Contracting Arbitrary Tensor Networks: General Approximate Algorithm and Applications in Graphical Models and Quantum Circuit Simulations

Feng Pan,[1,2,*] Pengfei Zhou,[1,2,*] Sujie Li,[1,2,*] and Pan Zhang[1,3,4,†]

[1]*CAS Key Laboratory for Theoretical Physics, Institute of Theoretical Physics, Chinese Academy of Sciences, Beijing 100190, China*
[2]*School of Physical Sciences, University of Chinese Academy of Sciences, Beijing 100049, China*
[3]*School of Fundamental Physics and Mathematical Sciences, Hangzhou Institute for Advanced Study, UCAS, Hangzhou 310024, China*
[4]*International Centre for Theoretical Physics Asia-Pacific, Beijing/Hangzhou, China*

(Received 10 December 2019; revised 30 March 2020; accepted 30 June 2020; published 7 August 2020)

We present a general method for approximately contracting tensor networks with an arbitrary connectivity. This enables us to release the computational power of tensor networks to wide use in inference and learning problems defined on general graphs. We show applications of our algorithm in graphical models, specifically on estimating free energy of spin glasses defined on various of graphs, where our method largely outperforms existing algorithms, including the mean-field methods and the recently proposed neural-network-based methods. We further apply our method to the simulation of random quantum circuits and demonstrate that, with a trade-off of negligible truncation errors, our method is able to simulate large quantum circuits that are out of reach of the state-of-the-art simulation methods.

DOI: 10.1103/PhysRevLett.125.060503

As a powerful method to alleviate the "curse of dimensionality" in high-dimensional modeling and data analysis, the tensor networks find wide applications in many areas of science and technology. In quantum many-body physics, tensor networks on lattices, including the matrix product states (MPS) [1,2] and the projected entangled pair states (PEPS) [3], have great success in the study of strongly correlated systems; in statistical mechanics, calculation of the partition function can be naturally converted to a tensor network contraction problem [4]; in computer science, the number of solutions of constraint satisfaction problems can be computed via tensor networks [5]; in data science, tensor networks and tensor decompositions are important tools for data compression and dimensionality reduction [6]. Recently, tensor network methods have been successfully extended to machine learning, in compressing a neural network [7], giving an efficient image classifier [8] and working as generative models in unsupervised learning [9,10].

Despite its wide use, however, the capability of the tensor networks is so far limited to either small-dimensional systems, where the exact contraction is tractable, or high-dimensional systems only on regular lattices with local interactions, where there exist efficient contraction algorithms, e.g., the renormalization group [4,11–13] and the block decimation [14]. On general systems with long range interactions and irregular connectivity (such as the graphs depicted in Fig. 1), the tensor network method is rarely applied, due to intractability of efficient contraction: to the best of our knowledge, there is no general method that exists for approximately contracting arbitrary tensor networks. This sets limitations on applying tensor networks to many areas, such as graphical models, statistical inference, and machine learning problems.

In this Letter, we aim to break this limitation. We propose a general method for approximately contracting tensor networks on an arbitrary graph, based on a method we term as "MPS calculus": the initial and intermediate tensors produced during the tensor contractions are represented, compressed, and operated using the matrix product states in the canonical form. This allows us to deal with large intermediate tensors, which cannot be stored in the memory in its original form. During the contraction process, we iteratively detect low-rank structures and apply low-rank approximations to reduce computational complexities of

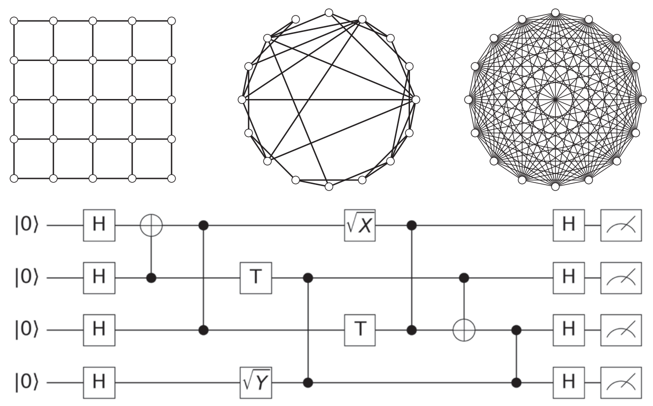

FIG. 1. Illustration of connectivity graph of the tensor networks we aim to contract: two-dimensional lattices, random graphs, fully connected graphs, and those defined by the quantum circuits.

the contraction, using approaches analogous to the density matrix renormalization group (DMRG) [11], until the final result, a scalar Z, is obtained. We show applications of our method in graphical models, where Z represents the normalization factor of the joint distribution of a large number of random variables (i.e., the partition function in physics), and applications in quantum circuit simulations, where Z represents a single amplitude of the quantum circuit.

Contracting arbitrary tensor networks.—Our method relies on two ideas: (1) representing every tensor in the network by a matrix product state in the canonical form and (2) performing low-rank approximations based on the MPS representations during contraction. The matrix product state, also known as the tensor train in mathematics [15], is a one-dimensional tensor network composed of three-way tensors (and matrices in the boundary). A straightforward advantage of MPS is the parameter efficiency: an n-way tensor $\mathcal{A} \in \mathbb{C}^{d^n}$ can be represented by a MPS of virtual bond dimension χ with only $(n-2)d\chi^2 + 2d\chi$ parameters, using, e.g., the DMRG [11]. With a large enough χ, the MPS can faithfully represent the original tensor and hence give an exact result. With limited computational resources, one would restrict the bond dimensions, performed as an approximation to the underlying raw tensor $\mathcal{A}$. Another characteristic of MPS is the canonical form, which can be achieved using QR decompositions or singular value decompositions [16,17]. The first advantage of the canonical form is fixing the gauge degree of freedom, which eliminates the nonuniqueness in representing a raw tensor. More importantly, in the canonical form, the sum of discarded squared singular values corresponds to the loss of $\mathcal{L}_2$ norm of the whole MPS, rather than the local three-way tensor, which allows low-rank approximations on a global scope.

Given a tensor network composed of tensors $\mathcal{A}^{(1)}\ldots\mathcal{A}^{(n)}$ and edges connecting the tensors, the high-level description of our algorithm, MPS calculus, is processed as follows: (1) Convert every tensor to a MPS. (2) If there are no edges left, return; else select an edge (ij) according to a contraction order. (3) "Contract" $\boldsymbol{\mathcal{A}}^{(i)}$ and $\boldsymbol{\mathcal{A}}^{(j)}$, store as $\boldsymbol{\mathcal{A}}^{(i)}$; delete $\boldsymbol{\mathcal{A}}^{(j)}$. (4) If $\boldsymbol{\mathcal{A}}^{(i)}$ connects to $\boldsymbol{\mathcal{A}}^{(k)}$ by two edges, "merge" the edges to a single edge using "swap" operations and low-rank approximations with singular value decomposition (SVD); then go to step 2.

A pictorial representation of the algorithm is sketched in Fig. 2 using a simple example of contracting a fully connected tensor network with five tensors, as shown in panel 1. In panel 2, every tensor that appears in 1 is converted to a MPS in the canonical form. During steps 3–8, edges of the tensor network are contracted one by one, finally producing a scalar in step 9. For further details about the algorithm and order choices, please refer to the Supplemental Material [18] and Refs. [4,11,12,16,17,19–26].

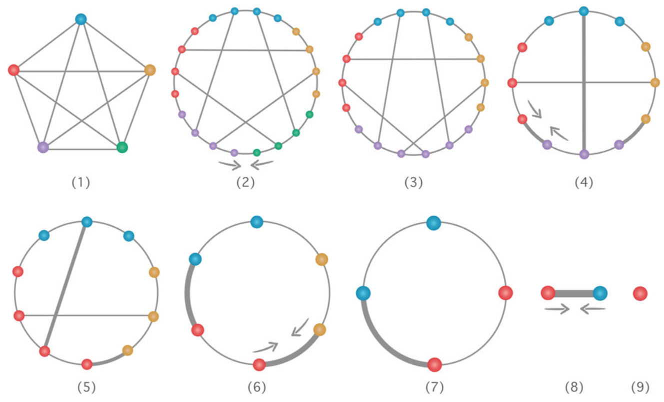

FIG. 2. Pictorial representation of our algorithm in contracting a tensor network with five tensors; see descriptions in main text.

The contract operation is processed by merging two tensors to a single tensor by summing over the common index (say i) of them. Since all of them are MPSs, we need to move the common index i to the tail of the first tensor and to the head of the second tensor, using the swap operations. The swap operation switches the positions of two indices in the original tensor, by swapping two adjacent tensors in the MPS, with a similar functionality as the swap gate in the quantum information. This operation increases entanglements of the MPSs, and the maximum bond dimension could increase to $d\chi$, where χ denotes the virtual bond dimension of the MPSs and d is the dimension of the physical indices. If $d\chi$ is greater than $\hat{\chi}$, the preset limit on the virtual bond dimension, we canonicalize the MPS, then truncate the bond dimension to $\hat{\chi}$ during the singular value decomposition. An example of swap and contract are illustrated using tensor diagram notations in Fig. 3, where the scissor symbol indicates truncating of the dimension in the diagonal matrix.

After the contraction, the obtained tensor could have two indices, say j (with bond dimension d_j) and k (with bond dimension d_k) linked together to another tensor, due to existence of a triangle with three end tensors. In this case, we move indices j and k to adjacent positions using the swap operations and merge the two corresponding tensors to a three-way tensor with a larger physical bond dimension d_jd_k. If it exceeds $\hat{D}$, the preset maximum physical bond dimension, we canonicalize both tensors, then do SVD together with a truncation on singular values to reduce the bond dimension from d_jd_k to $\hat{D}$. The process is illustrated in Fig. 3(c).

The operations swap, contraction, and merge are repeated until the overall tensor network is finally contracted to a scalar Z. Our algorithm takes two parameters, the maximum physical bond dimension $\hat{D}$ and the maximum virtual bond dimension $\hat{\chi}$ of the MPSs. The space complexity of the algorithm is bounded above by $\mathcal{O}(\hat{D}\hat{\chi}^2)$ and the time complexity is dominated by singular value

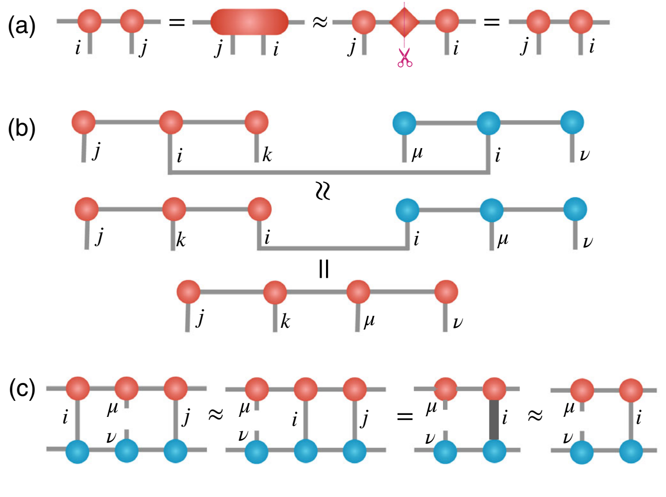

FIG. 3. Illustration of the (a) swap, (b) contract, and (c) merge operations. The scissor symbol indicates truncation of the singular values.

decompositions adopted in the swap operations, which is $\mathcal{O}(\hat{D}^3\hat{\chi}^3)$. Apparently, it is a polynomial algorithm that is able to contract arbitrary tensor networks with a limited amount of computational resources. Moreover, our method enjoys an efficient approximation scheme analogous to the DMRG method, which allows dynamically adjusting dimensions of the tensors. In the following text, we will give applications of our algorithms, the inference and learning in the graphical models, and the simulation of quantum circuits, to empirically evaluate our method.

We noticed that in [27] the author's have proposed a general tensor network contraction algorithm by representing large intermediate tensors using the tree tensor network and reducing loop length using local singular value decompositions. Compared with [27], our method is capable of using larger bound dimensions because the MPS has lower space complexity than the tree tensor network. Moreover, the canonical form of MPS allows more effective approximations.

Applications to graphical models.—Graphical models are important tools for representing joint probability distributions over a large number of random variables that interact with each other and find important applications in many fields in science and engineering. Without loss of generality, in this Letter we use the classic example of the graphical model, the Ising model, and spin glasses in the statistical physics to demonstrate the power of our method. In this problem, the joint probability of n spins $\mathbf{s} \in \{\pm 1\}^n$ follows the Boltzmann distribution $P(\mathbf{s}) = (1/Z)\exp[-\beta E(\mathbf{s})]$, where $E(\mathbf{s})$ is the energy function of a configuration $\mathbf{s}$, β is the inverse temperature, and Z is the partition function. Given a problem instance, an essential problem is computing the free energy $F = -(1/\beta)\ln Z$. However, this problem belongs to the class of #P problems, hence it is hopeless to find polynomial algorithms for solving it exactly. In physics, many approximate algorithms have been developed. These include Markov chain Monte Carlo methods [28] and mean-field methods that parametrize a variational distribution by minimizing the variational free energy. Recently, in [29], the mean-field methods have been extended by employing the autoregressive neural networks as a variational distribution, which, in principle, has a strong expressive power.

Any probability distribution over discrete variables is a tensor, thus every graphical model can be converted to a tensor network by introducing copy tensors on each node of the graph and matrices (or tensors) on each edge (or multibody factor) of the (factor) graph. The computation of the partition function Z naturally translates to contraction of the tensor network defined exactly on the same graph. As an example, consider the celebrated pairwise Ising spin glass model with n variables: its energy function is defined as $E(\mathbf{s}) = -\sum_{(ij)\in\mathcal{E}} J_{ij}s_is_j$, with $\mathcal{E}$ denoting a set of edges and J_{ij} denoting couplings between two spins i and j. The partition function can be written formally as

$$Z = \sum_{\mathbf{s}} \prod_{(ij)\in\mathcal{E}} e^{\beta J_{ij}s_is_j} = \mathbf{Tr}(\mathcal{A}^{(1)} \times \mathcal{A}^{(2)} \times \cdots \times \mathcal{A}^{(n)}), \quad (1)$$

where the symbol $\times$ represents contraction of tensors $\{\mathcal{A}^{(i)}\}$, each of which is given by contracting a copy tensor with matrices defined on the edges connected to node i,

$$\mathcal{A}^{(i)} = \mathcal{I}_{d_i\times d_i} \times \mathbf{B}_{j\in\partial i} \times \mathbf{B}_{k\in\partial i} \times \cdots \times \mathbf{B}_{l\in\partial i}.$$

Here $\mathcal{I}_{d_i\times d_i}$ is a copy tensor, i.e., a diagonal tensor with order equal to the degree (number of neighbors) d_i of node i, with one on the diagonal entries and zero on the other entries. ∂i denotes the set of neighbors of node i, and the matrix $\mathbf{B}_{j\in\partial i}$ is a 2×2 matrix with $[\cosh(\beta J_{ij})/2]^{1/2} + [\sinh(\beta J_{ij})/2]^{1/2}$ on the diagonal and $[\cosh(\beta J_{ij})/2]^{1/2} - [\sinh(\beta J_{ij})/2]^{1/2}$ on the off-diagonal entries.

After converting the graphical model to tensor network, our method directly applies to computing free energy of the problem defined on arbitrary graphs. Observe that our algorithm is exact when the graph is a tree, because, by minimizing the size of the intermediate tensor, it performs variable eliminations iteratively on leaves of the tree and hence reduces to the belief propagation algorithm. On other graphs, our algorithm might generate truncation error ϵ_{SVD}. Empirically we observe that the error ϵ_{SVD} is several magnitudes smaller than the error of the obtained free energy ϵ_F, but so far it is not clear to us how to relate the two errors analytically. We subject to numerical experiments to demonstrate the performance of our algorithm.

The experiments are carried out using the Ising models and spin glasses on various topologies, including 2D lattices, random graphs, small world graphs, and complete

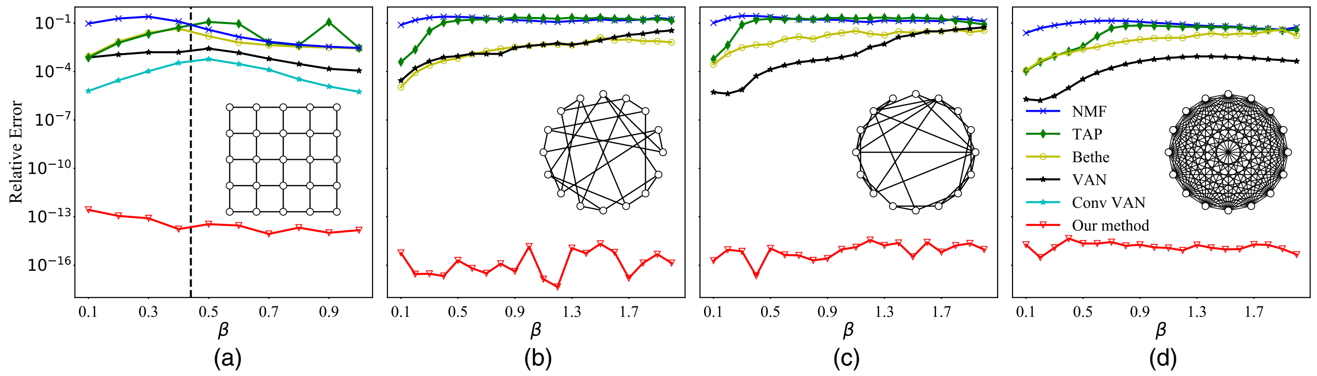

FIG. 4. Relative errors of the free energy to exact solutions obtained by different methods on various models. Insets: illustrations of the underlying connectivity graph with smaller sizes. (a) Ferromagnetic Ising model on a 16×16 square lattice; the exact solutions are given by [30], and the vertical dashed line represents the phase transition of an infinite system. (b) Ising spin glass model on random regular graphs of 80 nodes with degree $k = 3$; couplings J_{ij} are drawn from normal distribution with zero mean and unit variance. (c) Ising spin glass model on the Watts-Strogatz graphs of 70 nodes with average degree $c = 4$ and rewiring probability $p = 0.4$. The exact solutions are given by enumerating all configurations of feedback set of graphs [31]. (d) The Sherrington-Kirkpatrick model with $n = 20$ spins; exact solutions are given by enumerating 2^n configurations. Data points are averaged over 10 random instances.

graphs. Our results on error of free energies are compared against mean-field methods, including the naïve mean-field (NMF), Thouless-Anderson-Palmer equations (TAP), belief propagation, and the neural-network-based variational autoregressive networks (VAN). On the 2D lattice without the external field, the graph is planar, so there are exact solutions [30]. Whereas on the other graphs, we adopt the exact (carefully designed) exponential algorithms [31] (in a reasonable time) to compute exact free energy values for the evaluations.

The results are shown in Fig. 4. We can see that, in all experiments, our method outperforms all mean-field methods and the neural-network-based methods, to a large margin. In regular random graphs, small world networks, and the Sherrington-Kirkpatrick model, our accuracy is only limited by the machine precisions (10^{-16}). In the experiments, we choose $\hat{D} = 50$ and $\hat{\chi} = 500$, and the computational time on each instance is of a few seconds. Empirically, our method is faster than the mean-field methods and the neural-network-based methods. More results about the dependence of the bond dimensions and the computational time can be found in the Supplemental Material [18]. Moreover, it is worth noting that combining with the autodifferential for tensor networks [32] immediately gives our method an ability to perform learning tasks using graphical models. In the Supplemental Material [18], we give an example of using our method to learn a generative model [33–43] on hand-written digits of the MNIST dataset [44].

Application to quantum circuit simulations.—The problem of computing free energy of graphical models is similar to the problem of computing single amplitude estimates of a superconducting quantum circuit [45], which can be treated as a graphical model with complex couplings. Classical simulation of quantum circuits is important for verifying and evaluating the computational advances of quantum computers [20,22–24,46,47]. However, the near-term noisy intermediate-scale quantum circuits (including Google's recently announced "supremacy circuit" [48]) are not perfect: each operation of them contains a small error. Thus, an important open question is whether approximate simulations of quantum circuits could beat the noisy quantum device. Answering this question apparently requires advanced studies of approximate algorithms for simulating quantum circuits.

Our method directly applies to approximate single-amplitude simulation of quantum circuits with any kind of connectivities, such as two-dimensional lattice [23,24] and random regular graphs, as considered in the quantum approximate optimization algorithm [49], after converting the initial state, the measurement qubit string, and the gates into tensors. The key difference between our method and existing methods for quantum circuit simulation is that, by detecting low-rank structures in the circuit, our method heavily reduces the computational complexity. Although this introduces SVD truncation errors, we will illustrate that, at least in the shallow circuits, the error is almost negligible. We perform experiments using standard random circuits on two-dimensional lattices [22–24], which iteratively apply single-qubit gates and two-qubit controlled Z gates to the initial $|0, 0, \ldots, 0\rangle$ state, and finally measure the amplitude of a specific qubit string. The generation protocol is described in detail in the Supplemental Material [18]. We evaluate the performance of our method against the recently developed state-of-the-art exact tensor contraction method [24], which has a precisely predictable space and time complexity. With depth $d = 8$, our algorithm can handle circuits with at most $40 \times 40 = 1600$ qubits with SVD accumulated truncation error $\epsilon_{\mathrm{SVD}} \leq 10^{-12}$ on a workstation with 64 GB memory in an hour.

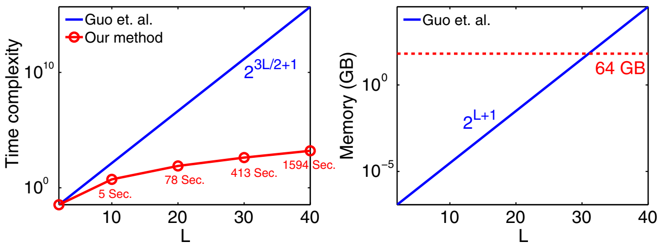

FIG. 5. Computational time and memory usage of our algorithm in simulating random quantum circuits with depth $d = 8$, comparing with the exact tensor network method of Guo *et al.* [24]. We ran our algorithm on a workstation with 64 GB memory (as indicated by the red dashed line). The blue lines with formulas in the figure represent the precise time and space complexity of the exact algorithm [24]. The memory usage is calculated based on double precision complex number. Each red point in the left panel is averaged over 10 random circuits, and the error bars are much smaller than the symbol size.

As compared in Fig. 5, the computational complexity of our method is much lower than the method of [24]. The right panel of Fig. 5 indicates that the method of [24] already costs at least 64 GB memory for storing the largest intermediate tensor with $L = 31$ and further requires 32 TB memory for handling $L = 40$. We note that so far our algorithm cannot handle the circuit with a large depth such as Google's circuit [48] with a small SVD error, because the current implementation of our algorithm only works on a single workstation; this prevents us from using a large bond dimension.

Discussions.—We have presented an algorithm for contracting arbitrary tensor networks, based on the matrix product state for automatic detecting of low-rank structures inside the tensor networks during the contraction process. We have demonstrated advances of our method in the inference and learning in graphical models and in simulation of shallow quantum circuits. The particular strength of our method is able to find the internal low-entanglement structures automatically in the irregular tensor networks. The MPS representation of tensors in our method naturally supports distributed storage. It is interesting to see how large a quantum circuit we can simulate if a supercomputer is accessible to our algorithm. Another interesting development is exploring learning with quantum circuits using our scheme and backpropagation. We hope more advanced arbitrary tensor network contraction methods inspired by our approach could fully release the numerical computational power of tensor networks to wider applications in science and engineering. A PYTHON implementation of our method is available at [50].

We thank Song Cheng, Haijun Liao, Chuang Wang, Lei Wang, Tao Xiang, Zhiyuan Xie, Haijun Zhou, and the BFS Tensor community for inspiring discussions, and Jinguo Liu for generating quantum circuits using *Yao.jl* [51]. P. Z. is supported by Project No. QYZDB-SSW-SYS032 of Chinese Academy of Sciences and the National Natural Science Foundation of China under Grants No. 11947302 and No. 11975294. Part of the computation was carried out at the HPC cluster of ITP,CAS.

[*]These authors contributed equally to this work.
[†]panzhang@itp.ac.cn

[1] G. Vidal, Efficient Simulation of One-Dimensional Quantum Many-Body Systems, Phys. Rev. Lett. **93**, 040502 (2004).
[2] F. Verstraete, J. J. Garcia-Ripoll, and J. I. Cirac, Matrix Product Density Operators: Simulation of Finite-Temperature and Dissipative Systems, Phys. Rev. Lett. **93**, 207204 (2004).
[3] F. Verstraete and J. I. Cirac, Renormalization algorithms for quantum-many body systems in two and higher dimensions, arXiv:cond-mat/0407066.
[4] M. Levin and C. P. Nave, Tensor Renormalization Group Approach to Two-Dimensional Classical Lattice Models, Phys. Rev. Lett. **99**, 120601 (2007).
[5] S. Kourtis, C. Chamon, E. R. Mucciolo, and A. E. Ruckenstein, Fast counting with tensor networks, SciPost Phys. **7**, 060 (2019).
[6] A. Cichocki, N. Lee, I. Oseledets, A.-H. Phan, Q. Zhao, D. P. Mandic *et al.*, Tensor networks for dimensionality reduction and large-scale optimization: Part 1 low-rank tensor decompositions, Found. Trends® Mach. Learn. **9**, 249 (2016).
[7] Z.-F. Gao, S. Cheng, R.-Q. He, Z. Y. Xie, H.-H. Zhao, Z.-Y. Lu, and T. Xiang, Compressing deep neural networks by matrix product operators, Phys. Rev. Research **2**, 023300 (2020).
[8] E. Stoudenmire and D. J. Schwab, Supervised learning with tensor networks, in *Advances in Neural Information Processing Systems* (Curran Associates, Inc., Barcelona, Spain, 2016), pp. 4799–4807.
[9] Z.-Y. Han, J. Wang, H. Fan, L. Wang, and P. Zhang, Unsupervised Generative Modeling Using Matrix Product States, Phys. Rev. X **8**, 031012 (2018).
[10] S. Cheng, L. Wang, T. Xiang, and P. Zhang, Tree tensor networks for generative modeling, Phys. Rev. B **99**, 155131 (2019).
[11] S. R White, Density Matrix Formulation for Quantum Renormalization Groups, Phys. Rev. Lett. **69**, 2863 (1992).
[12] Z. Y. Xie, J. Chen, M. P. Qin, J. W. Zhu, L. P. Yang, and T. Xiang, Coarse-graining renormalization by higher-order singular value decomposition, Phys. Rev. B **86**, 045139 (2012).
[13] D. Adachi, T. Okubo, and S. Todo, Anisotropic tensor renormalization group, arXiv:1906.02007.
[14] R. Orús and G. Vidal, Infinite time-evolving block decimation algorithm beyond unitary evolution, Phys. Rev. B **78**, 155117 (2008).
[15] I. V. Oseledets, Tensor-train decomposition, SIAM J. Sci. Comput. **33**, 2295 (2011).

[16] U. Schollwöck, The density-matrix renormalization group in the age of matrix product states, Ann. Phys. (Amsterdam) **326**, 96 (2011).

[17] R. Orús, A practical introduction to tensor networks: Matrix product states and projected entangled pair states, Ann. Phys. (Amsterdam) **349**, 117 (2014).

[18] See Supplemental Material at http://link.aps.org/supplemental/10.1103/PhysRevLett.125.060503 for detailed descriptions of our algorithm, more experimental results, and applications of our method to generative models.

[19] Z.-Y. Xie, H.-C. Jiang, Q. N. Chen, Z.-Y. Weng, and T. Xiang, Second Renormalization of Tensor-Network States, Phys. Rev. Lett. **103**, 160601 (2009).

[20] I. L. Markov and Y. Shi, Simulating quantum computation by contracting tensor networks, SIAM J. Comput. **38**, 963 (2008).

[21] E. F. Dumitrescu, A. L. Fisher, T. D. Goodrich, T. S. Humble, B. D. Sullivan, and A. L. Wright, Benchmarking treewidth as a practical component of tensor network simulations, PLoS One **13**, e0207827 (2018).

[22] J. Chen, F. Zhang, M. Chen, C. Huang, M. Newman, and Y. Shi, Classical simulation of intermediate-size quantum circuits, arXiv:1805.01450.

[23] S. Boixo, S. V. Isakov, V. N. Smelyanskiy, and H. Neven, Simulation of low-depth quantum circuits as complex undirected graphical models, arXiv:1712.05384.

[24] C. Guo, Y. Liu, M. Xiong, S. Xue, X. Fu, A. Huang, X. Qiang, P. Xu, J. Liu, S. Zheng, H.-L. Huang, M. Deng, D. Poletti, W.-S. Bao, and J. Wu, General-Purpose Quantum Circuit Simulator with Projected Entangled-Pair States and the Quantum Supremacy Frontier, Phys. Rev. Lett. **123**, 190501 (2019).

[25] J. Gray and S. Kourtis, Hyper-optimized tensor network contraction, arXiv:2002.01935.

[26] C. Huang, F. Zhang, M. Newman, J. Cai, X. Gao, Z. Tian, J. Wu, H. Xu, H. Yu, B. Yuan *et al.*, Classical simulation of quantum supremacy circuits, arXiv:2005.06787.

[27] A. Jermyn, Automatic contraction of unstructured tensor networks, SciPost Phys. **8**, 005 (2020).

[28] F. Wang and D. P. Landau, Efficient, Multiple-Range Random Walk Algorithm to Calculate the Density of States, Phys. Rev. Lett. **86**, 2050 (2001).

[29] D. Wu, L. Wang, and P. Zhang, Solving Statistical Mechanics Using Variational Autoregressive Networks, Phys. Rev. Lett. **122**, 080602 (2019).

[30] M. Kac and J. C. Ward, A combinatorial solution of the two-dimensional Ising model, Phys. Rev. **88**, 1332 (1952).

[31] F. Pan, P. Zhou, H.-J. Zhou, and P. Zhang, Solving statistical mechanics on sparse graphs with feedback set variational autoregressive networks, arXiv:1906.10935.

[32] H.-J. Liao, J.-G. Liu, L. Wang, and T. Xiang, Differentiable Programming Tensor Networks, Phys. Rev. X **9**, 031041 (2019).

[33] Y. LeCun, Y. Bengio, and G. Hinton, Deep learning, Nature (London) **521**, 436 (2015).

[34] D. P. Kingma and M. Welling, Auto-encoding variational Bayes, arXiv:1312.6114.

[35] L. Dinh, D. Krueger, and Y. Bengio, Nice: Non-linear independent components estimation, arXiv:1410.8516.

[36] L. Dinh, J. Sohl-Dickstein, and S. Bengio, Density estimation using real nvp, arXiv:1605.08803.

[37] D. Rezende and S. Mohamed, Variational inference with normalizing flows, in *Proceedings of the 32nd International Conference on Machine Learning*, edited by F. Bach and D. Blei, Proceedings of Machine Learning Research Vol. 37 (Lille, France, 2015), pp. 1530–1538.

[38] B. Uria, M.-A. Côté, K. Gregor, I. Murray, and H. Larochelle, Neural autoregressive distribution estimation, J. Mach. Learn. Res. **17**, 1 (2016).

[39] A. Van Oord, N. Kalchbrenner, and K. Kavukcuoglu, Pixel recurrent neural networks, in *Proceedings of The 33rd International Conference on Machine Learning*, edited by M. F. Balcan and K. Q. Weinberger, Proceedings of Machine Learning Research Vol. 48 (PMLR, New York, 2016), pp. 1747–1756.

[40] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, Generative adversarial nets, in *Advances in Neural Information Processing Systems* (Curran Associates, Inc., Montreal, Canada, 2014), pp. 2672–2680.

[41] D. H. Ackley, G. E. Hinton, and T. J. Sejnowski, A learning algorithm for Boltzmann machines, Cogn. Sci. **9**, 147 (1985).

[42] Y. A. LeCun, L. Bottou, G. B. Orr, and K.-R. Müller, Efficient backprop, in *Neural Networks: Tricks of the Trade* (Springer, New York, 2012), pp. 9–48.

[43] D. P. Kingma and J. Ba, Adam: A method for stochastic optimization, in *International Conference on Learning Representations, San Diego, CA, USA* (2015).

[44] Y. LeCun, C. Cortes, and C. J. C. Burges, The MNIST database of handwritten digits (1998), http://yann.lecun.com/exdb/mnist.

[45] S. Boixo, S. V. Isakov, V. N. Smelyanskiy, R. Babbush, N. Ding, Z. Jiang, M. J. Bremner, J. M. Martinis, and H. Neven, Characterizing quantum supremacy in near-term devices, Nat. Phys. **14**, 595 (2018).

[46] J. Napp, R. L. La Placa, A. M. Dalzell, F. G. S. L. Brandao, and A. W. Harrow, Efficient classical simulation of random shallow 2D quantum circuits, arXiv:2001.00021.

[47] R. Schutski, D. Lykov, and I. Oseledets, Adaptive algorithm for quantum circuit simulation, Phys. Rev. A **101**, 042335 (2020).

[48] F. Arute, K. Arya, R. Babbush, D. Bacon, J. C. Bardin, R. Barends, R. Biswas, S. Boixo, F. G. S. L. Brandao, D. A. Buell *et al.*, Quantum supremacy using a programmable superconducting processor, Nature (London) **574**, 505 (2019).

[49] E. Farhi, J. Goldstone, and S. Gutmann, A quantum approximate optimization algorithm, arXiv:1411.4028.

[50] https://github.com/panzhang83/catn.

[51] X.-Z. Luo, J.-G. Liu, P. Zhang, and L. Wang, Yao. jl: Extensible, efficient framework for quantum algorithm design, arXiv:1912.10877.