PHYSICAL REVIEW B **100**, 020302(R) (2019)

Rapid Communications

Accelerating lattice quantum Monte Carlo simulations using artificial neural networks: Application to the Holstein model

Shaozhi Li,[1,2] Philip M. Dee,[1] Ehsan Khatami,[3] and Steven Johnston[1,4]

[1]*Department of Physics and Astronomy, The University of Tennessee, Knoxville, Tennessee 37996, USA*
[2]*Department of Physics, University of Michigan, Ann Arbor, Michigan 48109, USA*
[3]*Department of Physics and Astronomy, San Jose State University, San Jose, California 95192, USA*
[4]*Joint Institute for Advanced Materials at the University of Tennessee, Knoxville, Tennessee 37996, USA*

(Received 17 May 2019; published 22 July 2019)

Monte Carlo (MC) simulations are essential computational approaches with widespread use throughout all areas of science. We present a method for accelerating lattice MC simulations using fully connected and convolutional artificial neural networks that are trained to perform *local* and *global* moves in configuration space, respectively. Both networks take local spacetime MC configurations as input features and can, therefore, be trained using samples generated by conventional MC runs on smaller lattices before being utilized for simulations on larger systems. This approach is benchmarked for the case of determinant quantum Monte Carlo (DQMC) studies of the two-dimensional Holstein model. We find that both artificial neural networks are capable of learning an unspecified effective model that accurately reproduces the MC configuration weights of the original Hamiltonian and achieve an order of magnitude speedup over the conventional DQMC algorithm. Our approach is broadly applicable to many classical and quantum lattice MC algorithms.

DOI: 10.1103/PhysRevB.100.020302

Introduction. As their full potential becomes apparent, machine-learning algorithms are assuming more prominent roles in the process of scientific discovery. Meanwhile, the boundary lines between industry applications of machine learning, data, and computer science, and other disciplines have blurred. Applications ranging from the high-quality feature extraction from astrophysical images of galaxies [1] to helping with the real-time data analysis of particle accelerators [2–4] to discovering phases of matter [5–7] have emerged.

A series of early studies have underscored the potential for machine learning in the context of condensed matter physics by using artificial neural networks and dimension-reduction techniques to locate phase transitions [5,6], or represent ground states of quantum many-body systems [8]. Machine-learning algorithms have also been employed to help gain insight into classical and quantum systems [9–17] as well as accelerate specific numerical algorithms [18–23]. These applications are not only helping to automate and streamline scientific processes that could take many years to accomplish with more conventional computational approaches, but they are also uncovering previously inaccessible phenomena.

One machine-learning application that has attracted significant attention is in accelerating Monte Carlo (MC) simulations [18,19,24–31]. For example, in the so-called self-learning Monte Carlo (SLMC) method [24], an effective bosonic model is trained to mimic the statistics of the original Hamiltonian. Once trained, the effective model is then used to perform the same simulations much more efficiently. The primary advantage of this approach is that the action of the effective model is often much easier to compute than the action for the full fermion model, thus granting access to larger system sizes. This approach has also been extended to include correlations in both the real-space and imaginary-time domains [25,26]. Despite their power, however, the SLMC methods require that the form of the effective model be known *a priori*. This limitation can be significant as different effective models may be required for the same fermionic model as the model parameters, system size, or simulation temperature changes, and the overall effectiveness of these approaches is severely limited if the wrong effective model is chosen. To overcome this problem, several groups have used artificial neural networks to learn the form of the model in some instances [27,29–31] [e.g., quantum Monte Carlo (QMC) simulations of the Anderson impurity model]; however, generalizing this approach to lattice QMC problems has not yet been achieved. One reason for this is the fact that such problems typically involve thousands of auxiliary spacetime fields and any neural network using that many input features will often generalize poorly.

Here, we show how to design artificial neural networks that can be trained to represent an effective bosonic model for lattice QMC simulations. Inspired by applications of the traveling cluster approximation to spin-fermion models [32,33], we design fully connected and convolutional neural networks that only require information from surrounding auxiliary fields (see Fig. 1) to perform both *local* and *global* moves of the MC configurations. This method does not suffer from the scaling issues restricting other self-learning methods and can be easily generalized across models and parameter regimes without changes in the underlying algorithm, provided the neural networks are versatile enough to learn the effective models. To demonstrate the efficiency of this approach, we apply it to determinant quantum Monte Carlo (DQMC) simulations of the two-dimensional Holstein model. This problem is particularly challenging owing to long autocorrelation times [34], the need for both local and global MC moves to ensure

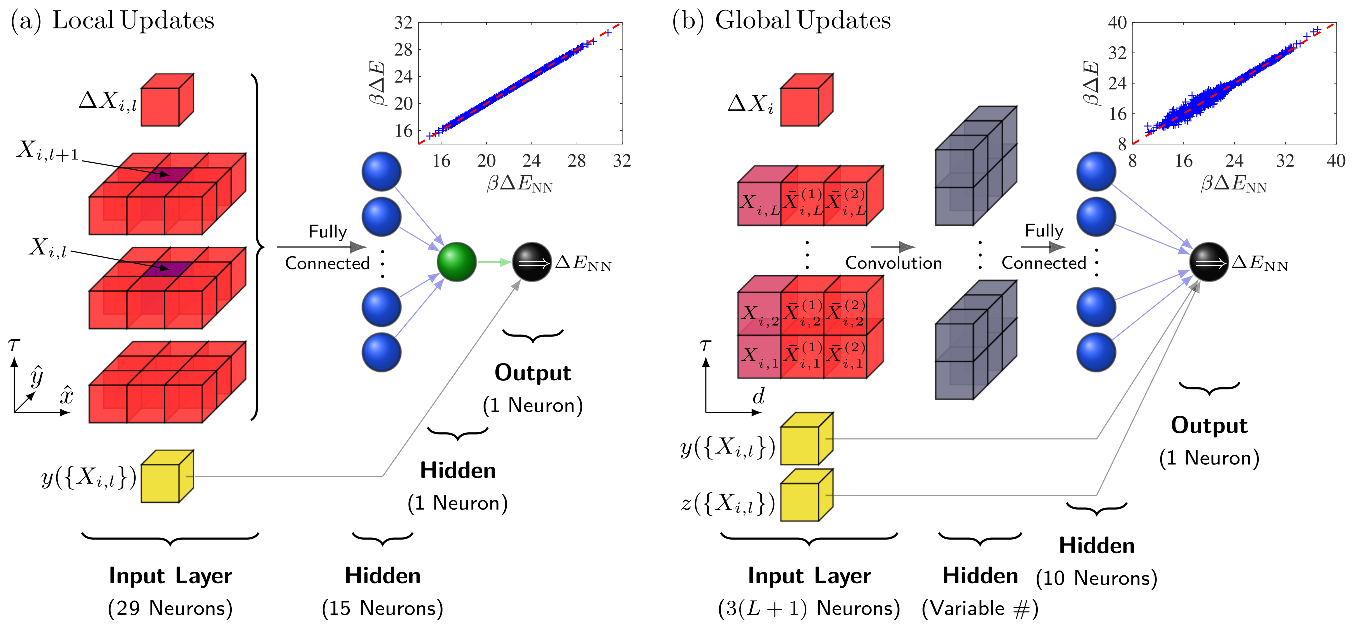

FIG. 1. A sketch of the architecture of the (a) fully connected and (b) convolutional neural networks (CNNs) used to perform local and global updates of the auxiliary fields, respectively. The first and second hidden layers of the fully connected neural network use softplus activation functions $f(x) = \ln(1 + e^x)$, while the output layer uses a linear activation function. The first and second hidden layers of the CNN use sigmoid functions $f(x) = (1 + e^{-x})^{-1}$, while the output layer uses a linear function. The number of neurons in the first hidden layer of the CNN is set by the stride and kernel. A measure of the performance of the two networks is presented in the insets, which compare the predicted $\beta\Delta E_{\rm NN}$ against the exact $\beta\Delta E$ for the fully connected and the CNN, respectively. These results were obtained using networks trained on an $N = 6 \times 6$ cluster, an inverse temperature $\beta = 4.1/t$, filling $\langle\hat{n}\rangle = 1$, $\lambda = t/2$, and $\Omega = t/2$.

ergodicity [35,36], and competition between charge-density-wave (CDW) and superconducting ground states [37] that may require different effective boson models. Reproducing known results, we obtain an order of magnitude of speedup with our algorithm.

Model. The single-band Holstein Hamiltonian [38] is $H = H_0 + H_{\rm lat} + H_{e\text{-ph}}$, where $H_0 = -t\sum_{\langle i,j\rangle,\sigma} c^\dagger_{i,\sigma}c_{j,\sigma} - \mu\sum_{i,\sigma}\hat{n}_{i,\sigma}$, $H_{\rm lat} = \sum_i(\frac{1}{2M}\hat{P}_i^2 + \frac{M\Omega^2}{2}\hat{X}_i^2)$, and $H_{e\text{-ph}} = g\sum_{i,\sigma}\hat{X}_i\hat{n}_{i,\sigma}$. Here, $\langle\cdots\rangle$ denotes a summation over nearest neighbors; $c^\dagger_{i,\sigma}$ ($c_{i,\sigma}$) creates (annihilates) an electron with spin σ on site i; $\hat{n}_{i,\sigma} = c^\dagger_{i,\sigma}c_{i,\sigma}$ is the particle number operator; t is the nearest-neighbor hoping integral; M is the ion mass and Ω is the phonon frequency; $\hat{X}_i$ and $\hat{P}_i$ are the lattice position and momentum operators, respectively; and g is the strength of the e-ph coupling. Throughout, we set $M = t = 1$ as the unit of mass and energy, and we study this Hamiltonian on an $N = N_x^2$ square lattice, where N_x is the linear size of the cluster. To facilitate a direct comparison with a recent state-of-the-art simulation [28], we focus on $\Omega = t/2$ and dimensionless e-ph coupling strength $\lambda = \frac{g^2}{8t\Omega^2} = 0.5$.

Determinant quantum Monte Carlo. DQMC is an auxiliary field, imaginary-time technique that computes expectation values of an observable within the grand canonical ensemble. In a DQMC simulation, the imaginary-time interval $\tau \in [0,\beta]$ is evenly divided into L discrete slices of length $\Delta\tau = \beta/L$ (=0.1 in this work). Using the Trotter approximation, the partition function is then given by $Z = {\rm Tr}(e^{-\Delta\tau LH}) \approx {\rm Tr}(e^{-\Delta\tau H_{e\text{-ph}}}e^{-\Delta\tau(H_0+H_{\rm lat})})^L$. After integrating out the electronic degrees of freedom, the partition function can be reduced to $Z = \int W(\{X\})dX$, where the configuration weight is $W(\{X\}) = e^{-S_{\rm ph}\Delta\tau}\det M^\uparrow \det M^\downarrow$. Here, $\int dX$ is shorthand for integrating over all continuous displacements $X_{i,l}$ defined at each spacetime point (i,l), $M^\sigma = I + B^\sigma_L B^\sigma_{L-1}\cdots B^\sigma_1$, where I is an $N\times N$ identity matrix, and $B^\sigma_l = e^{-\Delta\tau H_{e\text{-ph}}}e^{-\Delta\tau H_0}$, and $S_{\rm ph} = \frac{M}{2\Delta\tau^2}\sum_{i,l}(X_{i,l+1} - X_{i,l})^2 + \frac{M\Omega^2}{2}\sum_{i,l}X^2_{i,l}$ is the lattice's contribution to the total action. Note that B^σ_l matrices for the Holstein model do not depend on spin but are dependent on the fields $X_{i,l}$ through $H_{e\text{-ph}}$. For more details, we refer the reader to Refs. [36,39,40].

Two types of MC updates are needed in the simulation. The first are local updates $X_{i,l} \to X_{i,l} + \Delta X_{i,l}$, which are made at each spacetime point. The second are global or block updates, where the fields for a given site are updated simultaneously at all time slices $X_{i,l} \to X_{i,l} + \Delta X_i$, $\forall l \in [0,L]$. Such block updates are needed to help move phonon configurations out of local minima at low temperatures and large couplings [35,36]. DQMC accepts both kinds of moves with a probability $p = W(\{X'\})/W(\{X\}) \equiv e^{-\beta\Delta E}$, which requires the costly evaluation of matrix determinants. Moreover, since the matrices M^σ depend on the fields, these must also be updated after every accepted change in the phonon fields [see Refs. [19,24,25] and Eq. (1)]. While an efficient update algorithm exists for performing local updates [39], no such algorithm is known for block updates. The computational cost for performing a full sweep of (fast) local updates and block updates is $O(N^3L)$ and $O(N^4L)$ [41], respectively.

To reduce this cost, we train our networks to predict $\beta\Delta E$ appearing in the definition of the configuration weight given only changes in, and local information of, the phonon fields and their expected behavior at large displacements as input. This reduces the total computational complexity of determining whether both kinds of updates will be accepted to the time needed to evaluate the networks, which is $O(1)$ for the case of the simply connected network and $O(L)$ for the convolutional neural network. As with other SLMC methods, we then use the neural networks to propose many MC updates that are ultimately accepted or rejected based on the configuration weights of the original model. While determining this final acceptance probability requires the evaluation of the matrix determinants, this task can be done infrequently enough that a considerable speedup is achieved. Another advantage of our approach is that the networks can be trained using data generated by the conventional DQMC algorithm on small clusters before being generalized to larger systems. In this way, our method combines the flexibility of neural networks with the inexpensive training costs seen in SLMC approaches making use of largely local effective models. We have found that the precise design of the networks does require some experimentation; additional discussions of the physical and practical considerations informing our designs are provided in the Supplemental Material [42].

Local updates. Local updates are performed using a fully connected network with two hidden layers [Fig. 1(a)]. Assuming that the update is proposed at spacetime site (i, l), the learning objective is to predict $\beta\Delta E$ given only $\Delta X_{i,l}$ and the field values at the surrounding spacetime points as input features. Here, we include nearest- and next-nearest-neighbor phonon fields in both space and imaginary time, and neglect long-range correlations. While there is justification for a short-range effective interaction in proximity to the CDW phase at half filling [37], this approximation can also be systematically improved by taking more input features. We have found, however, that next-nearest-neighbor inputs are sufficient. We also supply an additional neuron in the input layer that enforces known behavior at large displacements [28,42].

Global updates. Global updates are performed using a convolutional neural network (CNN) with four layers [Fig. 1(b)], where the objective again is to predict $\beta\Delta E$ given only local information about the phonon fields. Assuming the update occurs at site i, the input layer has three columns of input features: The first contains fields $X_{i,l}$ across all imaginary time slices; the second and third columns contain averages $\bar{X}^{(1)}_{i,l} = \frac{1}{4}\sum_{\langle j\rangle} X_{j,l}$ and $\bar{X}^{(2)}_{i,l} = \frac{1}{4}\sum_{\langle\langle j\rangle\rangle} X_{j,l}$, respectively, at all time slices, where $\langle j\rangle$ and $\langle\langle j\rangle\rangle$ denote nearest- and next-nearest-neighbor sums around site i. The use of $\bar{X}^{(1)}_{i,l}$ and $\bar{X}^{(2)}_{i,l}$ enforces C_4 rotational symmetry and reduces the cost of training the CNN. The convolution operation from the input layer to the first hidden layer is standard [42].

For each set of (fixed) input parameters $\{\beta, \mu, \Omega, g\}$ we train both networks using training examples generated with the conventional DQMC algorithm on a 6×6 cluster. Throughout, we generated 8×10^4 samples, which were randomly partitioned into 6×10^4 training and 2×10^4 test samples. We first show results for their performance; the insets of Figs. 1(a) and 1(b) compare the predicted $\beta\Delta E_{\rm NN}$ against the exact $\beta\Delta E$ values obtained from our test data sets for the local and global updates, respectively. This simulation was performed close to the CDW transition for the model (Fig. 3).

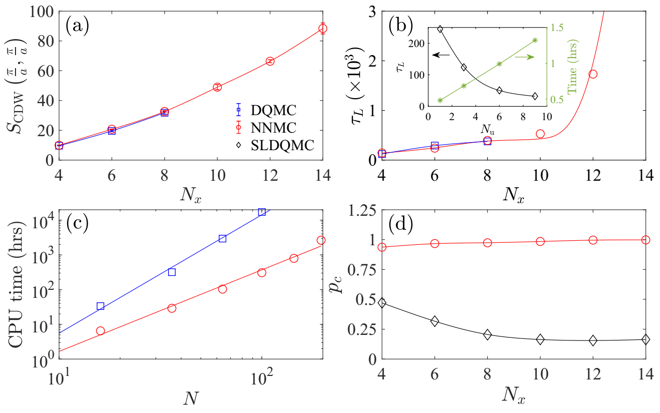

FIG. 2. (a) The CDW structure factor $S_{\rm CDW}(\frac{\pi}{a},\frac{\pi}{a})$ and (b) its autocorrelation time as a function of the linear cluster size N_x obtained with conventional DQMC and NNMC algorithms. The inset shows the reduction of the autocorrelation time and increase of the simulation runtime as the number of update sweeps per Monte Carlo sweep $N_{\rm u}$ is increased. (c) A comparison of CPU time to complete 8×10^4 warmup and 8×10^4 measurement sweeps as a function of N_x using the conventional and neural network sampling schemes. In both cases, we performed global updates randomly at all sites in the cluster after every one full spacetime sweep of local updates. To make a robust comparison between the two methods, we took identical parameters for both sets of simulations. The solid lines are fits to the data of the form $t_{\rm CPU} = \alpha N^z$. (d) The cumulative update ratio of the NNMC algorithm compared against the values achieved using the SLMC method as described in Ref. [28].

Both networks accurately predict the MC configure weights but the fully connected neural network is slightly more accurate. While the accuracy can be systematically improved by taking more input features, we find that the knowledge learned by both networks can be transferred to larger clusters remarkably well based on Fig. 2(a).

Once our networks have been trained and tested, we then define a full MC sweep as consisting of $N_{\rm u}$ complete sweeps of local updates performed at each spacetime point (i, l) using the fully connected neural network, followed by $N_{\rm u}$ sweeps of global updates performed at *every* lattice site i using the CNN. (This sampling procedure differs from the conventional one [36], where global updates are performed on a subset of sites to minimize the total computational cost.) After performing these sweeps, the original field configuration $\{X\}$ is replaced with a newly proposed one $\{X'\}$ in a cumulative update [19,24,25] with a probability $\min[1, p_c]$, where

$$p_c = \frac{W(\{X'\})}{W(\{X\})}\frac{\exp(-\beta E_{\rm NN}[\{X\}])}{\exp(-\beta E_{\rm NN}[\{X'\}])}. \tag{1}$$

Benchmarks. To benchmark the neural network Monte Carlo (NNMC), we performed direct comparisons with the conventional DQMC algorithm for the half-filled model $\langle\hat{n}\rangle = 1$ at $\beta = 4.1/t$, which is close to the CDW transition temperature for this parameter set. We emphasize that both the DQMC and NNMC simulations used the same sampling protocol with $N_{\rm u} = 1$. Figure 2(a) plots the CDW structure factor $S(\frac{\pi}{a},\frac{\pi}{a})$ [42] as a function of the linear cluster size N_x, and demonstrates that the NNMC algorithm accurately reproduces the results of the conventional DQMC algorithm for the

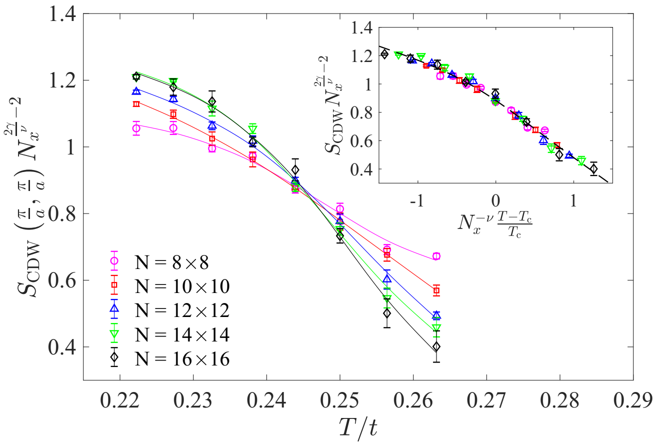

FIG. 3. A finite-size scaling analysis of the $\mathbf{q} = (\frac{\pi}{a}, \frac{\pi}{a})$ CDW structure factor $S_{\mathrm{CDW}}(\mathbf{q})N_x^{2\gamma/\nu-2}$ vs T/t. The CDW transition is in the 2D Ising universality class with critical exponents $\gamma = 1/8$ and $\nu = 1$. The inset shows the collapse of the data when $S_{\mathrm{CDW}}(\mathbf{q})N_x^{2\gamma/\nu-2}$ is plotted vs $N_x^{-\nu}\frac{T-T_c}{T_c}$ with a critical temperature $T_c = 0.244$.

accessible lattice sizes. Figure 2(b) compares the autocorrelation time τ_L of $S_{\mathrm{CDW}}(\mathbf{q})$ for both techniques, which again yields similar results. We note, however, that the autocorrelation time can be reduced significantly by increasing the number of update sweeps N_u that are performed before computing the cumulative update acceptance probability, as shown in the inset of Fig. 2(b).

To address how NNMC reduces the computational cost, we compare the time to solution for both algorithms in Fig. 2(c). Fitting a power law $t_{\mathrm{CPU}} = \alpha N^z$ to the data yields $z = 3.41$ and 2.35 for DQMC and NNMC, respectively, a significant reduction in the scaling. We note that a similar speedup was obtained using SLMC [28]; however, the NNMC does not require the functional form of the effective model to be specified *a priori*. Moreover, the NNMC method is more efficient at generating accepted MC moves, particularly as it is generalized to larger system sizes. We highlight this aspect in Fig. 2(d), which shows the cumulative acceptance ratio p_c obtained using NNMC and compares it with SLMC. As the methods are generalized to larger cluster sizes, p_c decreases for the SLMC method while the NNMC method proposes cumulative moves that are almost always accepted, and becomes more accurate on larger clusters. The decrease of p_c in SLMC is due to the poor performance of the regression model for predicting global updates, which requires a more sophisticated effective model.

We now demonstrate that the NNMC approach can also be used to study the finite-size scaling of the CDW structure factor and obtain the transition temperature in the thermodynamic limit. Figure 3 presents a scaling analysis carried out in the temperature range $\beta = 3.8/t$–$4.5/t$. At the critical point, the finite-size scaling behavior has the form $S_{\mathrm{CDW}}(\frac{\pi}{a}, \frac{\pi}{a})/N_x^2 = N_x^{-2\gamma/\nu} f(N_x^{1/\nu}\frac{T-T_c}{T_c})$, where $\gamma = \frac{1}{8}$ and $\nu = 1$ are the two-dimensional (2D) Ising critical exponents. The critical temperature $T_c/t \approx 0.244$ ($\beta_c = 4.1/t$) is determined by the common intersection point of the curves. The inset replots $S_{\mathrm{CDW}}N_x^{-7/4}$ against $N_x^{1/\nu}\frac{T-T_c}{T_c}$, showing the expected data collapsing to a single curve, consistent with Ref. [28].

Summary and conclusions. We have extended the use of artificial neural networks in self-learning Monte Carlo methods to lattice Monte Carlo simulations. Our approach overcomes many of the scaling issues associated with other SLMC implementations and can be widely applied to classical and quantum Monte Carlo simulations on extended lattices. We then applied this methodology to DQMC studies of the Holstein model. We designed fully connected and convolutional neural networks capable of performing accurate *local* and *global* moves in configuration space. Using this method we are able to reproduce results of the charge-density-wave transition in this model without specifying the effective model in advance.

The success of our network architectures indicates that the effective interactions in the Holstein model are sufficiently short ranged that the lattice dynamics can be captured using relatively small clusters. Prior work examining the effective electron-electron interaction within the Migdal approximation also supports this view [37]. In the future, one could envision examining the structure of the trained networks to study the effective interactions of other Hamiltonians.

Our approach can be generalized for performing machine-learning accelerated lattice Monte Carlo simulations (even the Fermi-Hubbard model), provided that the neural networks are sophisticated enough to learn the effective model in the relevant parameter ranges. In this context, we note that the specific network architectures depicted in Fig. 1 accurately predict the CDW transition but are not well suited for the metal-to-superconducting transition away from half filling (this is also true for the effective model used by the SLMC [28]), and that different network architectures may be needed in that case. Finally, we stress that our approach allows one to compute physical quantities that can be compared with experiments.

Acknowledgements. S.L., P.D., and S.J. are supported by the Scientific Discovery through Advanced Computing (SciDAC) program funded by the U.S. Department of Energy, Office of Science, Advanced Scientific Computing Research and Basic Energy Sciences, Division of Materials Sciences and Engineering. E.K. is supported by the National Science Foundation Grant No. DMR-1609560. An award of computer time was provided by the INCITE program. This research also used resources of the Oak Ridge Leadership Computing Facility, which is a DOE Office of Science User Facility supported under Contract No. DE-AC05-00OR22725, and the Spartan high-performance facility at San Jose State University, which is supported by the NSF under Grant No. OAC-1626645.

[1] K. Schawinski, C. Zhang, H. Zhang, L. Fowler, and G. K. Santhanam, Mon. Not. R. Astron. Soc.: Lett. **467**, L110 (2017).

[2] A. L. Edelen, S. G. Biedron, B. E. Chase, D. Edstrom, S. V. Milton, and P. Stabile, IEEE Trans. Nucl. Sci. **63**, 878 (2016).

[3] A. Radovic, M. Williams, D. Rousseau, M. Kagan, D. Bonacorsi, A. Himmel, A. Aurisano, K. Terao, and T. Wongjirad, Nature (London) **560**, 41 (2018).
[4] A. Scheinker, A. Edelen, D. Bohler, C. Emma, and A. Lutman, Phys. Rev. Lett. **121**, 044801 (2018).
[5] J. Carrasquilla and R. G. Melko, Nat. Phys. **13**, 431 (2017).
[6] L. Wang, Phys. Rev. B **94**, 195105 (2016).
[7] Y. Zhang, A. Mesaros, K. Fujita, S. D. Edkins, M. H. Hamidian, K. Ch'ng, H. Eisaki, S. Uchida, J. C. S. Davis, E. Khatami, and E.-A. Kim, Nature (London) **570**, 484 (2019).
[8] G. Carleo and M. Troyer, Science **355**, 602 (2017).
[9] G. Torlai and R. G. Melko, Phys. Rev. B **94**, 165134 (2016).
[10] K. Ch'ng, J. Carrasquilla, R. G. Melko, and E. Khatami, Phys. Rev. X **7**, 031038 (2017).
[11] D.-L. Deng, X. Li, and S. Das Sarma, Phys. Rev. B **96**, 195145 (2017).
[12] E. P. L. van Nieuwenburg, Y.-H. Liu, and S. D. Huber, Nat. Phys. **13**, 435 (2017).
[13] Y. Zhang and E.-A. Kim, Phys. Rev. Lett. **118**, 216401 (2017).
[14] S. J. Wetzel, Phys. Rev. E **96**, 022140 (2017).
[15] W. Hu, R. R. P. Singh, and R. T. Scalettar, Phys. Rev. E **95**, 062122 (2017).
[16] G. Torlai, G. Mazzola, J. Carrasquilla, M. Troyer, R. Melko, and G. Carleo, Nat. Phys. **14**, 447 (2018).
[17] F. Schindler, N. Regnault, and T. Neupert, Phys. Rev. B **95**, 245134 (2017).
[18] L. Huang and L. Wang, Phys. Rev. B **95**, 035105 (2017).
[19] J. Liu, H. Shen, Y. Qi, Z. Y. Meng, and L. Fu, Phys. Rev. B **95**, 241104(R) (2017).
[20] S.-H. Li and L. Wang, Phys. Rev. Lett. **121**, 260601 (2018).
[21] R. Fournier, L. Wang, O. V. Yazyev, and Q. Wu, arXiv:1810.00913.
[22] J. Nelson, R. Tiwari, and S. Sanvito, Phys. Rev. B **99**, 075132 (2019).
[23] H. Suwa, J. S. Smith, N. Lubbers, C. D. Batista, G.-W. Chern, and K. Barros, Phys. Rev. B **99**, 161107 (2019).
[24] J. Liu, Y. Qi, Z. Y. Meng, and L. Fu, Phys. Rev. B **95**, 041101(R) (2017).
[25] X. Y. Xu, Y. Qi, J. Liu, L. Fu, and Z. Y. Meng, Phys. Rev. B **96**, 041119(R) (2017).
[26] Y. Nagai, H. Shen, Y. Qi, J. Liu, and L. Fu, Phys. Rev. B **96**, 161102(R) (2017).
[27] H. Shen, J. Liu, and L. Fu, Phys. Rev. B **97**, 205140 (2018).
[28] C. Chen, X. Y. Xu, J. Liu, G. Batrouni, R. Scalettar, and Z. Y. Meng, Phys. Rev. B **98**, 041102(R) (2018).
[29] Y. Nagai, M. Okumura, and A. Tanaka, arXiv:1807.04955.
[30] E. M. Inack, G. E. Santoro, L. Dell'Anna, and S. Pilati, Phys. Rev. B **98**, 235145 (2018).
[31] T. Song and H. Lee, arXiv:1901.01501.
[32] A. K. Sen, R. Mills, T. Kaplan, and L. J. Gray, Phys. Rev. B **30**, 5686 (1984).
[33] A. Mukherjee, N. D. Patel, C. Bishop, and E. Dagotto, Phys. Rev. E **91**, 063303 (2015).
[34] M. Hohenadler and T. C. Lang, in *Computational Many-Particle Physics*, edited by H. Fehske, R. Schneider, and A. Weiße (Springer, Berlin, 2008), pp. 357–366.
[35] R. T. Scalettar, R. M. Noack, and R. R. P. Singh, Phys. Rev. B **44**, 10502 (1991).
[36] S. Johnston, E. A. Nowadnick, Y. F. Kung, B. Moritz, R. T. Scalettar, and T. P. Devereaux, Phys. Rev. B **87**, 235133 (2013).
[37] P. M. Dee, K. Nakatsukasa, Y. Wang, and S. Johnston, Phys. Rev. B **99**, 024514 (2019).
[38] T. Holstein, Ann. Phys. **8**, 325 (1959).
[39] S. R. White, D. J. Scalapino, R. L. Sugar, E. Y. Loh, J. E. Gubernatis, and R. T. Scalettar, Phys. Rev. B **40**, 506 (1989).
[40] R. T. Scalettar, N. E. Bickers, and D. J. Scalapino, Phys. Rev. B **40**, 197 (1989).
[41] C.-R. Lee, I.-H. Chung, and Z. Bai, in *Proceedings of 24th IEEE International Parallel and Distributed Processing Symposium* (IEEE, Piscataway, NJ, 2010).
[42] See Supplemental Material at http://link.aps.org/supplemental/10.1103/PhysRevB.100.020302 for detailed algorithms used in this work.