Can neural quantum states learn volume-law ground states?

Giacomo Passetti,[1] Damian Hofmann,[2] Pit Neitemeier,[1]
Lukas Grunwald,[1, 2] Michael A. Sentef,[2] and Dante M. Kennes[1, 2]

[1]*Institut für Theorie der Statistischen Physik, RWTH Aachen University and
JARA-Fundamentals of Future Information Technology, 52056 Aachen, Germany*
[2]*Max Planck Institute for the Structure and Dynamics of Matter, Center for
Free-Electron Laser Science (CFEL), Luruper Chaussee 149, 22761 Hamburg, Germany*
(Dated: December 6, 2022)

We study whether neural quantum states based on multi-layer feed-forward networks can find ground states which exhibit volume-law entanglement entropy. As a testbed, we employ the paradigmatic Sachdev-Ye-Kitaev model. We find that both shallow and deep feed-forward networks require an exponential number of parameters in order to represent the ground state of this model. This demonstrates that sufficiently complicated quantum states, although being physical solutions to relevant models and not pathological cases, can still be difficult to learn to the point of intractability at larger system sizes. This highlights the importance of further investigations into the physical properties of quantum states amenable to an efficient neural representation.

Introduction.— The exponential complexity of representing general quantum many-body states is a key challenge in computational quantum physics. To simulate systems beyond small sizes tractable by exact diagonalization methods, it is necessary to find an efficient representation of quantum states of interest. This is made possible by the fact that physically relevant states usually possess a high degree of structure, compared to an arbitrary Hilbert space vector. As a prominent example, ground states of local, gapped Hamiltonians exhibit an area law of the entanglement entropy, i.e., an entanglement entropy that scales like the boundary of the subregion instead of its volume. For systems with a low dimensionality, typically 1D, the area law allows for an efficient representation of the wave function as a matrix product state, which can be simulated by algorithms such as the density matrix renormalization group (DMRG) [1–5].

However, many quantum states of physical interest display a volume law scaling of the entanglement entropy [6], for which generally applicable efficient representations are not known to this date. One class of variational approximations that has been studied to overcome this challenge are neural quantum states (NQS) [7], which are based on an artificial-neural-network representation of the wave function's probability amplitudes [8–10] and have shown promising results for the study of discrete lattice models even beyond one dimension [11–19]. Notably, it has been shown that, in specific settings, a shallow NQS ansatz is able to efficiently represent a volume-law quantum state [20], suggesting that this method could complement tensor networks techniques for the purpose of uncovering the physics of highly entangled states. Nevertheless, while for matrix product states and more general tensor-network-based approaches it is known how the entanglement scaling limits the representation capabilities of the ansatz [3], there is so far no analogous physical property that directly relates to the ability of an NQS to learn a given quantum state. Universal approximation theorems, which have been proven for several broad classes of neural networks, guarantee that, in the

limit of infinite network size, a neural network ansatz can theoretically represent any continuous function to arbitrary precision [21–24]. Still, these results do not provide bounds on the scaling of the required number of parameters with the system size. For practical applications of NQS, it is thus a central question to determine which classes of quantum many-body states can be efficiently represented, that are impossible to tackle with other established variational ansätze.

In this Letter, we investigate the capabilities of NQS based on shallow and deep feed-forward neural networks (FFNNs) to represent ground states of the Sachdev-Ye-Kitaev (SYK) model [25–27], which is a paradigmatic model for quantum chaos and non-Fermi liquid behavior [28] and which features a volume-law entanglement in the ground state [29]. We present a systematic study of the representation accuracy achieved by the FFNN in dependence of the network hyperparameters, using state-of-the-art training protocols. We find an exponential dependence on the system size for the number of network parameters required to learn the SYK ground state. This key result clearly demonstrates that present-day NQS wavefunctions are not a universal remedy for all quantum many-body problems of interest.

Model.— The SYK model describes strongly correlated fermions on L sites and is defined by the Hamiltonian [25–27]

$$\hat{H}_{\text{SYK}}(J) = \frac{1}{(2L)^{3/2}} \sum_{ijkl} J_{ij;kl} \, \hat{c}_i^\dagger \hat{c}_j^\dagger \hat{c}_k \hat{c}_l, \qquad (1)$$

where $\hat{c}_i^{(\dagger)}$, $i \in \{1, \dots L\}$ are fermionic ladder operators. The vertices $J_{ij;kl}$ have the symmetry $J_{ij;kl}^* = J_{lk;ji}$ and $J_{ij;kl} = -J_{ji;kl}$ and are random, uncorrelated, all-to-all couplings that are drawn from a Gaussian unitary ensemble (GUE) [30] with mean $\mathbb{E}[J_{ij;kl}] = 0$ and variance $\mathbb{E}[|J_{ij;kl}|^2] = 1$ [28]. Consequently, quantities of physical interest are expectation values over the ensemble of couplings J, which is evaluated after the quantum-expectation value. The ground state of the SYK model
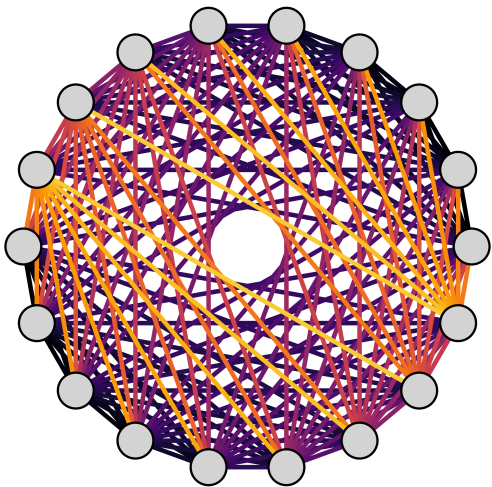
FIG. 1. Cartoon representation of the SYK model. Gray circles represent lattice sites and every different colour shown has two corresponding lines in total connecting four sites. Each color represents one element of the coupling matrix $J_{ij;kl}$ of the SYK model defined by Eq. (1).

describes a strongly correlated non-Fermi liquid without quasi-particle excitations [28], that exhibits volume-law entanglement entropy [31, 32]. In the thermodynamic limit the model becomes self-averaging and exactly solvable, but despite this exact solvability, the ground state is not a Gaussian state, i.e. not a product of single particle wave-functions [33]. At finite sizes, particularly studied in the context of quantum chaos [34] and experimental realizations [35], no exact solutions are known. Here the model can be analyzed by employing approximations, or numerically, by drawing a set of couplings $\{J^{(n)}\}_{n=1}^{N}$ from the GUE, constructing the corresponding Hamiltonians $\hat{H}_{\text{SYK}}(J^{(n)})$, and solving for the ground states $|\Psi_{\text{GS}}(J^{(n)})\rangle$. Finally, the properties of interest, such as expectation values, are averaged over the hence obtained ground state ensemble. Because of the self-averaging property of the SYK model, it suffices to evaluate expectation values for a single realization of J in the thermodynamic limit [28].

Network architecture.— We use a fully-connected FFNN [Figs. 2(a), 3(a)]

$$F(x) = f^{(\mu)} \circ \cdots \circ f^{(1)}(x),$$
$$f^{(l)}(y) = \phi(W^{(l)}y + b^{(l)}) \tag{2}$$

which is a composition of μ layers $f^{(l)}$, each applying an affine transformation and a scaled exponential linear unit (SELU) activation function ϕ [36] as pointwise non-linearity. Each layer has αL neurons, where α is the fixed hidden unit density. The output of the final layer is reduced to a (scalar) log-probability amplitude with respect to the computational basis $\{|x\rangle\}$ by an exponential

sum,

$$\log \langle x|\psi_\theta \rangle = \log \sum_{i=1}^{\alpha L} \exp[F_i(x)]. \tag{3}$$

Here, θ denotes the vector of all variational parameters, which contains all entries of the weight matrices $W^{(l)}$ and bias vectors $b^{(l)}$. The variational parameters and therefore network outputs are complex numbers, with the activation function being applied separately to real and imaginary parts. The total number of network parameters scales as $N_{\text{par}} = \mathcal{O}(\mu\,\alpha^2 L^2)$. We study the SYK model at half filling, which fixes the fermion number to $L/2$. Therefore, the input to the neural network (2) is a vector of occupation numbers $x \in \{0,1\}^L$ such that $\sum_i x_i = L/2$.

We have verified our results for several variations of this network architecture. In particular, we have evaluated using tanh as nonlinear activation function as well as the addition of skip connections, which can be used to counteract the increased training complexity of networks beyond a certain depth [37, 38]. These variations did not achieve better results compared to those presented in the main text. Details can be found in Section III of the supplemental material (SM) [39].

Optimization.— The ground state of the network is obtained by numerically minimizing the overlap difference

$$\delta O(\theta, J) = 1 - \left| \frac{\langle \psi_\theta | \psi_{\text{GS}}(J) \rangle}{\langle \psi_\theta | \psi_\theta \rangle} \right| \tag{4}$$

between the variational state $|\psi_\theta\rangle$ and the ground state $|\psi_{\text{GS}}(J)\rangle$ with respect to the variational parameters θ using Adam [40]. We work with system sizes up to $L = 18$ sites, which are accessible via exact diagonalization (ED) and thus enable training using a supervised learning (SL) protocol targeting the overlap with the ED ground state $|\psi_{\text{GS}}(J)\rangle$ [41]. The system size allows us to evaluate the loss function (4) by summation over the full Hilbert space (preventing any potential errors arising from Monte Carlo sampling) and to assess the quality of our results using the relative energy error

$$\delta E(\theta; J) = \frac{E(\theta; J) - E_{\text{GS}}(J)}{E_{\text{GS}}(J)} \tag{5}$$

compared to the target ground state energy $E_{\text{GS}}(J) = \langle \psi_{\text{GS}}(J)|\hat{H}_{\text{SYK}}(J)|\psi_{\text{GS}}(J)\rangle$. Details on the optimization scheme are reported in Section II of the SM [39].

Results.— To start, we discuss the minimum energy error $\delta E_{\text{min}} = \min|_{t \in [0, t_{\text{max}}]} \delta E(\theta, J)$ reached within a maximum number of iterations t_{max} of the optimization protocol. Figure 2(b) shows the dependence of δE_{min} on the network width α for a network with a fixed number of $\mu = 2$ layers, while Fig. 3(b) shows the results as a function of network depth μ for deep networks with constant width $\alpha = 4$. We select $\delta E_{\text{threshold}} = 10^{-3}$ as a threshold error to assess successful convergence to the
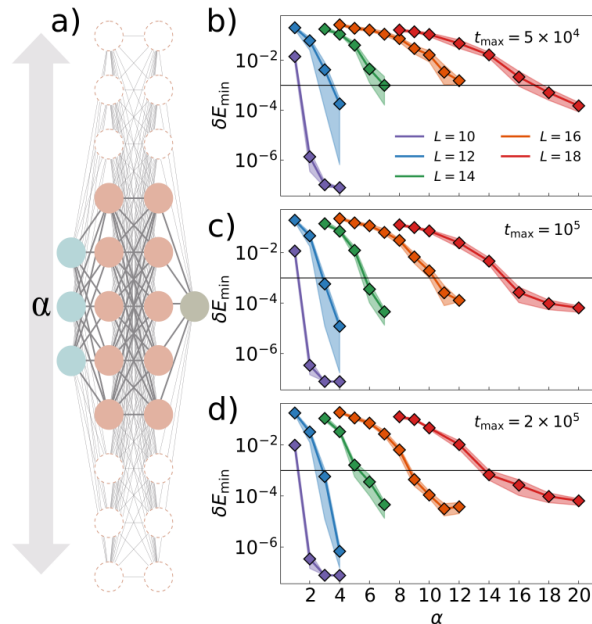
FIG. 2. (a) Shallow fully-connected feed-forward neural network, α denotes the hidden unit density of each layer and thus parametrizes the width of the network. (b), (c), (d) Relative ground state energy error δE as function of the network width α for several system sizes and random initializations after (b) 5×10^4, (c) 10^5, and (d) 2×10^5 simulation steps, respectively. The color of each set of data points corresponds to the average over four independent realizations of the network initial weights, for the system size L as indicated in the legend. The coloured areas give the maximum and minimum values of δE for the four independent runs. Black bars indicate $\delta E_{\text{threshold}} = 10^{-3}$.
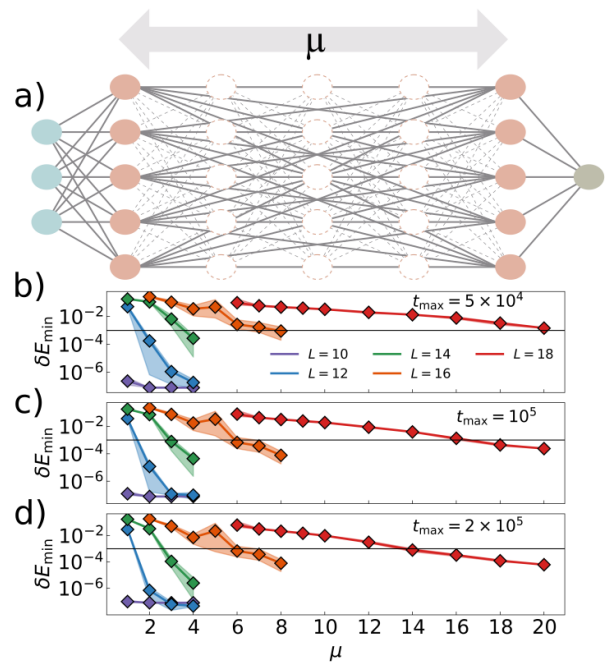

FIG. 3. (a) Deep fully-connected feed forward neural network, μ denotes the number of layers and thus the network depth. (b), (c), (d) Relative energy error δE as function of the network number of layers μ for several system sizes and random initializations after (b) 5×10^4, (c) 10^5, and (d) 2×10^5 simulation steps, respectively. The color of each set of data points corresponds to the average over four independent realizations of the network's initial weights, for the system size L as indicated in the legend. The coloured areas give the maximum and minimum values of δE for the four independent runs. Black bars indicate $\delta E_{\text{threshold}} = 10^{-3}$.

desired ground state. With this threshold, one can see in Figs. 2(b) and 3(b) that at any fixed number of training iterations t_{max} there is a systematic improvement of the accuracy with respect to increasing both α and μ, as one would expect given the increased representation capabilities of the network at larger sizes.

Next, we determine the minimum number of variational parameters at which the network is able to learn the ground state with the desired energy of $\delta E_{\text{threshold}}$. Especially for the smallest system sizes, there is a clear transition between regimes where the network is able or unable to learn the state (in particular as a function of α in the shallow network). For larger system sizes, it is somewhat more difficult to assess convergence. While both very small and very large networks converge to energies above or below the desired threshold within a reasonable optimization time, there is an intermediate regime where the energy gets close to the threshold but only converges at very long time scales. In order to systematically identify a value of α or μ at that boundary, we have developed a criterion used to truncate optimization runs after a reasonable optimization time when those

runs are predicted to ultimately converge to a $\delta E(\theta, J)$ higher than $\delta E_{\text{threshold}}$. See Section II B of the SM [39] for details. In Fig. 4 we show the number of network parameters at the critical α_{min} or μ_{min} at which the network is able to reach the target energy accuracy threshold. This allows for a comparison of network expressiveness for both varying width and depth on equal footing. We find that for both the shallow and deep network, an exponentially growing number of parameters is needed to achieve the target energy error. A comparison with the Hilbert space dimension reveals that the network only reaches this threshold once the number of variational parameters exceeds the number of probability amplitudes contained in the respective state vector. Hence, we find that our deep feed-forward NQS ansatz as trained here does not learn a more efficient representation of the SYK ground state than the full state vector representation. It is conceivable, in particular given the fully-connected nature of our ansatz, that there is some redundancy in the learned variational parameters, which could be used to achieve a degree of compression after training. In order to investigate this possibility, we have performed a low-rank approximation based on singular value decomposition of the weight matrices [42], the details of which
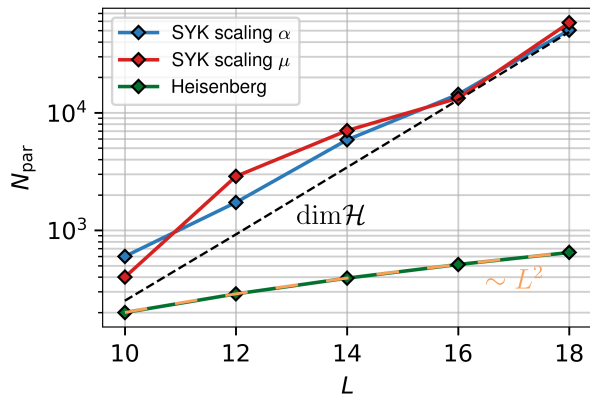
FIG. 4. Minimum number of parameters N_{par} required for the FFNN to learn the ground state of the SYK model as function of the system size L. Results are shown for the scaling with network width in a shallow ($\mu = 2$) network (blue lines) and for the scaling with network depth for fixed $\alpha = 4$ (red line). In both cases, an exponential scaling in the system size is observed, which matches the scaling of the full Hilbert space dimension $\dim \mathcal{H}$ (dashed line). The N_{par} scaling for the ground state of the Heisenberg model (blue) and the associated quadratic polynomial law are reported for comparison.

are reported in Section V of the SM [39]. This analysis, however, has not revealed such an redundancy.

Our scaling results cannot be interpreted as an immediate consequence of the entanglement scaling of the SYK model, as NQS are known to be able to efficiently represent some volume-law quantum states [20], while they seem to fail for others (as shown here). While a particular realization of the SYK Hamiltonian is of significantly higher complexity than a low-dimensional local lattice Hamiltonian (both because of its fully connected structure and the $\propto L^4$ randomly drawn interaction matrix elements), its ground state still exhibits more structure than a random Hilbert space vector. Since it is well known that deep (and, in fact, already two-layer) networks are able to memorize even completely random data once the number of network parameters exceeds the number of data points [43], these results provide evidence that our FFNN ansatz does not learn to utilize any of this structure but only manages to learn it as unstructured random data. This is in stark contrast to more structured lattice Hamiltonians, where it is clear from previous works that neural quantum states can approximate ground state energies with sub-exponential scaling and thus do manage to make use of structure present in the quantum ground state [44, 45] (although exponential scaling results as a function of real time have been previously found for time-evolved states in a one-dimensional lattice spin model [46]). We have found comparable behavior when evaluating our training procedure on the ground state of the Heisenberg spin model $\hat{H}_{\text{Heisb}} = \sum_{i=1}^{N} \sum_{q=1}^{3} \hat{\sigma}_i^{(q)} \hat{\sigma}_{i+1}^{(q)}$ on a one-dimensional chain with periodic boundary conditions diagonalized in the same zero-

magnetization subspace used for the SYK computations. The scaling of the required number of parameters to reach $\delta E_{\text{threshold}}$ in this model is also reported in Fig. 4. In this case, a relatively small and fixed $\alpha = 1$ and $\mu = 2$ independent of the system size are sufficient to reach this threshold, implying a polynomial scaling of the required number of parameters $N_{\text{par}} = \mathcal{O}(L^2)$. This corresponds to an effective compression of the information contained in the exact state vector and allows to study sizes beyond those tractable by full state simulation [44, 47]. However, the same approach fails to be useful in the more complex SYK model case.

Discussion.— We have tackled the prototypical SYK model using an NQS variational ansatz, presenting a systematic study of the ability of deep FFNNs to learn the volume-law entangled ground states of this model. Focusing on the scaling of the required number of parameters to describe the ground state to a desired and fixed accuracy we find that the size of the FFNN ansatz needs to grow exponentially in the system size. With this we show explicitly that the neural network ansatz is unable to efficiently represent SYK ground states in larger systems in spite of general results raising such hopes. We have performed this analysis using a variety of training techniques (as detailed in the SM [39]), showing that the observed scaling is robust to such implementation choices. While the proven capability of NQS to represent volume-law quantum states [20] clearly shows that this method has the potential to tackle problems out of the reach of established tensor-network based methods, our results demonstrate that the entanglement entropy is not the property that determines whether or not a physical quantum state can be efficiently represented by an NQS. It remains an intriguing open question which other properties of a physical quantum state determine the efficient applicability of NQS-based methods. Exploring the intermediate region between states that can be learned with compression (such as those that naturally appear in the Heisenberg and similar spin models) and states that cannot (such as the SYK results presented here) may help shed light on this question and may provide an improved understanding of the complexity of physical quantum states.

[1] F. Verstraete and J. I. Cirac, Matrix product states represent ground states faithfully, Physical Review B **73**, 10.1103/physrevb.73.094423 (2006).

[2] F. Verstraete, V. Murg, and J. Cirac, Matrix product states, projected entangled pair states, and variational renormalization group methods for quantum spin systems, Advances in Physics **57**, 143 (2008).

[3] J. Eisert, M. Cramer, and M. B. Plenio, Colloquium: Area laws for the entanglement entropy, Reviews of Modern Physics **82**, 277 (2010).

[4] U. Schollwöck, The density-matrix renormalization group in the age of matrix product states, Annals of Physics **326**, 96 (2011).

[5] J. I. Cirac, D. Pérez-García, N. Schuch, and F. Verstraete, Matrix product states and projected entangled pair states: Concepts, symmetries, theorems, Reviews of Modern Physics **93**, 10.1103/revmodphys.93.045003 (2021).

[6] E. Bianchi, L. Hackl, M. Kieburg, M. Rigol, and L. Vidmar, Volume-law entanglement entropy of typical pure quantum states, PRX Quantum **3**, 030201 (2022).

[7] G. Carleo, Y. Nomura, and M. Imada, Constructing exact representations of quantum many-body systems with deep neural networks, Nature Communications **9**, 10.1038/s41467-018-07520-3 (2018).

[8] J. Schmidhuber, Deep learning in neural networks: An overview, Neural Networks **61**, 85 (2015).

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

[10] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning* (MIT Press, 2016) http://www.deeplearningbook.org.

[11] I. Glasser, N. Pancotti, M. August, I. D. Rodriguez, and J. I. Cirac, Neural-network quantum states, string-bond states, and chiral topological states, Physical Review X **8**, 10.1103/physrevx.8.011006 (2018).

[12] S. R. Clark, Unifying neural-network quantum states and correlator product states via tensor networks, Journal of Physics A: Mathematical and Theoretical **51**, 135301 (2018).

[13] R. Kaubruegger, L. Pastori, and J. C. Budich, Chiral topological phases from artificial neural networks, Physical Review B **97**, 10.1103/physrevb.97.195136 (2018).

[14] K. Choo, T. Neupert, and G. Carleo, Two-dimensional frustrated j1-j2 model studied with neural network quantum states, Physical Review B **100**, 10.1103/physrevb.100.125124 (2019).

[15] G. Fabiani and J. H. Mentink, Investigating ultrafast quantum magnetism with machine learning, SciPost Phys. **7**, 004 (2019).

[16] M. Schmitt and M. Heyl, Quantum many-body dynamics in two dimensions with artificial neural networks, Physical Review Letters **125**, 10.1103/physrevlett.125.100503 (2020).

[17] G. Fabiani, M. Bouman, and J. Mentink, Supermagnonic propagation in two-dimensional antiferromagnets, Physical Review Letters **127**, 10.1103/physrevlett.127.097202 (2021).

[18] N. Astrakhantsev, T. Westerhout, A. Tiwari, K. Choo, A. Chen, M. H. Fischer, G. Carleo, and T. Neupert, Broken-symmetry ground states of the heisenberg model on the pyrochlore lattice, Physical Review X **11**, 10.1103/physrevx.11.041021 (2021).

[19] C. Roth, A. Szabó, and A. MacDonald, High-accuracy variational monte carlo for frustrated magnets with deep neural networks (2022).

[20] D.-L. Deng, X. Li, and S. Das Sarma, Quantum entanglement in neural network states, Phys. Rev. X **7**, 021021 (2017).

[21] G. Cybenko, Approximation by superpositions of a sigmoidal function, Mathematics of Control, Signals, and Systems **2**, 303 (1989).

[22] K. Hornik, Approximation capabilities of multilayer feedforward networks, Neural Networks **4**, 251 (1991).

[23] A. Pinkus, Approximation theory of the MLP model in neural networks, Acta Numerica **8**, 143 (1999).

[24] P. Kidger and T. Lyons, Universal Approximation with Deep Narrow Networks, in *Proceedings of Thirty Third Conference on Learning Theory*, Proceedings of Machine Learning Research, Vol. 125, edited by J. Abernethy and S. Agarwal (PMLR, 2020) pp. 2306–2327.

[25] S. Sachdev and J. Ye, Gapless spin-fluid ground state in a random quantum heisenberg magnet, Phys. Rev. Lett. **70**, 3339 (1993).

[26] A. Kitaev, A simple model of quantum holography (2015).

[27] J. Maldacena and D. Stanford, Remarks on the Sachdev-Ye-Kitaev model, Phys. Rev. D **94**, 106002 (2016).

[28] D. Chowdhury, A. Georges, O. Parcollet, and S. Sachdev, Sachdev-Ye-Kitaev models and beyond: Window into non-Fermi liquids, Reviews of Modern Physics **94**, 035004 (2022).

[29] C. Liu, X. Chen, and L. Balents, Quantum entanglement of the sachdev-ye-kitaev models, Phys. Rev. B **97**, 245126 (2018).

[30] G. Akemann, J. Baik, and P. D. Francesco, eds., *The Oxford Handbook of Random Matrix Theory* (Oxford University Press, 2015).

[31] W. Fu and S. Sachdev, Numerical study of fermion and boson models with infinite-range random interactions, Phys. Rev. B **94**, 035135 (2016).

[32] P. Zhang, Quantum Entanglement in the Sachdev-Ye-Kitaev Model and its Generalizations, (2022).

[33] A. Haldar, O. Tavakol, and T. Scaffidi, Variational wave functions for sachdev-ye-kitaev models, Phys. Rev. Research **3**, 023020 (2021).

[34] A. Altland and D. Bagrets, Quantum ergodicity in the SYK model, Nuclear Physics B **930**, 45 (2018).

[35] M. Brzezinska, Y. Guan, O. V. Yazyev, S. Sachdev, and A. Kruchkov, Engineering SYK interactions in disordered graphene flakes under realistic experimental conditions, (2022), arXiv:2208.01032.

[36] G. Klambauer, T. Unterthiner, A. Mayr, and S. Hochreiter, Self-normalizing neural networks (2017), arXiv:1706.02515.

[37] K. He, X. Zhang, S. Ren, and J. Sun, Deep residual learning for image recognition, in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2016) pp. 770–778.

[38] H. Li, Z. Xu, G. Taylor, C. Studer, and T. Goldstein, Visualizing the loss landscape of neural nets, in *Advances in Neural Information Processing Systems 31: Annual Con-*

ference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada, edited by S. Bengio, H. M. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (2018) pp. 6391–6401.

[39] See Supplementary Material at *[link to be inserted]* for details on the network architecture, training protocols, and comparisons with different optimisation schemes.

[40] D. P. Kingma and J. Ba, Adam: A method for stochastic optimization (2014), arXiv:1412.6980.

[41] B. Jónsson, B. Bauer, and G. Carleo, Neural-network states for the classical simulation of quantum computing (2018).

[42] J. Xue, J. Li, and Y. Gong, Restructuring of deep neural network acoustic models with singular value decomposition, in *Interspeech 2013* (ISCA, 2013).

[43] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, Understanding deep learning (still) requires rethinking generalization, Commun. ACM **64**, 107 (2021).

[44] D. Sehayek, A. Golubeva, M. S. Albergo, B. Kulchytskyy, G. Torlai, and R. G. Melko, Learnability scaling of quantum states: Restricted boltzmann machines, Physical Review B **100**, 10.1103/physrevb.100.195125 (2019).

[45] L. L. Viteritti, F. Ferrari, and F. Becca, Accuracy of restricted boltzmann machines for the one-dimensional \$j_1-j_2\$ heisenberg model, SciPost Physics **12**, 10.21468/scipostphys.12.5.166 (2022).

[46] S.-H. Lin and F. Pollmann, Scaling of neural-network quantum states for time evolution, physica status solidi (b) **259**, 2100172 (2022).

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

[48] F. Vicentini, D. Hofmann, A. Szabó, D. Wu, C. Roth, C. Giuliani, G. Pescia, J. Nys, V. Vargas-Calderón, N. Astrakhantsev, and G. Carleo, NetKet 3: Machine Learning Toolbox for Many-Body Quantum Systems, SciPost Phys. Codebases , 7 (2022).

[49] G. Carleo, K. Choo, D. Hofmann, J. E. Smith, T. Westerhout, F. Alet, E. J. Davis, S. Efthymiou, I. Glasser, S.-H. Lin, M. Mauri, G. Mazzola, C. B. Mendl, E. van Nieuwenburg, O. O'Reilly, H. Théveniaut, G. Torlai, F. Vicentini, and A. Wietek, NetKet: A machine learning toolkit for many-body quantum systems, SoftwareX **10**, 100311 (2019).

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

[51] P. Thörnig, JURECA: Data centric and booster modules implementing the modular supercomputing architecture at jülich supercomputing centre, Journal of large-scale research facilities JLSRF **7**, 10.17815/jlsrf-7-182 (2021).

Supplementary Material for:
Can neural quantum states learn volume-law ground states?

Giacomo Passetti,[1] Damian Hofmann,[2] Pit Neitemeier,[1]
Lukas Grunwald,[1,2] Michael A. Sentef,[2] and Dante M. Kennes[1,2]

[1]*Institut für Theorie der Statistischen Physik, RWTH Aachen University and*
JARA-Fundamentals of Future Information Technology, 52056 Aachen, Germany
[2]*Max Planck Institute for the Structure and Dynamics of Matter, Center for*
Free-Electron Laser Science (CFEL), Luruper Chaussee 149, 22761 Hamburg, Germany

I. NETWORK ARCHITECTURE

A fully-connected feed-forward network (FFNN) is a composition of $\mu \in \mathbb{N}_{>0}$ layers where the lth network layer, $1 \le l \le \mu$, applies the transformation (writing $[n] := \{1, \ldots, n\}$ for any $n \in \mathbb{N}_{>0}$)

$$z_i^{(l)} = \sum_j W_{ij}^{(l)} y_j^{(l)} + b_i^{(l)}, \qquad y_i^{(l+1)} = \phi(z_i^{(l)})$$

$$\begin{cases} i \in [L], \, j \in [\alpha L], & l = 1, \\ i, j \in [\alpha L], & l > 1, \end{cases} \tag{S1}$$

from input $y^{(l)}$ to output signal $y^{(l+1)}$. This is an affine transformation with weight matrix (or kernel) $W^{(l)}$, bias $b^{(l)}$, and a nonlinear activation function ϕ. The full set of network parameters, consisting of all weights and biases, is denoted by $\theta \in \mathbb{C}^{N_{\mathrm{par}}}$. For the results presented in the main text, we have used the SELU activation function

$$\mathrm{SELU}(x) = \lambda \begin{cases} x, & x > 0, \\ (a-1)e^x, & x \le 0 \end{cases} \tag{S2}$$

(where $\lambda \approx 1.0507$ and $a \approx 1.6733$), as implemented in JAX [1], which helps circumvent the vanishing gradient problem of training deep networks [2]. Since in our case the network acts on complex-valued signals, we apply SELU separately to real and imaginary part of the input (`reim_selu` in NETKET [3]), giving the complex activation

$$\phi(z) = \mathrm{SELU}(\mathrm{Re}\, z) + i\, \mathrm{SELU}(\mathrm{Im}\, z). \tag{S3}$$

Using the hyperbolic tangent as activation provides equivalent or (for deep networks) worse results, as we discuss below (Section III D). The log-probability amplitudes of the variational state are then obtained by performing the logsumexp operation on the output of the final layer, i.e.,

$$\log \langle x | \psi_\theta \rangle = \log \left(\sum_{i=1}^{\alpha L} \exp(y_i^{(\mu+1)}) \right), \qquad y_i^{(1)} = x_i, \tag{S4}$$

with $\{|x\rangle \mid x \in \{0,1\}^L\}$ denoting the computational basis.

II. TRAINING PROTOCOL

A. Supervised learning

In this work we considered only system sizes up to $L = 18$, which are solvable via exact diagonalization (ED). We solve the ED problem for a given realization of the model $\hat{H}_{\mathrm{SYK}}(J)$, finding the corresponding ground state $|\psi_{\mathrm{GS}}(J)\rangle$. The key result of this work, the exponential scaling of the required network size to reach a specific error bound, is independent from the specific J realization for the couplings of the SYK model. This has been checked by performing the full analysis as described in the following paragraphs for two independent realizations of the random coupling matrix J.
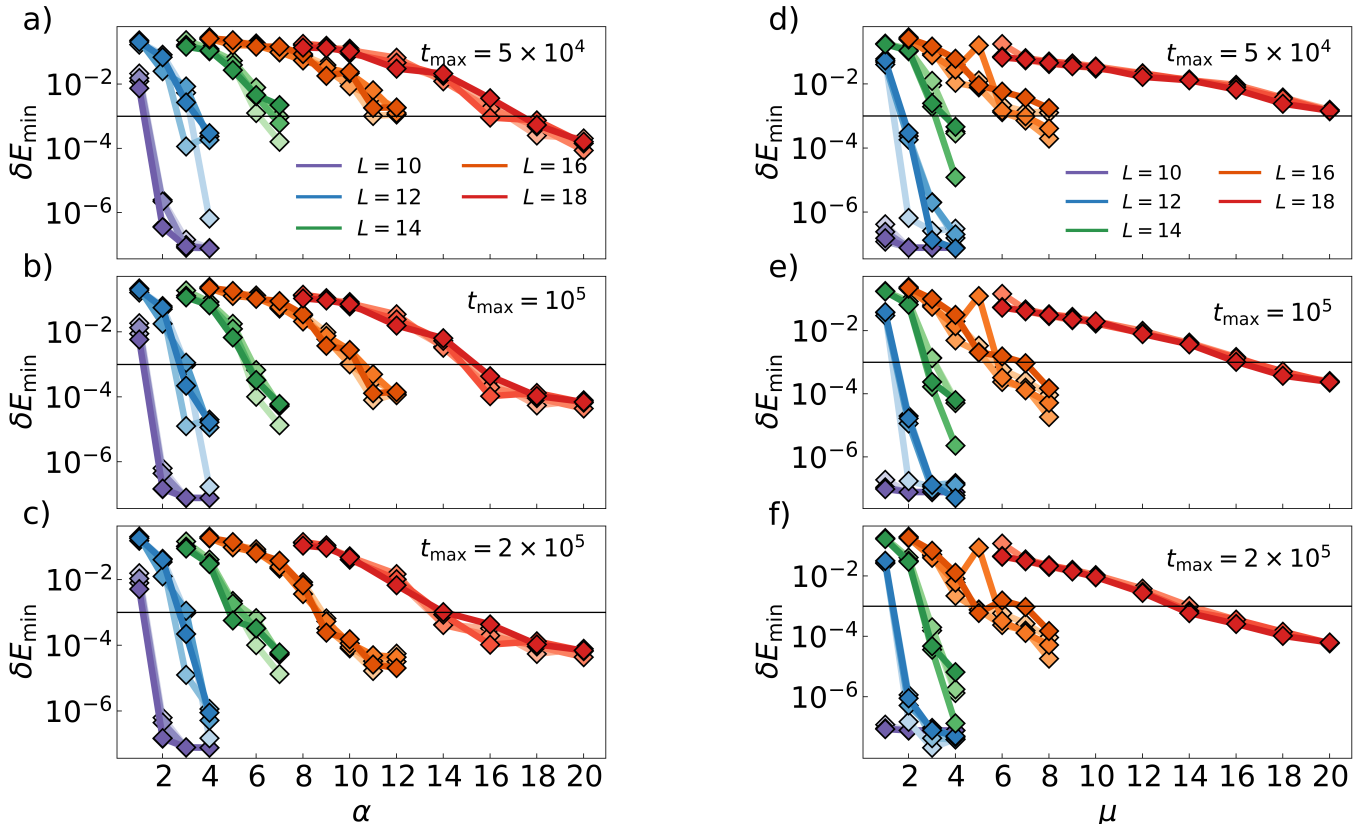
FIG. S1. Relative energy error δE as function of [(a), (b), (c)] the network width α and of [(d), (e), (f)] the number of layer μ for several system sizes and random initializations after [(a), (d)] 5×10^4, [(b), (e)] 10^5, and [(c), (f)] 2×10^5 simulation steps, respectively. Here we report the full dataset from which the average value plotted in the main text has been derived. The color of each set of data points corresponds to the system size L as indicated in the legend while different shades of the same color belong to independent realizations of the initial network weights.

Given the ED solution $|\psi_{\mathrm{GS}}\rangle$, it is possible to train the network to reproduce the correct probability amplitudes (up to norm and phase gauge freedoms) via supervised learning [4]. In practice, we have done so by optimizing the loss function

$$\delta O(\theta, J) = 1 - \left| \frac{\langle \psi_\theta | \psi_{\mathrm{GS}}(J) \rangle}{\langle \psi_\theta | \psi_\theta \rangle} \right| \tag{S5}$$

using the Adam optimizer [5]. Another training scheme can be implemented by directly minimizing the expected energy $E(\theta, J) = \langle \hat{H}_{\mathrm{SYK}}(J) \rangle_{|\psi_\theta\rangle}$ of the system as the loss function. For this loss function, it is possible to extend the training to system sizes beyond the reach of ED studies through variational Monte Carlo (VMC) sampling [6], which is not possible in the SL scheme employed here due to its reliance on the knowledge of the full solution vector. Since all our system sizes could be treated via full summation, we have not relied on VMC sampling in this work.

As our goal is to find the network size required to represent the SYK ground states at a given system size, we have compared training runs for both SL and energy minimization routines, finding the SL to be the best performing choice in the regime under consideration (compare Section III B). Therefore, we have selected the SL results for presentation in the main text. Our observations regarding the exponential scaling of required network size hold also for the variational energy optimization runs.

The optimization problem for a neural network is challenging on its own and there is unfortunately no general way to systematically identify the best choice of hyperparameters. Here, we have chosen to study the scaling with regard to width α and depth μ of the network by performing two sets of runs: Varying the width for a two-layer network and varying the depth of a network of fixed width $\alpha = 4$. Considering the trajectory of weights $\{\theta(t)\}_{t \in [t_{\max}]}$ (where $[t_{\max}] := \{1, \ldots, t_{\max}\} \subseteq \mathbb{N}$) over the sequence of optimization steps, we can the define the optimal energy relative error as $\delta E_{\min}(\theta, J) = \min_{t \in [t_{\max}]} E(\theta(t), J)$. In Fig. S1 we report $\delta E_{\min}$ for SL protocols for three different number of total iterations $t_{\max}$ as function of α. All the data reported in this plot have been obtained with $\mu = 2$, keeping the learning rate of the Adam optimizer constant throughout all the simulation. From this analysis we can make
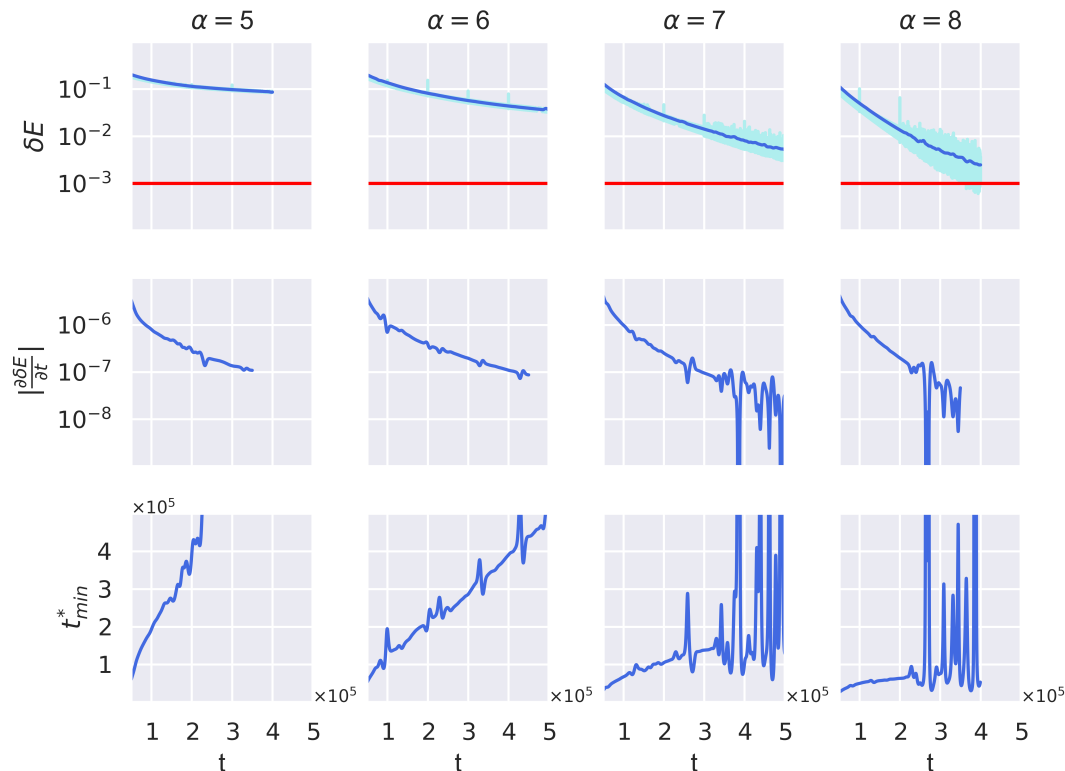
FIG. S2. (a) Relative energy error δE as function of the iteration step for shallow networks with $\mu = 2$ layers and width α as reported in the respective panels. The system size is $L = 16$. In light blue we report the raw data, in dark blue the result of a window smoothing filter applied to the raw data. (b) Absolute value of the numerical derivative $|\frac{\partial(\delta E)}{\partial t}|$. (c) Estimated minimum number of steps required to convergence $t^*_{\min}$ as defined by Eq. (S6).

two relevant observations. The first is that up to the network sizes simulated there is a monotonic improvement of the error with increasing width α. The second observation is that at each fixed number of simulation steps there is an exponential scaling of the $\alpha_{\min}$ required to bring $\delta E_{\min}$ below an arbitrary threshold (10^{-3} in our example) as a function of the system size. With the data reported in this paragraph, it cannot be ruled out that this exponential cost in network size is due to an exponential scaling of the number of steps required to bring the small α values to convergence. However, through an analysis of the training curves it is possible to make a strong argument to identify the $\alpha_{\min}$ discussed in the main text as a quantity independent from the number of training steps. This analysis is explained in the following section.

B. Truncation scheme

In the first row of Fig. S2, we report $\delta E(\theta(t), J)$ for four different alpha values distributed around what we are going to define as the critical $\alpha_{\min}$. The light blue line corresponds to the raw data obtained from the training process. In order to remove the fluctuations around the moving average, we apply a flat window filter over the raw error, obtaining the profile corresponding to the dark blue line. This running average error is smooth and monotonically decreasing. Furthermore, the slope of these curves is also decreasing (in absolute value), implying a slowing rate of convergence of the optimization. In order to conserve computational resources, we truncate the runs that are expected to converge to a $\delta E(\theta, J)$ above the energy threshold of 10^{-3} based on the following criterion: Assuming that the absolute value of the slope is monotonically decreasing throughout all the training iterations, it is possible to estimate, at any step
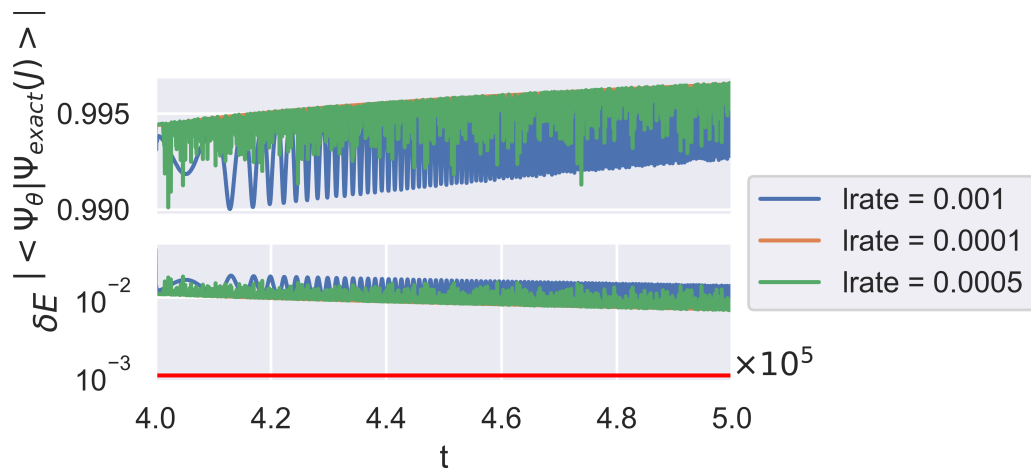
FIG. S3. (a) Overlap $\langle \Psi_\theta | \Psi_{\text{exact}}(J) \rangle$ and (b) relative energy error δE for a training protocol in the interval $t \in [4 \times 10^5, 5 \times 10^5]$ steps. The same pre-trained network here it has been updated with the three different learning ratios reported in the legend.

t, a lower bound for the number of steps required to reach the error threshold as

$$t_{\min}^*(t) = \frac{\delta E(\theta(t), J) - \delta E^{\text{threshold}}}{|\partial_t \delta E(\theta(t), J)|}. \tag{S6}$$

Within the assumption of constant slope, the equality $\delta E(\theta(t + t_{\min}^*(t)), J) = \delta E^{\text{threshold}}$ holds. If now we define a control interval Δt big enough to average out the noise around training curve, one can check if a candidate run is trending towards convergence below the error threshold based on the slope of $t_{\min}^*(t)$,

$$\delta t_{\min}^*(t) = \frac{t_{\min}^*(t + \Delta t) - t_{\min}^*(t)}{\Delta t}. \tag{S7}$$

The runs that are expected to reach convergence are those that satisfy

$$\delta t_{\min}^*(t + \Delta t) < \delta t_{\min}^*(t). \tag{S8}$$

We simulated all runs for at least $t_{\max} = 2 \times 10^5$, for those runs that where still above the threshold for that step we used the criterion (S8) (with $\Delta t = 10^5$) in order to assess whether they can still be expected to reach the error threshold, based on their current rate of convergence. The runs for which converge below the threshold was ruled out by our criterion were stopped in order to conserve computational resources, while we did perform additional blocks of 10^5 steps for the remaining simulations, until the truncation criterion (S8) did apply or the error threshold was reached. The resulting $\alpha_{\min}$ values reported in Fig. 4 of the main text are the smallest simulated value of α for which convergence below the threshold was reached. The μ scaling analysis was performed in an equivalent way.

III. OVERVIEW OF ALTERNATIVE TRAINING METHODS INVESTIGATED

In the previous paragraphs we have described the protocol and data analysis necessary to reproduce the results that we discussed in the main text. Here we are showing alternatives to the training scheme and hyperparameters. The alternatives which we have tested achieved either worse accuracy or no significant improvement compared to the results we selected for the main text.

A. Different learning rate

We have used the Adam optimizer using a constant learning rate throughout the simulations reported in the main text. One advantage of Adam with respect to SGD is that it automatically adapts the step size based on the history of loss gradients. Still, Adam has to be initialized with a specific learning rate and this can potentially influence the effectiveness of the training. While determining the optimal learning rate for our simulations, we have also tried
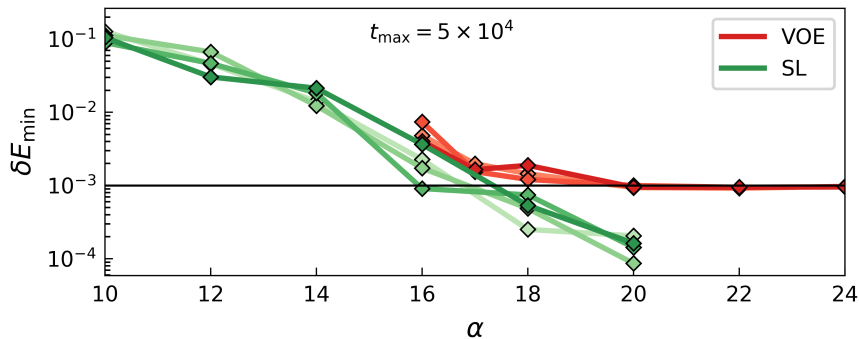
FIG. S4. Minimum relative energy error $\delta E_{\min}$ obtained using the SL (green) and a VOE (red) loss functions using $t_{\max} = 5 \times 10^4$ steps as function of the number of network width α. For the data reported in this plot, we considered system size $L = 18$ and a fixed $\mu = 2$
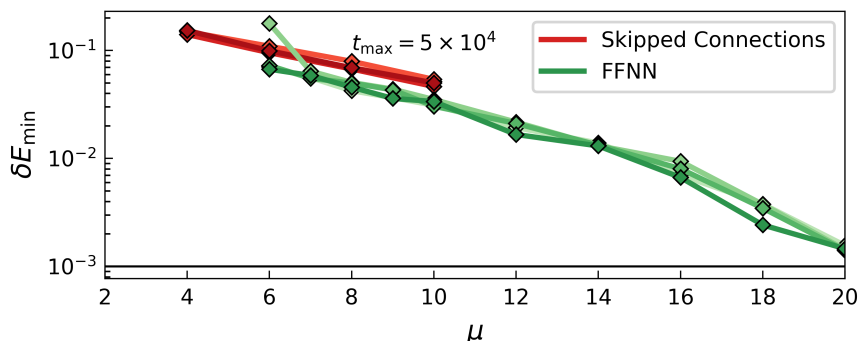

FIG. S5. Minimum relative energy error $\delta E_{\min}$ obtained during a SL training protocol of a deep FFNN (green) and of a Network with Skipped connections (red) with a number of steps set to $t_{\max} = 5 \times 10^4$ as function of the number of network laters μ. For the data reported in this plot we considered system size $L = 18$ and a fixed $\alpha = 4$

schemes involving an update of the learning rate during the simulation. We report an example of an unconverged training curve in Fig. S3 for which, after $4 \cdot 10^5$ steps at a fixed learning rate, we performed further 10^5 steps with three different adjusted learning rates. These changes of learning rate did not result in an improvement of training performance.

B. Variational energy optimization

Beside the SL protocol described above, we have also considered directly optimizing the variational energy

$$E(\theta, J) = \frac{\langle \psi_\theta | \hat{H}_{\mathrm{SYK}}(J) | \psi_\theta \rangle}{\langle \psi_\theta | \psi_\theta \rangle}. \tag{S9}$$

For disambiguation, we refer to this loss function as the variationally optimized energy (VOE). Note that this loss does not involve the ED solution of the model and can be applied to system sizes beyond those limits, if it is used together with MC sampling of the quantum states [6]. However, as we are interested in comparing the performance of the SL and VOE protocols in the ED regime, we only use the loss (S9) obtained by full summation in the following. Within the parameter range tested, the SL protocol achieved systematically lower errors than the VOE loss. As an example of this, we show in Fig. S4 the resulting $\delta E_{\min}$ for a given system size ($L = 18$) and a fixed number of iteration steps. Although the trend is similar between the two curves, the SL reaches the convergence threshold at lower α values.
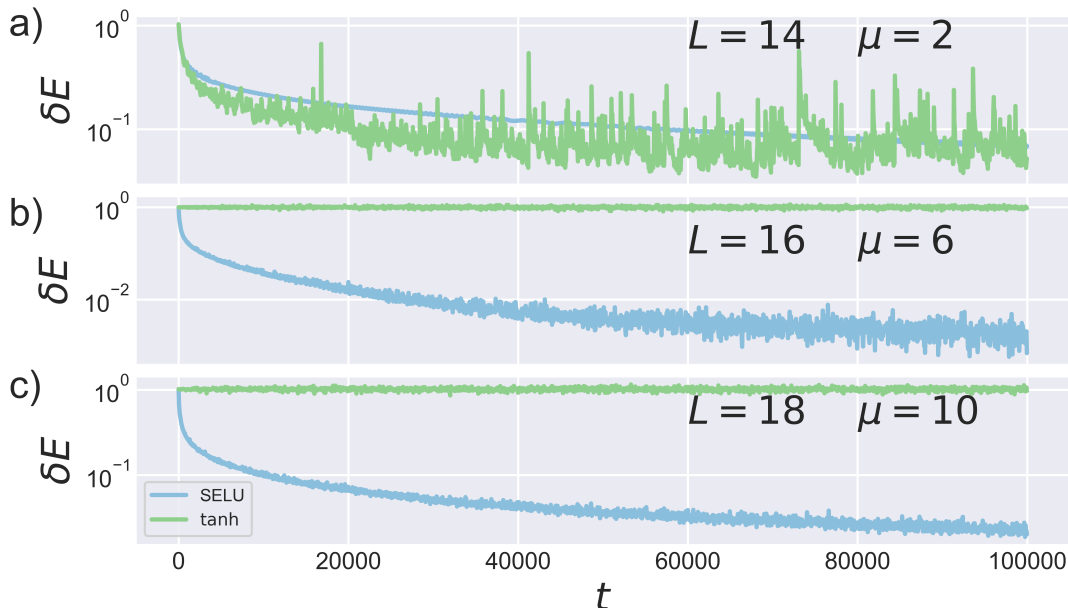
FIG. S6. Relative energy error δE as function of the iteration step t using the SL training protocol with SELU (light blue) and tanh (light green) activation functions. The figure shows the data for (a) $L = 14$, $\mu = 2$; (b) $L = 16$, $\mu = 6$; and (c) $L = 18$, $\mu = 10$.

C. Deep networks with skip connections

In the main part of this work, we presented an extensive discussion of the results obtained when training a FFNN with the architecture defined in Section I. Among the two different hyperparameter dependencies discussed there, the number of layers μ scaling requires some extra attention. It is known that while adding more layers enhances the networks expressive capability, at the same time it can make the training more difficult [7]. One common remedy to this is the addition of skip connections between the layers of the neural network [8]. We report in Fig. S5 a comparison between the training at a fixed number of iteration steps $t_{\max} = 5 \cdot 10^4$ between the FFNN discussed in the main text and a fully-connected network with skip connections. Specifically, our network is divided into n_B blocks, each containing ℓ layers. The total number of layers is thus $\mu = n_B \ell$. After the application of each block (i.e., after applying ℓ fully-connected layers), the content of the original input layer (with the first affine transformation applied) is added to the output before it is passed to the following block. For the SL simulation runs considered (Fig. S5), this architecture does not give an improvement in energy error when compared to a simple FFNN. We have found improved convergence (compared to the simple FFNN) of the skip connection networks in some runs using the VOE loss. However, in those cases the number of layers already exceeded the critical μ (around which the network with skip connections still performed worse), so no improvement with regard to compression was obtained this way.

D. Hyperbolic tangent activation function

As already stated, for all the data discussed in the main part, the FFNN implemented where following the definition presented in Eq. (S1) using SELU as activation function ϕ. While in principle there are many viable choices with regard to which specific activation function utilizing, we opted to restrict our studies to a particular activation in order to investigate extensively the dependence of the representability power as function of the network hyperparameters, keeping a manageable number of simulations to perform. Thus the need to choose an activation function able to achieve reliable performance when dealing with both deep and shallow networks. We present here a comparison of the performances achieved by the same training protocol and network architecture while using another common activation function, the hyperbolic tangent. In Fig. S6, we present training profiles for the relative energy error δE as function of the iteration step t. We show this for both a network with SELU and a network with tanh activation function for several numbers of layers μ. As one could expect from the vanishing gradient problem that effects $tanh$, we observe that also for the network sizes relevant to our problem, above a certain number of layers the tanh activated networks are not able to improve the energy error anymore. At the same time SELU displayed a monotonic improvement of
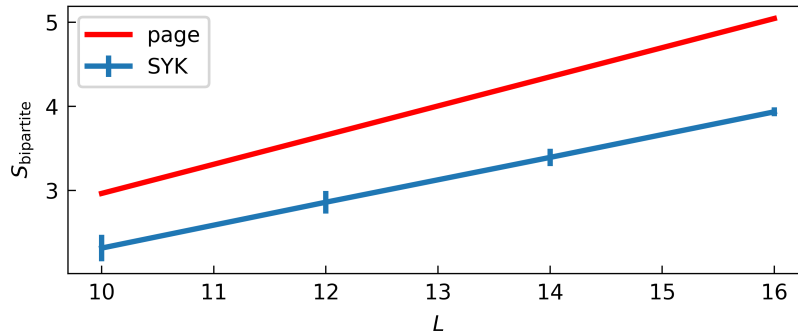
FIG. S7. Bipartite entanglement entropy averaged over 4 independent realizations of the SYK model (error bar shows the standard deviation) compared to bipartite entanglement of a random state given by the Page value (red).

the energy error with the number of layers (up to scales relevant for our specific problem) and thus is an effective choice (compare Fig. S1).

IV. SYK BIPARTITE ENTANGLEMENT

Following the standard definition, the bipartite entanglement of a density matrix ρ corresponds to

$$S_{\text{bipartite}} = -\text{Tr}\left[\rho_A \log\left(\rho_A\right)\right] = -\text{Tr}\left[\rho_B \log\left(\rho_B\right)\right] \tag{S10}$$

where $\rho_{A(B)} = \text{Tr}_{B(A)}\left[\rho\right]$ is the reduced density matrix obtained from the density matrix ρ and tracing over the degrees of the subpartition B(A) of the total hilbert space, in the special case of $\dim(A) = \dim(B)$. We compare the $S_{\text{bipartite}}$ scaling for four independent finite size SYK model realizations together with the Page value [9], that quantifies the entanglement for a pure random state. In the special case case of bipartite entanglement, the page value is

$$S_{\text{page}} = N \log(2) - \frac{1}{2}. \tag{S11}$$

As one can see in Fig. S7, the entanglement scaling of the SYK model follows a volume law scaling that does not saturate the page value, showing that the SYK states exhibit structure beyond a pure random state.

V. NETWORK COMPRESSION

Trained neural networks can be amenable to compression in order to reduce the total number of parameters. This is usually done for the purpose of reducing memory and computational requirements in order to run a network on a lower powered device. Here, we have explored a compression scheme based on a lower-rank approximation of the weight matrices in each layer [10] in order to gauge whether any potential redundancy in the learned variational parameters can be identified this way. This is done by performing a singular value decomposition (SVD)

$$W^{(l)} = U^{(l)} S^{(l)} V^{(l)} \tag{S12}$$

where $U^{(l)}$, $V^{(l)}$ are unitary matrices and $S^{(l)} = \text{diag}(\sigma_1^{(l)}, \ldots, \sigma_M^{(l)})$ is the matrix of singular values $\sigma_1^{(l)} \geq \ldots \geq \sigma_M^{(l)} \geq 0$. We truncate this spectrum by discarding all singular values below a threshold λ relative to the largest singular value, i.e., based on the criterion $\sigma_i^{(l)}/\sigma_1^{(l)} < \lambda$. For simplicity, we chose λ uniformly for each layer. As a measure of the approximation error, we report in Fig. S8 the relative energy error after truncation as a function of the fraction

$$q = \frac{\text{singular values retained}}{\text{total singular values}} \tag{S13}$$

of singular values retained over all weight matrices of the network for different system sizes and choices of hyper-parameters. This data shows that already a limited amount of truncation ($0.95 \leq q < 1.0$) results in a significant
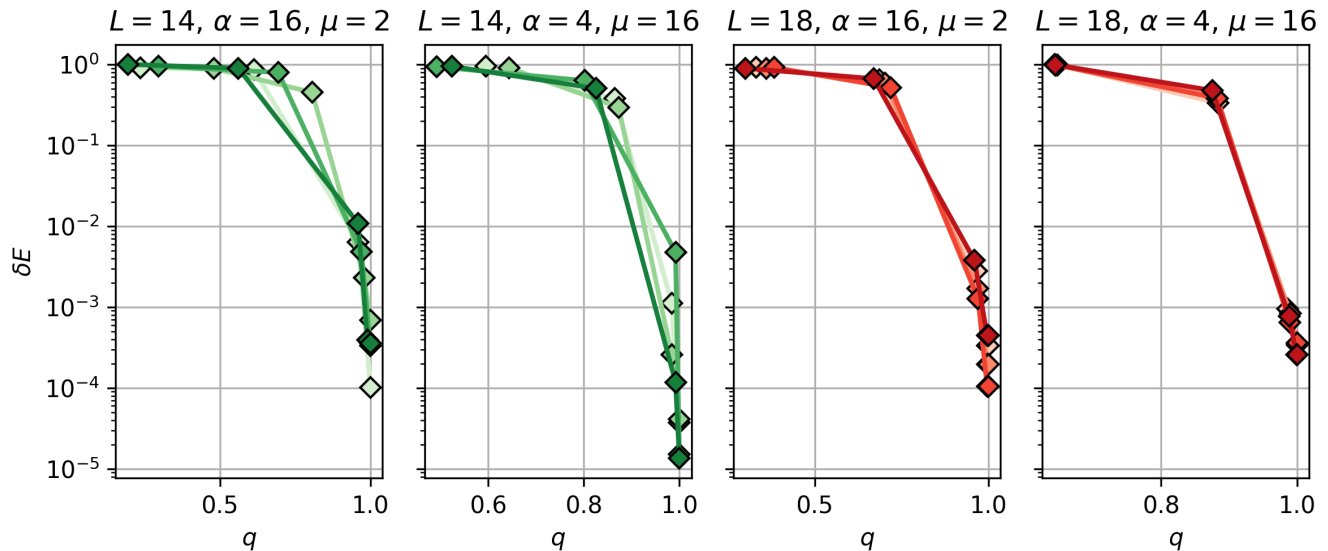
FIG. S8.   Relative energy error as a function of the fraction of retained singular values q for four different choices of hyperparameters indicated in the title of each panel. The color corresponds to different network initializations as in Fig. S1.

increase in energy error. We therefore conclude that at uniform SVD truncation of the weight matrices does not reveal redundancy in our learned SYK ground states.

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

[2] G. Klambauer, T. Unterthiner, A. Mayr, and S. Hochreiter, Self-normalizing neural networks (2017), arXiv:1706.02515.

[3] F. Vicentini, D. Hofmann, A. Szabó, D. Wu, C. Roth, C. Giuliani, G. Pescia, J. Nys, V. Vargas-Calderón, N. Astrakhantsev, and G. Carleo, NetKet 3: Machine Learning Toolbox for Many-Body Quantum Systems, SciPost Phys. Codebases , 7 (2022).

[4] B. Jónsson, B. Bauer, and G. Carleo, Neural-network states for the classical simulation of quantum computing (2018).

[5] D. P. Kingma and J. Ba, Adam: A method for stochastic optimization (2014), arXiv:1412.6980.

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

[7] K. He, X. Zhang, S. Ren, and J. Sun, Deep residual learning for image recognition, in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2016) pp. 770–778.

[8] H. Li, Z. Xu, G. Taylor, C. Studer, and T. Goldstein, Visualizing the loss landscape of neural nets, in *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, edited by S. Bengio, H. M. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (2018) pp. 6391–6401.

[9] D. N. Page, Average entropy of a subsystem, Phys. Rev. Lett. **71**, 1291 (1993).

[10] J. Xue, J. Li, and Y. Gong, Restructuring of deep neural network acoustic models with singular value decomposition, in *Interspeech 2013* (ISCA, 2013).