Gauge Invariant Autoregressive Neural Networks for Quantum Lattice Models

Di Luo, 1, 2, * Zhuo Chen, 1, * Kaiwen Hu, 1 Zhizhen Zhao, 3 Vera Mikyoung Hur, 4 and Bryan K. Clark 1, 2

1 Department of Physics, University of Illinois at Urbana-Champaign, IL 61801, USA
2 IQIUS and Institute for Condensed Matter Theory and NCSA Center for Artificial Intelligence Innovation, University of Illinois at Urbana-Champaign, IL 61801, USA
3 Department of Electrical and Computer Engineering and CSL, University of Illinois at Urbana-Champaign, Urbana, IL 61801, USA
4 Department of Mathematics, University of Illinois at Urbana-Champaign, Urbana, IL 61801, USA

Gauge invariance plays a crucial role in quantum mechanics from condensed matter physics to high energy physics. We develop an approach to constructing gauge invariant autoregressive neural networks for quantum lattice models. These networks can be efficiently sampled and explicitly obey gauge symmetries. We variationally optimize our gauge invariant autoregressive neural networks for ground states as well as real-time dynamics for a variety of models. We exactly represent the ground and excited states of the 2D and 3D toric codes, and the X-cube fracton model. We simulate the dynamics of the quantum link model of U(1) lattice gauge theory, obtain the phase diagram for the 2D Z_2 gauge theory, determine the phase transition and the central charge of the SU(2), anyonic chain, and also compute the ground state energy of the SU(2) invariant Heisenberg spin chain. Our approach provides powerful tools for exploring condensed matter physics, high energy physics and quantum information science.

I. INTRODUCTION

In recent years, there has been a growing interest in machine learning approaches to simulating quantum many-body systems [1–20]. An important step in this direction is the use of neural networks, e.g., restricted Boltzmann machines, to represent variational wave functions. However, many neural networks do not automatically enforce the symmetries of physical models. A considerable amount of work has been devoted to remedy the deficiency for several classes of global symmetries, such as translational symmetry [3], discrete rotational symmetry [3], global U(1) symmetry [4], and anti-symmetry [5–7].

In addition to global symmetries, local symmetries can be encoded through gauge invariance. The notion of gauge invariance is crucial in quantum mechanics. In high energy physics, theory is required to be invariant under the action of gauge symmetry groups [21]. Gauge invariance appears naturally in various condensed matter physics models. For example, topological states of toric code and double semion models arise as the ground states of their gauge-invariant Hamiltonians [22, 23]. Also, novel quantum matter such as fracton is the ground state of a Hamiltonian where the subsystem symmetry is gauged [24]. In quantum information, various quantum error correction codes can be viewed as eigenstates in a certain gauge-invariant code space [25]. The study of quantum lattice models with gauge symmetries is significant to enhance our understanding of high energy physics, condensed matter physics, and quantum information science.

Simulating quantum many-body gauge theory is exponentially costly. There has been much effort to efficiently simulate quantum lattice gauge theory with both digital and analog quantum computers [26], but more effort is required experimentally to achieve good fidelity. Two standard approaches to simulating gauge theory classically are stochastic, integrating an effective Lagrangian by sampling, and variational. When simulating gauge theory, the stochastic approach naturally obeys gauge invariance but is plagued with exponential costs associated with the sign problem in models with finite density of fermions or involving quantum dynamics [27]. The variational approach overcomes the difficulty by being constrained to an approximate variational space. Imposing gauge symmetries in the variational approach is particularly important and challenging as, otherwise, lower energy states can exist in the gauge-violating part of a Hilbert space. Therefore gauge symmetries must be explicitly constrained. While the stochastic approach has been well studied, there have been limited attempts at using the variational approach for gauge theory. Tensor networks can be readily applied to gauge theory in one dimension and ongoing efforts are required to work with challenges in higher dimensions [26, 28]. A variational approach based on gauge equivariant networks has been introduced very recently [29–31].

We develop for the first time, a general approach to constructing gauge invariant autoregressive neural networks (AR-NN) for quantum lattice models with gauge symmetries. We also extend our AR-NN to algebraic structures beyond gauge invariance, such as the fusion rule for anyons. Autoregressive neural networks, such as recurrent neural networks (RNN) [32, 33], pixel convolutional neural networks (pixelCNN) [34], and Transformers [35], have revolutionized the fields of computer vision and language translation and generation, among many others. Autoregressive neural networks quantum states have recently been introduced in quantum many-body physics [4, 36, 37] and shown to be capable of rep-

* Co-first authors.
II. GAUGE INVARIANT CONSTRUCTION

Our goal in this work is to generate autoregressive neural networks (AR-NN) which variationally represent wave functions of quantum lattice models and explicitly obey their gauge symmetries—i.e. given a set of gauge symmetry operators \( \{ G_i \} \) with local support, we would like to construct a wave function \( |\psi\rangle \) such that \( G_i |\psi\rangle = |\psi\rangle \) for each \( i \). To do this, we will work within the ‘gauge basis’ \( \{ |x\rangle \} \) which is diagonal in the gauge, \( \langle x|\psi\rangle = \langle x|G_i|\psi\rangle \). A sufficient condition of gauge invariance of the wave function is to ensure that the gauge-violating basis elements \( |x\rangle \) have zero amplitude in \( |\psi\rangle \). Throughout this work, we will primarily work with gauges \( G_i \) which are local—i.e. \( G_i |x\rangle \) only affects a compact range of sites within the vicinity of site \( i \).

While we would typically want our AR-NN to take as input the configuration \( \{ x_1, x_2, \ldots, x_N \} \) and evaluate \( \psi(x_1, x_2, \ldots, x_N) \), we will find it useful to instead evaluate \( \psi(\vec{x}) \) where \( \vec{x} \equiv (x_1, x_2, \ldots, x_N) \), \( \vec{x}_i \equiv (x_{1i}, x_{2i}, \ldots, x_{Ni}) \), is a composite particle specifying the configuration of not only site \( i \) but also some number of nearby sites. The motivation for working with composite particles is that a particular local gauge constraint \( G_i \) might only depend on composite particle \( \vec{x}_i \) (and potentially \( \vec{x}_{i+1} \)), making it easier to apply the gauge constraints. Different composite particles can naturally overlap in physical sites and we will simply augment our gauge constraints to require that the configurations of the composite particles agree on the state of a physical site—i.e. basis states of composite particles which map to disagreeing physical states should also have zero amplitude.

AR-NN perform two functions: sampling and evaluation. AR-NN can sample configurations \( \vec{x} \) from \( |\psi(\vec{x})|^2 \). This is done sequentially (in some pre-determined order) one composite particle \( \vec{x}_i \) at a time; the probability to sample \( \vec{x}_i \) is equal to \( a^2(\vec{x}_i|\bar{x}_{<i}) \) where \( a(\vec{x}_i|\bar{x}_{<i}) > 0 \) is a function which returns the conditional amplitude. Evaluation of the AR-NN gives a value \( \psi(\vec{x}) = \prod_{i=1}^{N} a(\vec{x}_i|\bar{x}_{<i})e^{i\theta(\vec{x}_i|\bar{x}_{<i})} \) where \( \theta(\vec{x}_i|\bar{x}_{<i}) \) is a function which returns the conditional phase. Evaluation and sampling rely on the existence of a gauge block which takes \( \vec{x}_1, \ldots, \vec{x}_{k-1} \) and outputs the possible values \( \{ z_k \} \) of \( \vec{x}_k \) along with their respective amplitudes \( a(z_k|\vec{x}_{<k}) \) and phases \( \theta(z_k|\vec{x}_{<k}) \), ensuring that the amplitude of any configuration which is going to violate the gauge constraint is set to zero. To build this gauge block, we start with an autoregressive neural network block which returns a list of amplitudes which do not constrain the gauge (such blocks are standard in autoregressive models such as Transformers and RNN); we then zero out those partial configurations which break the gauge (on the already established composite particles) and renormalize the probabilities in this list (see Fig. 1(a)). Given the gauge block, it is then straightforward to both sample and evaluate (see Fig. 1(b, c)). Note the probability induced by our AR-NN is different from the probability induced by the WR-NN with only the autoregressive neural network block even if one projects out the gauge-breaking configurations from the latter network.

It is worth noticing that the gauge invariant autoregressive neural networks are not limited to gauge theory, but can be generalized to wave functions with either local or global constraints which are checked in the same way as gauge constraints are checked. This will be helpful for describing constraints from certain global symmetries or special algebraic structure, such as the SU(2) symmetry for the Heisenberg model and the SU(2) \(_k\) fusion rules for non-abelian anyons.

III. OPTIMIZATION ALGORITHMS

We use AR-NN to calculate both ground states and real-time dynamic properties. In both cases, we need to optimize our AR-NN. For ground states, an AR-NN is optimized with respect to energy and for real-time dynamics, we optimize an AR-NN at time-step \( t + 2\tau \) given a network at time \( t \). We describe the details of these optimizations. As these optimization approaches are general, we use \( x \) to denote a configuration, but for the context of the paper, \( x \) should be viewed as a composite particle configuration.

For the ground state optimization, we stochastically minimize the expectation of energy for a Hamiltonian \( H \) and a wave function \( |\psi_0\rangle \) as

\[
\langle \psi_0|H|\psi_0\rangle \approx \frac{1}{N} \sum_{x \sim |\psi_0|} \frac{H\psi_0(x)}{\psi_0(x)} \approx \frac{1}{N} \sum_{x \sim |\psi_0|} E_{\text{loc}}(x),
\]  

(1)
where $N$ is the batch size and the gradient is given by
\[ \frac{\partial}{\partial \theta} \langle \psi_\theta | H | \psi_\theta \rangle \approx \frac{2}{N} \sum_{x \sim |\psi_\theta|^2} R \left\{ E_{\text{loc}}(x) \frac{\partial}{\partial \theta} \log \psi_\theta^*(x) \right\}. \] 

We further control the sampling variance [45] by subtracting from $E_{\text{loc}}(x)$ the average over the batch, $E_{\text{avg}} \equiv 1/N \sum_{x \in \text{batch}} E_{\text{loc}}(x)$, and define the stochastic variance reduced loss function as
\[ \mathcal{L}_d = \frac{2}{N} \sum_{x \sim |\psi_\theta|^2} R \left\{ [E_{\text{loc}}(x) - E_{\text{avg}}] \log \psi_\theta^*(x) \right\}, \] (3)

where the gradient is taken on $\log \psi_\theta^*$ using PyTorch’s [46] automatic differentiation.

With this loss function, we also use transfer learning techniques [47, 48]. We train our neural networks in smaller systems and use these parameters as the initial starting points for optimizing for larger systems (see Appendix D for details).

For the dynamics optimization, we use a stochastic version of the logarithmic forward-backward trapezoid method [49], which can be viewed as a higher order generalization of IT-SWO [50] and the logarithmic version of the loss functions in Refs. 9 and 36. We initialize two copies of the neural network $\psi_{\theta(t)}$ and $\psi_{\theta(t+2\tau)}$. At each time step, we train $\psi_{\theta(t+2\tau)}$ to match
\[ (1 + iH\tau) |\psi_{\theta(t+2\tau)} \rangle \equiv |\Psi_\theta \rangle \] and \[ (1 - iH\tau) |\psi_{\theta(t)} \rangle \equiv |\Phi \rangle \] by minimizing the negative logarithm of the overlap,
\[ -\log (\langle \Psi_\theta | \Phi \rangle |\Psi_\theta \rangle) / (\langle \Psi_\theta | \Psi_\theta \rangle |\Phi \rangle) \]. (Since we only take the gradient on $\theta(t+2\tau)$, for simplicity, we write $\theta$ for $\theta(t+2\tau)$ and neglect $\theta(t)$.) The inner products related to $\theta$ can be evaluated stochastically as
\[ \langle \Psi_\theta | \Phi \rangle \approx \frac{1}{N} \sum_{x \sim |\psi_\theta|^2} |\psi_\theta(x) |\Phi(x) |^2 \equiv \frac{1}{N} \sum_{x \sim |\psi_\theta|^2} \alpha(x), \] (4)
\[ \langle \Psi_\theta | \Psi_\theta \rangle \approx \frac{1}{N} \sum_{x \sim |\psi_\theta|^2} |\psi_\theta(x) |^2 \equiv \frac{1}{N} \sum_{x \sim |\psi_\theta|^2} \beta(x). \] (5)

The gradient of the negative logarithm of the overlap can be evaluated stochastically as
\[ \frac{\partial}{\partial \theta} \left( -\log (\langle \Psi_\theta | \Phi \rangle |\Psi_\theta \rangle) / (\langle \Psi_\theta | \Psi_\theta \rangle |\Phi \rangle) \right) \approx \frac{2}{N} \sum_{x \sim |\psi_\theta|^2} R \left\{ \left[ \frac{\beta(x)}{\beta_{\text{avg}}} - \frac{\alpha(x)}{\alpha_{\text{avg}}} \right] \frac{\partial}{\partial \theta} \log \psi_\theta^*(x) \right\}, \] (6)

where $\alpha_{\text{avg}}$ and $\beta_{\text{avg}}$ are respectively the average values of $\alpha(x)$ and $\beta(x)$ over the batch of samples. We can then define the loss function as
\[ \mathcal{L}_d \approx \frac{2}{N} \sum_{x \sim |\psi_\theta|^2} R \left\{ \left[ \frac{\beta(x)}{\beta_{\text{avg}}} - \frac{\alpha(x)}{\alpha_{\text{avg}}} \right] \log \psi_\theta^*(x) \right\}. \] (7)
where the gradient is taken on \( \log \Psi^2 \) using PyTorch’s [46] automatic differentiation.

For both optimizations, \( \psi_0(x) \) is evaluated as described in Fig. 1(a) and \( z \) is sampled from \( |\psi_0|^2 \) as described in Fig. 1(b). The full derivations of the stochastic gradients for both optimizations are in Appendix E.

### IV. APPLICATIONS IN QUANTUM LATTICE MODELS

#### A. U(1) Quantum Link Model

The quantum link model (QLM) of U(1) lattice gauge theory in 1+1 dimensions in the Hamiltonian formulation with staggered fermions [39] is defined as

\[
H_{QLM} = -\sum_i \left[ \psi_i^+ U_{i,i+1} \psi_{i+1} + \psi_{i+1}^+ U_{i,i+1}^\dagger \psi_i \right] + m \sum_i (-1)^i \psi_i^+ \psi_i + \frac{g^2}{2} \sum_i E_{i,i+1}^2,
\]

where \( m \) is the staggered fermion mass, \( g \) is the gauge coupling, \( i = 1, 2, ... \) labels the lattice site, \( \psi_i \) is the fermion operator, \( U_{i,i+1} \) is the link variable and \( E_{i,i+1} \) the electric flux for the U(1) gauge field on link \((i, i+1)\) [39]. We denote by \( |q_i\rangle \) the basis state at site \( i \), and by \( |e_{i,i+1}\rangle \) the basis at link \((i, i+1)\). Each unit cell is defined to include two sites and two links. The operators \( E_{i,i+1} \) and \( U_{i,i+1} \) satisfy the following commutation relations:

\[
[E_{i,i+1}, U_{i,i+1}] = U_{i,i+1}, \quad [E_{i,i+1}, U_{i,i+1}^\dagger] = -U_{i,i+1}^\dagger \quad \text{and} \quad [E_{i,i+1}, U_{i,i+1}^\dagger] = 2E_{i,i+1}.
\]

The gauge constraint is given by the Gauss’s law operator \( \tilde{G}_i = \psi_i^\dagger \psi_i - E_{i,i+1} + E_{i-1,i} + \frac{1}{2}((-1)^i - 1) \) such that the ground state \( |\psi\rangle \) satisfies \( \tilde{G}_i |\psi\rangle = 0 \) for each \( i \). The QLM has gained growing interests and been studied in different settings in recent years [26, 28, 51–53]. We focus on the (1+1)D QLM with the \( S = 1/2 \) representation for the link operators \( U_{i,i+1} \) and \( E_{i,i+1} \). Under the Jordan-Wigner transformation, Eq. 8 becomes [54]

\[
H = -\sum_i \left[ S_i^+ S_{i,i+1}^+ S_{i,i+1}^- + \text{H.c.} \right] + m \sum_i (-1)^i (S_i^2 + \frac{1}{2}) + \frac{g^2}{2} \sum_i \frac{1}{4},
\]

where \( S_i^\pm \equiv S_i^1 \pm i S_i^2 \), \( S_i^1, S_i^2, S_i^3 \) are the Heisenberg matrices, and the Gauss’s law operator becomes \( \tilde{G}_i = S_i^3 - S_{i,i+1}^3 + S_{i-1,i}^3 + \frac{1}{2}((-1)^i) \). For the \( S = 1/2 \) representation, the last term on the right side of Eq. 9 is constant and, hence, can be discarded.

We define the composite particles of our gauge invariant AR-NN as in Fig. 2 and choose an order from left to right. Each composite particle \( |\sigma_i\rangle \) consists of a fermion \( |q_i\rangle \), which can be either \( |\bullet\rangle \) or \( |\circ\rangle \), and a gauge field in the link \( |e_{i,i+1}\rangle \), which can be either \( |\rightarrow\rangle \) or \( |\leftarrow\rangle \). Note that in this case the composite particles do not overlap. The Gauss’s law operator \( G_i \) acts on \( |\sigma_i\rangle \) and \( |\sigma_{i+1}\rangle \) to determine allowed configurations and so can be checked in the gauge block which generates the composite particle at site \( i + 1 \). For example, given \( |\sigma_i\rangle = |\bullet\rangle \), \( |\sigma_{i+1}\rangle \) can only be \( |\rightarrow\rangle \) or \( |\leftarrow\rangle \) if \( i \) is even, and \( |\circ\rangle \) if \( i \) is odd.

![FIG. 2. Composite particles for the quantum link model. Each composite particle is defined as \( |\sigma_i\rangle \equiv |q_i, e_{i,i+1}\rangle \). We check Gauss’s law between \( |\sigma_i\rangle \) and \( |\sigma_{i+1}\rangle \).](image)

![FIG. 3. Variational ground state optimization for the 6-unit-cell (12 sites and 12 links) open-boundary QLM for \( m = 0 \) with and without gauge invariant construction. The gauge invariant autoregressive neural network reaches an accurate ground state while the ansatz without gauge constraints arrives at a non-physical state in the optimization. We use the Transformer neural network with 1 layer, 32 hidden dimensions and the real-imaginary parameterization (see Fig. A3). The neural network is randomly initialized and is trained for 1000 iterations with 12000 samples in each iteration. The neural network architecture and optimization details are discussed in Appendix D.](image)

We implement and variationally optimize this AR-NN for the ground state of Eq. 9. Fig. 3 shows the results for 6 unit cells (i.e. 12 particles) which closely match the energy of the exact solution. More importantly, the gauge invariant construction guarantees that the solution is in the physical space, while the neural network without gauge constraint (i.e. removing the gauge-checking from the AR-NN) finds a lower energy but non-physical state. We also consider the real-time dynamics for \( m = 0.1 \) and \( m = 2.0 \) for 12 unit cells. Fig. 4(a) shows the conservation of energy for different ansatzes. The total energy is \(-1.2\) for \( m = 0.1 \), and \(-24\) for \( m = 2.0 \). Although the gauge invariant AR-NN exhibits jumps in energy at certain points due to optimization instability, it preserves energy more smoothly in most steps compared with the non-gauge ansatz. We find that our gauge in-
variant AR-NN captures the correct electric field oscillation compared with the non-gauge ansatz (see Fig 4(b)) and, additionally, the anticipated string inversion of the electric flux for small mass (and respectively the static electric flux for large mass) (see Fig 5).

While the current work focuses on the $S = 1/2$ representation, our construction can be generalized to an arbitrary $S$ representation. For a higher spin $S$, composite particles can be defined similarly (see Fig. 2) except that the degree of freedom for each $e_{i,i+1}$ increases to $2S + 1$ as $S$ increases.

![FIG. 4. Dynamics for the 12-unit-cell (24 sites and 24 links) open-boundary QLM for $m = 0.1$ and $m = 2.0$ with and without gauge invariant construction. The dashed curves are the exact results from the exact diagonalization for 6 unit cells, which can be used as a reference for the 12-unit-cell neural network results. (a) The change in the energy during the dynamics. (b) The expectation value of the electric field averaged over all links. We use the Transformer neural network with 1 layer, 32 hidden dimensions and the real-imaginary parameterization. The initial state is $|\bullet \rightarrow o \rightarrow \rangle$ for each unit cell and we train the neural network using the forward-backward trapezoid method with the time step $\tau = 0.005$, 600 iterations in each time step, and 12000 samples in each iteration. The neural network architecture, initialization and optimization details are discussed in Appendix D.](image1)

![FIG. 5. Dynamics of the gauge invariant AR-NN for the 12-unit-cell QLM with (a) $m = 0.1$ and (b) $m = 2.0$. The ansatz, initialization and optimization are the same as in Fig. 4 and are discussed in Appendix D.](image2)

### B. 2D $Z_N$ Gauge Theory

For the 2D toric code [23], consider an $L \times L$ periodic square lattice, where each edge has the basis $\{|0\rangle, |1\rangle\}$. Let $V, P, E$ denote the sets of vertices, plaquettes and edges of the lattice, respectively, such that $|V| = L^2, |P| = L^2, |E| = 2L^2$. Here we consider the toric code with a transverse magnetic field

$$H = H_{TC} - h \sum_{e \in E} \sigma^z_e,$$

where $H_{TC}$ is the toric code Hamiltonian

$$H_{TC} = - \sum_{v \in V} A_v - \sum_{p \in P} B_p,$$

$A_v \equiv \prod_{e \ni v} \sigma^z_e$ (the star operator), $B_p \equiv \prod_{e \ni p} \sigma^x_e$, and $h$ is the strength of the transverse field. Note that $A_v$ is the gauge constraint such that the ground state $|\psi\rangle$ of Eq. 10 and Eq. 11 satisfies $A_v |\psi\rangle = |\psi\rangle$ for each $v$.

The composite particle construction is illustrated in Fig. 6. We order our consecutive particles by an “S” shape going up one row and down the next (see Fig. A8(b) in Appendix D). Two constraints must be checked in the gauge checking process of a gauge block.

When working on the gauge block associated with composite particle $\vec{x}_v$, we check that $A_v |\vec{x}_v\rangle = |\vec{x}_v\rangle$ (despite $A_v$ acting on an entire state, this can be checked locally on a single composite particle). In addition, composite particles overlap with their four immediately adjacent composite particles. The gauge block for the composite particle at $v$ therefore checks consistency of the physical sites with the first $v - 1$ composite particles. For example, given the configuration $|0\rangle_1 |0\rangle_0$ for a composite particle, the composite particle to the right can only be
The ground state of Eq. 11 is \( |\psi\rangle = \prod_{v \in V} (I + A_v) |+\rangle^\otimes n \) where \(|+\rangle = (|0\rangle + |1\rangle)/\sqrt{2}\) — i.e. an equal superposition of all configurations in the gauge basis which do not violate the gauge constraint. In our construction, if we use an autoregressive neural network block which gives equal weight to all the configurations (this is straightforward to arrange by setting the last linear layer’s weight matrices to zero and bias vectors to equal amplitudes), we exactly achieve this state. Checking the \( A_v \) does not affect the relative probabilities because it is not conditional involving only one composite particle. On the other hand, the ‘gauge constraints’ which verify consistency of the underlying state of the sites leave equal probability between all consistent states. To see this, we examine the effect of the gauge constraint on \(|a(x_k|x_{<k})\rangle^2\) for any given \( k \), which is the conditional probability of the composite particle \( x_k \). Due to the conditioning from previous composite particles \( \{x_{<k}\} \), some sites of the composite particle \( x_k \) are fixed. For the Gauss’s law gauge constraints to be 1, the product of all the unchosen site configurations in \( x_k \) must be either 1 or −1, depending on the chosen site configurations. Let \( S_1 = \{b_1, \ldots, b_b|\prod_{j=1}^b b_j = 1\} \) and \( S_{-1} = \{c_1, \ldots, c_r|\prod_{j=1}^r c_j = -1\} \) be the two possible sets of unchosen site configurations, where \( b_j \) and \( c_j \) are the configurations of the unchosen sites in \( x_k \). Consider a function \( f : S_1 \to S_{-1} \) such that \( f(b_1) = -c_1 \) and \( f(b_j) = c_j \) otherwise. Notice that \( f \) is bijective and thus \( S_1 \) and \( S_{-1} \) have the same cardinality, implying that after normalization \(|a(x_k|x_{<k})|^2\) will have the same amplitude for any \( \{x_{<k}\} \). We can also generate excited states by changing the \( A_v \) for a fixed (even) number of vertices to constrain this local eigenvalue to be −1 instead of 1. We provide a numerical verification of this by computing the energy for an exactly represented tower of ground and excited states in Fig. 7.

With a nonzero value of the external field \( h \), the ground state of Eq. 10 is no longer exactly representable, and we variationally optimize our AR-NN to compute the ground state energy. Fig. 8 shows the minimum energy for Eq. 10 for different \( h \) and the energy derivative, computed using the Hellman-Feynman theorem [55]. The toric code is expected to exhibit a quantum phase transition between the topological and trivial phases at an intermediate value of \( h \), and the sharp change of the energy derivative around \( h = 0.34 \) is an indicator of this phase transition, which is consistent with the theoretical prediction of \( h \approx 0.32 \) in the thermodynamic limit [56, 57]. We can additionally identify the transition by considering the Wilson loop operator \( W_C = \prod_{e \in C} \sigma_z^e \) for a closed loop \( C \). It is predicted that the topological order phase follows an area law decay, \( \langle W_C \rangle \sim \exp(-\alpha A_C) \), and the trivial phase follows a perimeter law decay, \( \langle W_C \rangle \sim \exp(-\beta P_C) \), where \( A_C, P_C \) are the enclosed area and perimeter of the loop \( C \) [58]. Fig. 9 shows the values of \( \langle W_C \rangle \) using our variationally optimized AR-NN. By comparing the respective fits to the area and perimeter laws we again see the transition at \( h = 0.34 \). Finally, we compare the non-local string correlation operators \( S_{\gamma} = \prod_{e \in \gamma} \sigma_z^e \) of our variational states which could be viewed as a measure of the correlation of a pair of excited particle and anti-particle along a path \( \gamma \). In the topological order phase the non-local string operators will decay to zero while they will remain constant at the trivial phase [59]. In Fig. 10, this is seen clearly on both sides of the transition.

![FIG. 6. Composite particles for the 2D toric code. (a) Physical structure of 2D toric code with red circles specifying composite particles. Note multiple composite particles share the same physical sites. (b) Composite particles. We define each star as a composite particle (red circle) and check bond consistency for physical sites shared by adjacent composite particles (blue dashed ovals).](image)

![FIG. 7. Energies of the analytical constructions of the ground and excited states of the 11×11 2D and 4×4×4 3D toric code, and the 4×4×4 X-cube fracton model. The dashed lines are the exact values for each model. The analytical construction generates the same values as the exact up to stochastic errors from sampling.](image)
Our approach can be naturally generalized to 2D $\mathbb{Z}_N$ gauge theory, which can be described in the language of Kitaev’s $D(G)$ model with group $G = \mathbb{Z}_N$ (see Appendix A). In this case, the basis at each edge becomes a group element in $\mathbb{Z}_N$. Similarly to Fig. 6, one can define a composite particle over four edges from a vertex and impose gauge invariance. We can also extend our approach to the (1+1)D $\mathbb{Z}_N$ lattice quantum electrodynamics (QED) model, which is discussed in Appendix B.

C. 3D Toric Code and Fracton Model

We turn to gauge invariant AR-NN for the ground and excited states of the 3D toric code [40] and the fracton model [41, 60]. The 3D toric code generalizes the 2D toric code to an $L \times L \times L$ periodic cube where each edge has the basis $\{|0\rangle, |1\rangle\}$. The Hamiltonian takes the same form as the 2D model (see Eq. 11) except that for each $A_v \equiv \prod_{e \in \partial v} \sigma_e^z$ there are six edges $e$ associated with each vertex $v$. A ground state of the 3D toric code similarly satisfies $A_v |\psi\rangle = B_p |\psi\rangle = |\psi\rangle$ for each $v, p$. One of the degenerate ground states can also be expressed as $|\psi\rangle = \prod_{i \in V} (1 + A_i) |+\rangle^\otimes n$. The excited states can be generated by breaking certain constraints from $A_i$ and $B_i$ as in the 2D case.

The X-cube fracton model [41] is also defined on an $L \times L \times L$ periodic cube where each edge has the basis $\{|0\rangle, |1\rangle\}$. The Hamiltonian takes the following form

$$H_{\text{fracton}} = - \sum_{v \in V} A_v - \sum_{c \in C} B_c,$$

where $B_c \equiv \prod_{e \in c} \sigma_e^x$ over the edges in a small cube. The gauge constraint, i.e. Gauss’s law, is $B_c |\psi\rangle = |\psi\rangle$. There are three $A_v \equiv \prod_{e \in \partial v} \sigma_e^x$ for three choices of $i = zy, xy, xz$, depending on which 2D plane $A_v$ acts on.
The operators are illustrated in Fig. 11. A ground state of the X-cube fracton model satisfies $A^u_i |\psi\rangle = B_c |\psi\rangle = |\psi\rangle$ for each $i, v, c$. One of the ground states can be expressed as $|\psi\rangle = \prod (1+B_c) |+\rangle^{\otimes n}$. The excited states break some constraints such that $A^v_i |\psi\rangle = -|\psi\rangle$ or $B_c |\psi\rangle = -|\psi\rangle$ for certain $A^v_i, B_c$.

The composite particles for the 3D toric code and the X-cube fracton model are illustrated in Fig. 12. For the 3D toric code, a composite particle is made up of six particles associated with a vertex. The ground state can be constructed by initializing the bias of the autoregressive neural network to be all the $|+\rangle$ state and imposing gauge checking on each composite particle to be 1. The excited states can be constructed by forcing even numbers of composite particles to have gauge checking value $-1$. For the X-cube fracton model, a composite particle consists of twelve particles on each mini cube. The ground state comes from initializing all biases of the autoregressive neural network to be $|+\rangle$ and requiring all composite particles to have the gauge checking value 1. The excited states break the gauge checking value on sets of four nearby composite particles to be $-1$. We numerically verify the exact representations of ground and excited states of the 3D toric code and the X-cube fracton model in Fig. 7, where the energy is shown to be exactly the same as the theoretical predictions.

Our approach can be naturally generalized to the Haah’s code fracton [60] and checkerboard fracton [41] models. Similarly to 2D $\mathbb{Z}_N$ gauge theory (see Section IV B), one can consider applying gauge invariant AR-NN to study the 3D $\mathbb{Z}_N$ gauge theory in the context of the 3D toric code or the X-cube fracton model [24] with an external field.

## D. SU($2)_k$ Anyonic Chain and SU(2) Symmetry

Non-abelian anyons play a crucial role in universal topological quantum computation. Here we consider a chain of Fibonacci anyons, which can be regarded as an SU($2)_{k=3}$ deformation of the ordinary quantum spin-1/2 chain [44]. In this model, there is one type of anyon $\tau$ and a trivial vacuum state $\mathbb{1}$ for each site. The constraint from symmetry requires that $\tau$ and $\mathbb{1}$ satisfy the following fusion rule: $\tau \otimes \tau = \mathbb{1} \otimes \mathbb{1}$, $\tau \otimes \mathbb{1} = \mathbb{1} \otimes \tau = \tau$. We work directly in this basis where each site is either $\mathbb{1}$ or $\tau$, generating a gauge invariant AR-NN. We then proceed to work out the entire phase diagram of the Fibonacci...
anyons. This can be done particularly efficiently compared with standard Monte Carlo sampling [13] thanks to the exact sampling of the AR-NN.

Our gauge invariant AR-NN is constructed so that it obeys the anyon fusion rule directly by checking two adjacent input configurations and imposing zero amplitude when both are \( \tau \). Each anyon is a composite particle and the gauge checking implements the constraint from the anyon fusion rule.

We consider the Hamiltonian [43]

\[
H(\theta) = -\cos \theta \sum_{k} H_{1}^{(2)} + \sin \theta \sum_{k} H_{1}^{(3)}, \tag{13}
\]

where the two-anyon interactions can be described by the golden chain Hamiltonian [42, 43]

\[
H_{1}^{(2)} = |\tau \tau \tau | \tau \tau | + \phi^{-2} |\tau \tau \tau | \tau \tau | + \phi^{-1} \tau \tau \tau | \tau \tau | + \phi^{-3/2} (|\tau \tau \tau | \tau \tau | + \text{H.c.}), \tag{14}
\]

and the three-anyon interactions can be described by the Majumdar-Gosh chain Hamiltonian [43]

\[
H_{1}^{(3)} = |\tau \tau \tau | \tau \tau \tau | + (1 - \phi^{-2}) |\tau \tau \tau | \tau \tau \tau | + (1 - \phi^{-1}) (|\tau \tau \tau | \tau \tau \tau | + |\tau \tau \tau | \tau \tau \tau | + |\tau \tau \tau | \tau \tau \tau | + \text{H.c.}) - \phi^{-5/2} (|\tau \tau \tau | \tau \tau \tau | + |\tau \tau \tau | \tau \tau \tau | + \text{H.c.}) + \phi^{-2} (|\tau \tau \tau | \tau \tau \tau | + \text{H.c.}). \tag{15}
\]

\( \phi = (\sqrt{5} + 1) / 2 \) is the golden ratio.

This model is predicted to exhibit five phases with respect to different \( \theta \) [43]. Fig. 13 shows the optimized energies of the Hamiltonian in Eq. 13 for different \( \theta \) and the energy derivative computed using the Hellmann–Feynman theorem [55] as \( d \langle H \rangle / d \theta = (dH / d\theta) = -\sin \theta \sum_{k} (H_{1}^{(2)}) - \cos \theta \sum_{k} (H_{1}^{(3)}) \), versus \( \theta \). Phase transitions occur when the energy function is not differentiable. Vertical lines in orange are located at the exact phase transition points [43]. We use the 1D RNN with 3 layers, 36 hidden dimensions and the real-imaginary parameterization. We use the transfer learning technique where we first train the neural network on a 32-anyon model for 3000 iterations and then we transfer the neural network to the 40-anyon model for another 3000 iterations. In each iteration, we use 12000 samples. The neural network architecture, initialization, and optimization details are discussed in Appendix D.

Since each \( j_{i} \) is included as the SU(2) \( k \) fusion rule outcome of \( j_{i-1} \) and an extra 1/2 angular momentum, one can view \( j_{i} \) as a composite particle and gauge checking is the fusion rule.

The Fibonacci anyon is a special case of the SU(2) \( k=3 \) formulation, considering the mapping \( \tau \rightarrow j = 1 \) and \( \mathbb{I} \rightarrow j = 0 \) and applying 3/2 × 3 = 9/2 - \( j \) from the SU(2) \( 3 \) fusion rule to the even-number sites [42]. Note that this gives a slightly different AR-NN from what is described above. Besides the Fibonacci anyon, one can consider the Yang-Lee anyon, which follows the SU(2) \( 3 \) fusion rule [62].

Using this framework, one can also consider the Heisenberg spin chain with SU(2) symmetry since it can be considered as the SU(2) \( k \) deformation of the ordinary quantum spin-1/2 chain [44] as \( k \rightarrow \infty \). In Appendix C, we provide the detailed construction of an SU(2) invariant autoregressive neural network for the Heisenberg model.
which can be viewed as the case of SU(2)_{k=\infty}, and obtain accurate results for the 1D Heisenberg model.

V. CONCLUSION

We have provided a general approach to constructing gauge invariant autoregressive neural network wave functions for various quantum lattice models with gauge constraints or special algebraic structures. These wave functions explicitly satisfy the gauge or algebraic constraints, allow for perfect sampling of configurations, and are capable of explicitly returning the amplitude of a configuration including normalization. To accomplish this, we have upgraded standard AR-NN in such a way that the constraints can be autoregressively satisfied.

We have given explicit constructions of AR-NN which exactly represent the ground and excited states of several models, including the 2D and 3D toric codes as well as the X-cube fracton model. For those models for which exact representations are unknown, we variationally optimize our gauge invariant AR-NN to obtain either high-quality ground states or time-dependent wave functions. This has been done for the U(1) quantum link model, Z_N gauge theory, the SU(2)_3 anyonic chain, and the SU(2) quantum spin-1/2 chain. For these systems we are able to measure dynamical properties, produce phase diagrams, and compute observables accurately.

Our gauge invariant autoregressive neural networks open up the possibility of probing a larger variety of models and the physics associated with them. For example, the higher spin representation S > 1/2 in the (1+1)D QLM models would allow one to probe the quantum chromodynamics related physics of confinement and string breaking [63]. For the (3+1)D QLM models, there is the Coulomb phase which manifests in pyrochlore spin liquids [64]. For Z_N gauge theory, it will be interesting to consider the general Z_N toric code with transverse field or disorder, with a goal of understanding its phase diagram. Recently, there have been proposals to understand the Z_N X-cube fracton model with non-trivial statistical phases [24]. For non-abelian anyons, the general SU(2)_k formulation exhibits rich physics for different k and one can study the corresponding topological liquids and edge states [44]. Our approach can also be extended to study the phase diagram for the 2D Heisenberg models with SU(2) symmetry.

Besides exploring various models in condensed matter physics and high energy physics, our approach can also be further applied to quantum information and quantum computation. Fibonacci anyons are known to support universal topological quantum computation, which is robust to local perturbations [65]. It will be interesting to see how well one can approximately simulate topological quantum computation or different braiding operations with gauge invariant autoregressive neural networks. As toric codes are an important example of quantum error correction code, our approach can be used to approximately study the performance of a toric code under different noise conditions. With respect to the recent efforts on simulating lattice gauge theories with quantum computation, our approach also provides an alternative method to compare to and benchmark quantum computers. In summary, the approach we have developed is versatile and powerful for investigating condensed matter physics, high energy physics and quantum information science.

ACKNOWLEDGEMENT

Di Luo is grateful for insightful discussion in high energy physics with James Stokes and Jiayu Shen. Di Luo acknowledges helpful discussion with Luke Yeo, Oleg Dubinkin, Ryan Levy, Peijun Xiao, Ruoyu Sun and Giuseppe Carleo. This work utilizes resources supported by the National Science Foundation’s Major Research Instrumentation program, grant #1725729, as well as the University of Illinois at Urbana-Champaign [66]. Di Luo and Bryan Clark acknowledge support from the U.S. Department of Energy, Office of Science, Office of High Energy Physics QuantISED program. Zhizhen Zhao is partially supported by NSF DMS-1854791, NSF OAC-1934757, and Alfred P. Sloan Foundation. Vera Mikyoung Hur is partially supported by NSF DMS-1452597 and DMS-2009981.


Appendix A: Kitaev’s $D(G)$ Model and Exact Representation of Ground State

We generalize our gauge invariant autoregressive construction for the 2D $\mathbb{Z}_2$ toric code. Kitaev’s $D(G)$ model [23] is defined on an $L \times L$ periodic square lattice where each edge has a basis $\{ |g \rangle, g \in G \}$ for some group $G$. Here we focus on finite groups, in particular $G = \mathbb{Z}_N$ for $\mathbb{Z}_N$ theory. Without loss of generality, we attach an upward arrow for each edge in the $y$-direction and a right arrow for each edge in the $x$-direction. We employ the notation of Sec. IV B and introduce operators $A^g_0$ and $B^h_{p,h_i,h_i,h_i}$ as in Fig. A1.

The Hamiltonian defined on $\mathcal{H}(G)^{\otimes E}$ is

$$H = -\sum_{v \in E} A_v - \sum_{p \in P} B_p,$$

(A1)

where $A_v = \frac{1}{|G|} \sum_{g \in G} A^g_0$ is Gauss’s law and the gauge constraint, and $B_p = \sum_{h_{a,b} = h_{d,f} = 1} B^h_{p,h_{a,b},h_{d,f}}$.

Let $|\rangle = \frac{1}{\sqrt{|G|}} \sum_{g \in G} |g \rangle$, and $|\psi \rangle = \prod_{p \in E} B_p |\rangle^{\otimes E}$ is the ground state. This is because $|\psi \rangle$ is a ground state for each $A_v$ and $B_p$. It is easy to verify that $B_p |\psi \rangle = |\psi \rangle$. To see $A_v |\psi \rangle = |\psi \rangle$, notice that $A_v$ and $B_p$ commute and $A_v |\rangle^{\otimes E} = |\rangle^{\otimes E}$. Similarly to the $\mathbb{Z}_2$ toric code, the ground state can be constructed using gauge invariant autoregressive neural networks by defining each star as a composite particle and checking Gauss’s law and bond consistency.

Appendix B: $(1+1)$D $\mathbb{Z}_N$ Lattice QED Model

Our approach in Sec. IV A can be applied to the $(1+1)$D $\mathbb{Z}_N$ lattice quantum electrodynamics (QED) model, which is a discretization of the Schwinger model for the continuous-space QED in 1+1 dimensions [67]. The $(1+1)$D $\mathbb{Z}_N$ model takes a similar form as the $(1+1)$D QLM, which has fermions on sites and electric fields on links between two sites. Let $\{ |e_{i,i+1} \rangle \}$ for $1 \leq e_{i,i+1} \leq N$ denote the orthonormal basis on each link $(i, i+1)$. The $(1+1)$D $\mathbb{Z}_N$ gauge theory can take the following form [67]

$$H = -\sum_i [\psi_i^\dagger U_{1,i+1} \psi_{i+1} + \psi_{i+1}^\dagger U_{i,i+1} \psi_i] + m \sum_i (-1)^i \psi_i^\dagger \psi_i + \frac{g^2}{8} \sum_i (V_{i,i+1} - 1)(V_{i,i+1}^\dagger - 1),$$

(B1)

where $U_{1,i+1} |e_{i,i+1} \rangle = |(e_{i,i+1} + 1) \bmod N \rangle$, and $V_{i,i+1} |e_{i,i+1} \rangle = e^{-i2\pi m/N} |e_{i,i+1} \rangle$ for $m = e_{i,i+1}$. The Gauss’s law operator $G_i$ of the model can be written as

$$G_i = e^{i\frac{\pi}{N} \sum_{\psi_i} (\psi_i^\dagger \psi_i + \frac{1}{2}(-1)^i - \frac{1}{2}) V_{i,i+1} V_{i,i-1}^\dagger}$$

(B2)

such that $G_i |\psi \rangle = |\psi \rangle$ for each $i$ [67].

Similarly to the $(1+1)$D QLM, one can construct the gauge invariant autoregressive neural network as Fig. 2 and perform gauge checking with $G_i$ in Eq. B2.

Appendix C: SU(2) Invariant Autoregressive Neural Network for Heisenberg Model

The 1D Heisenberg Model is described as

$$H = \sum_i \sigma_i^x \sigma_{i+1}^x + \sigma_i^y \sigma_{i+1}^y + \sigma_i^z \sigma_{i+1}^z.$$

(C1)

We work in the angular momentum basis $\{|j_1,j_2,j_3,\ldots,j_n\rangle\}$ similarly to [44], instead of the spin basis, to construct an SU(2) invariant autoregressive wave function. Each $j_i$ is the total angular momentum quantum number for spins from 1 to $i$ and $j_n \equiv J$ is the total angular momentum quantum number for all spins. For the ground state of the Heisenberg model, the total angular momentum is zero, so $j_n = 0$. We define the first composite particle as $j_1$ and the $i$’th composite particle as the difference $j_i - j_{i-1}$. Note that this uniquely defines a physical state. We then autoregressively enforce $j_{i+1} \geq 0$ and $j_n = 0$ as gauge checking, to achieve the SU(2) invariant property.

Fig. A2 demonstrates the performance of our SU(2) invariant autoregressive neural network on the Heisenberg model with SU(2) symmetry.

Appendix D: Neural Network Architecture

1. Complex Parameterization

Wave functions are complex in general but both the Transformer network and 1D/2D RNN are real. We use two approaches (Fig. A3)—(a) amplitude-phase and (b) real-imaginary—to parametrize complex wave functions from real neural networks. In both parameterizations, the input configuration $\bar{x}_n$, together with a default configuration $\bar{x}_0$, is embedded (i.e. each state of a composite particle is mapped to a unique vector) before fed into the Transformer or 1D/2D RNN. Certain gauge blocks in an
2. Transformer

The Transformer used in this work (Fig. A4) is the same as the Transformer used in Ref. 8, which can be viewed as the standard Transformer encoder with masked multi-head attention from Ref. 35 but without an additional add & norm layer. The Transformer consists of a standard positional encoding layer, which uses sinusoidal functions to encode the positional information of the embedded input. After positional encoding, the input is fed into the standard masked multi-head attention mechanism. The mask here is crucial for autoregressiveness, as it only allows each site to depend on the previous sites. The output of the attention layer is then passed through a standard feed forward layer. The detailed explanation of the Transformer can be found in Refs. 8 and 35. This transformer is essentially equivalent to the standard PyTorch implementation [46], but was implemented independently because that implementation did not exist at the start of our work.

3. RNN Cells

For all RNNs in this work, we used the gated recurrent unit (GRU) cell [32] (Fig. A5) in PyTorch [46], which takes one input vector $x_k$, the hidden input $h_{k-1}$, and

AR-NN take a default state $\bar{x}_0$ as opposed to any state of the composite particles; the embedded vector of this default state has arbitrary parameters which are trained during optimization.
computes
\[
\begin{align*}
r &= \sigma(W_{x}x_{k} + b_{x} + W_{h}h_{k-1} + b_{h}), \\
z &= \sigma(W_{z}x_{k} + b_{z} + W_{h}h_{k-1} + b_{h}), \\
n &= \tanh(W_{n}x_{k} + b_{n} + r \odot (W_{h}h_{k-1} + b_{h})), \\
h_{k} &= (1 - z) \odot n + z \odot h_{k-1}, \\
y_{k} &= h_{k},
\end{align*}
\]

where \(\sigma\) is the sigmoid function and \(\odot\) means element-wise product.

![Diagram](a) 1D RNN cell. A ResNet (skip connection) [68] is added between the input \(x_{k}\) and output \(y_{k}\). To be noted that \(x_{k}, y_{k}, h_{\text{old}}\) and \(h_{\text{new}}\) has the same dimension. (b) 2D RNN cell with periodic boundary condition. This cell requires four hidden inputs \(h_{\text{old},i=1,2,3,4}\) and generates one hidden output \(h_{\text{new}}\). Skip connections are added for both output \(y_{k}\) and hidden output \(h_{\text{new}}\). The average pooling reduces the hidden output to have the same dimension as each hidden input. To be noted that the dimension of \(x_{k}\) and \(y_{k}\) is four times the dimension of each \(h_{\text{old}}\) and \(h_{\text{new}}\).

We then build 1D and periodic 2D RNN cells (Fig. A6) based on the GRU cell. The 1D RNN cell computes
\[
\begin{align*}
(y_{\text{raw}}, h_{\text{new}}) &= \text{GRUcell}(x_{k}, h_{\text{old}}), \\
y_{k} &= y_{\text{raw}} + x_{k},
\end{align*}
\]
whereas the periodic 2D RNN cell computes
\[
\begin{align*}
(y_{\text{raw}}, h_{\text{raw}}) &= \text{GRUcell}(x_{k}, [h_{\text{old}1}, h_{\text{old}2}, h_{\text{old}3}, h_{\text{old}4}]), \\
[h_{\text{new}1}, h_{\text{new}2}, h_{\text{new}3}, h_{\text{new}4}] &= h_{\text{raw}} + x_{k}, \\
h_{\text{new}} &= \frac{1}{4}(h_{\text{new}1} + h_{\text{new}2} + h_{\text{new}3} + h_{\text{new}4}), \\
y_{k} &= y_{\text{raw}} + x_{k}.
\end{align*}
\]

4. RNNs

With the RNN cells, we can build 1D and periodic 2D RNNs.

The 1D RNN (Fig. A7) has a multi-layer design and shares the same structure as the Pytorch [46] GRU [32]. The embedded input configuration is fed into the cells one at a time through multiple layers and produces a raw output. In our work, the cells at different layers share the weight matrices and bias vectors.

The periodic 2D RNN has a more complicated design to capture the most correlations and can be viewed as a periodic extension of the 2D RNN in Ref. 4. In each layer, the hidden vector \(h\) is passed around according to Fig. A8(a), where each cell receives a maximum number of four hidden vectors and concatenates them according to Fig. A6(b). When the number of hidden vectors received is less than four, zero vectors are used to pad the concatenated hidden vector to the correct length. The configuration is evaluated and sampled in a zigzag 5 path (Fig. A8(b)) to ensure autoregressiveness.

Before the first layer of the periodic 2D RNN, a special concatenation of the embedded input needs to be performed. At each location, the concatenation layer takes the four surrounding inputs (periodically) and concatenates them into a single vector. If any (or all) of the surrounding inputs lies later in the conditioning order in Fig. A8(b), the corresponding input is replaced with a default input \(x_{0}\). For a \(4 \times 4\) 2D input array shown in Fig. A9(a), some concatenation examples are shown in Fig. A9(b). After the first layer, the output of a previous layer can be directly fed into the next layer similar to a regular RNN without any further process.

A multi-layer periodic 2D RNN consists of one input concatenation layer and several 2D RNN layers as shown.
5. Initialization and Optimization

We use different initialization techniques for different models. For the QLM model, the initialization is done through tomography, minimizing $-\log |\psi(x)|^2$ for a desired configuration $x$ (\(\bullet \rightarrow \circ \rightarrow \)) for each unit cell in this case. For the 2D toric code model, we set the weight matrix in the last linear layer to be 0 and the bias such that the wave function for each composite particle is $|0,0\rangle + \sqrt{0.11}|0,0\rangle + \sqrt{0.11}|0,1\rangle + \sqrt{0.11}|1,0\rangle + \sqrt{0.11}|1,1\rangle$, which empirically produces a very low initial energy. We used a transfer learning technique, where we first train the neural network on 32 anyons with $\theta = 0, \pi/4, \pi/2, 3\pi/4, \pi, 5\pi/4, 3\pi/2, 7\pi/4, 2\pi$ for 3000 iterations, and then transfer the model with $\theta$ that is closest to the desired value of $\theta$ for 40 anyons for another 3000 iterations. Similar to the toric code case, when transferred to the larger system, the weight and bias in the last linear layer is dropped and replaced by the initialization described above. In all models, except the last layer, the weights and biases are initialized using PyTorch’s default initialization. For optimization, we used the Adam optimizer with an initial learning rate of 0.01.

Appendix E: Derivation of Stochastic Gradients for Variational and Dynamics Optimization

In Sec. III, we presented stochastic gradients of the variational and dynamics optimizations. This section includes their derivations.

The variational optimization has been widely used and derived many times in other works [1, 4]. Here we present
the derivation for the sake of completeness:

\[
\frac{\partial \langle \psi \rangle}{\partial \theta} | H | \langle \psi \rangle = \langle \psi \rangle H \frac{\partial \langle \psi \rangle}{\partial \theta} + \langle \psi \rangle H \frac{\partial \langle \psi \rangle}{\partial \theta} \]
\[
= 2 \sum_x R \left\{ \frac{\partial \psi^*(x)}{\partial \theta} H \psi(x) \right\} \]
\[
= 2 \sum_x R \left\{ \psi^*(x) H \psi(x) \frac{\partial}{\partial \theta} \log \psi^*(x) \right\} \]
\[
= 2 \sum_x R \left\{ \mathrm{E}_{\text{loc}}(x) \frac{\partial}{\partial \theta} \log \psi^*(x) \right\},
\]

(E1)

where the local energy is \( \mathrm{E}_{\text{loc}}(x) \equiv H \psi(x)/\psi(x) \). We can further control the variance by subtracting from the \( \mathrm{E}_{\text{loc}}(x) \) the average energy \( \mathrm{E}_{\text{avg}} = \sum_{x \in |\psi|^2} \mathrm{E}_{\text{loc}}(x)/N \) over the batch of samples [45] as we did in Sec. III and use the stochastic variance reduced gradient as

\[
\frac{2}{N} \sum_{x \sim |\psi|^2} R \left\{ \left[ \mathrm{E}_{\text{loc}}(x) - \mathrm{E}_{\text{avg}} \right] \frac{\partial}{\partial \theta} \log \psi^*(x) \right\},
\]

(E2)

which has the same expectation as Eq. E1.

For the dynamics optimization gradient, as in Sec. III, we define \( |\Psi_\theta \rangle = (1 + i H \tau) |\psi_{\theta(t+2\tau)} \rangle \) and \( |\Phi \rangle = (1 - i H \tau) |\psi_{\theta(t)} \rangle \), and we drop \( \theta(t) \) and name \( \theta \equiv \theta(t+2\tau) \). We start by splitting the negative log overlap:

\[
- \log \langle \Psi_\theta | \Phi \rangle \langle \Phi | \Psi_\theta \rangle \\
= - \log \langle \Psi_\theta | \Phi \rangle - \log \langle \Phi | \Psi_\theta \rangle + \log \langle \Psi_\theta | \Psi_\theta \rangle + \log \langle \Phi | \Phi \rangle,
\]

(E3)

We then compute the gradient term by term. The first term on the right side of Eq. E3 becomes

\[
\frac{\partial}{\partial \theta} \log \langle \Psi_\theta | \Phi \rangle \\
= \frac{1}{\langle \Psi_\theta | \Phi \rangle} \left\langle \frac{\partial \psi}{\partial \theta} \right\rangle \\
= \frac{1}{\sum_x |\Psi_\theta(x)|^2} \sum_x \frac{\partial \psi^*(x)}{\partial \theta} |\Psi_\theta(x)|^2 \\
= \frac{1}{\sum_x |\Psi_\theta(x)|^2} \sum_x \frac{\partial \psi^*(x)}{\partial \theta} \log |\Psi_\theta(x)|^2 \\
= \frac{1}{N} \sum_{x \sim |\psi|^2} \alpha(x) \frac{\partial}{\partial \theta} \log |\Psi_\theta(x)|^2,
\]

(E4)

where \( \alpha(x) = |\Psi_\theta(x)|^2/|\psi(x)|^2 \) and \( \alpha_{\text{avg}} = \sum_{x \sim |\psi|^2} \alpha(x)/N \) as in Sec. III. The second term on the right side of Eq. E3 is just the complex conjugate of the first term, whereby

\[
\frac{\partial}{\partial \theta} \log \langle \Phi | \Psi_\theta \rangle \\
\approx \frac{1}{N} \sum_{x \sim |\psi|^2} \beta(x) \frac{\partial}{\partial \theta} \log |\Psi_\theta(x)|^2 \\
= \frac{2}{N} \sum_{x \sim |\psi|^2} R \left\{ \beta(x) \frac{\partial}{\partial \theta} \log |\Psi_\theta(x)|^2 \right\},
\]

(E6)

where \( \beta(x) = |\Psi_\theta(x)|^2/|\psi(x)|^2 \) and \( \beta_{\text{avg}} = \sum_{x \sim |\psi|^2} \beta(x)/N \) as in Sec. III. The last term on the right side of Eq. E3 is \( \theta \) independent such that

\[
\frac{\partial}{\partial \theta} \log \langle \Phi | \Phi \rangle = 0.
\]

(E7)
Combining all the terms together,

\[
\frac{\partial}{\partial \theta} \left( -\log \frac{\langle \Psi_\theta | \Phi \rangle \langle \Phi | \Psi_\theta \rangle}{\langle \Psi_\theta | \Psi_\theta \rangle \langle \Phi | \Phi \rangle} \right) 
\approx \frac{2}{N} \sum_{x \sim |\psi_\theta|^2} \Re \left\{ \left[ \frac{\beta(x)}{\beta_{\text{avg}}} - \frac{\alpha(x)}{\alpha_{\text{avg}}} \right] \frac{\partial}{\partial \theta} \log \Psi_\theta^*(x) \right\}. \tag{E8}
\]