hinton, On the importance of initialization and momentum in deep learning, in *Proceedings of the 30th International Conference on Machine Learning, Atlanta, USA*, Proceedings of Machine Learning Research (PMLR, Georgia, USA, 2013), Vol. 28, pp. 1139–1147.

[59] G. Klambauer, T. Unterthiner, A. Mayr, and S. Hochreiter, Self-normalizing neural networks, in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17 (Curran Associates Inc., Red Hook, NY, 2017), pp. 972–981.