communications physics

Article

https://doi.org/10.1038/s42005-024-01732-4

A simple linear algebra identity to optimize large-scale neural network quantum states

Check for updates

Riccardo Rende [1,3] ✉, Luciano Loris Viteritti [2,3] ✉, Lorenzo Bardone [1], Federico Becca [2] & Sebastian Goldt [1] ✉

Neural-network architectures have been increasingly used to represent quantum many-body wave functions. These networks require a large number of variational parameters and are challenging to optimize using traditional methods, as gradient descent. Stochastic reconfiguration (SR) has been effective with a limited number of parameters, but becomes impractical beyond a few thousand parameters. Here, we leverage a simple linear algebra identity to show that SR can be employed even in the deep learning scenario. We demonstrate the effectiveness of our method by optimizing a Deep Transformer architecture with 3×10^5 parameters, achieving state-of-the-art ground-state energy in the J_1–J_2 Heisenberg model at $J_2/J_1 = 0.5$ on the 10×10 square lattice, a challenging benchmark in highly-frustrated magnetism. This work marks a significant step forward in the scalability and efficiency of SR for neural-network quantum states, making them a promising method to investigate unknown quantum phases of matter, where other methods struggle.

Deep learning has become crucial in many research fields, with neural networks being the key to achieve impressive results. Well-known examples include deep convolutional neural networks (CNNs) for image recognition[1,2] and Deep Transformers for language-related tasks[3–5]. The success of deep networks comes from two ingredients: architectures with a large number of parameters (often in the billions), which allow for great flexibility, and training these networks on large amounts of data. However, to successfully train these large models in practice, one needs to navigate the complicated and highly non-convex landscape associated with this extensive parameter space.

The most used methods rely on stochastic gradient descent (SGD), where the gradient of the loss function is estimated from a randomly selected subset of the training data. Over the years, variations of traditional SGD, such as Adam[6] or AdamW[7], have proven highly effective, leading to more accurate results. In the late 1990s, Amari et al.[8,9] suggested to use the knowledge of the geometric structure of the parameter space to adjust the gradient direction for non-convex landscapes, defining the concept of natural gradients. In the same years, Sorella[10,11] proposed a similar method, now known as stochastic reconfiguration (SR), to enhance the optimization of variational functions in quantum many-body systems. Importantly, this latter approach typically outperforms other methods such as SGD or Adam, leading to significantly lower variational energies. The main idea of SR is to exploit information about the curvature of the loss landscape, thus improving the convergence speed in landscapes which are steep in some directions and shallow in others[12]. For physically inspired wave functions (e.g., Jastrow-Slater[13] or Gutzwiller-projected states[14]) the original SR formulation is a highly efficient method since there are few variational parameters, typically from $O(10)$ to $O(10^2)$.

Over the past few years, neural networks have been extensively used as powerful variational *Ansätze* for studying interacting spin models[15], and the number of parameters has increased significantly. Starting with simple restricted Boltzmann machines (RBMs)[15–20], more complicated architectures as CNNs[21–23] and recurrent neural networks[24–27] have been introduced to handle challenging systems. In particular, Deep CNNs[28–31] have proven to be highly accurate for two-dimensional models, outdoing methods as density-matrix renormalization group (DMRG) approaches[32] and Gutzwiller-projected states[14]. These deep learning models have great performances when the number of parameters is large. However, a significant bottleneck arises when employing the original formulation of SR for optimization, as it is based on the inversion of a matrix of size $P \times P$, where P denotes the number of parameters. Consequently, this approach becomes computationally infeasible as the parameter count exceeds $O(10^4)$, primarily due to the constraints imposed by the limited memory capacity of current-generation GPUs.

Recently, Chen and Heyl[30] made a step forward in the optimization procedure by introducing an alternative method, dubbed MinSR, to train

[1]International School for Advanced Studies (SISSA), Trieste, Italy. [2]Dipartimento di Fisica, Università di Trieste, Trieste, Italy. [3]These authors contributed equally: Riccardo Rende, Luciano Loris Viteritti. ✉e-mail: rrende@sissa.it; lucianoloris.viteritti@phd.units.it; sgoldt@sissa.it

neural-network quantum states. MinSR does not require inverting the original $P \times P$ matrix but instead a much smaller $M \times M$ one, where M is the number of configurations used to estimate the SR matrix. This is convenient in the deep learning setup where $P \gg M$. Most importantly, this procedure avoids allocating the $P \times P$ matrix, reducing the memory cost. However, this formulation is obtained by minimizing the Fubini-Study distance with an ad hoc constraint. In this work, we first use a simple relation from linear algebra to show, in a transparent way, that SR can be rewritten exactly in a form which involves inverting a small $M \times M$ matrix (in the case of real-valued wave functions and a $2M \times 2M$ matrix for complex-valued ones) and that only a standard regularization of the SR matrix is required. Then, we exploit our technique to optimize a Deep Vision Transformer (Deep ViT) model, which has demonstrated exceptional accuracy in describing quantum spin systems in one and two spatial dimensions[33–36]. Using almost 3×10^5 variational parameters, we are able to achieve state-of-the-art ground-state energy in the most paradigmatic example of quantum many-body spin model, the J_1–J_2 Heisenberg model on square lattice:

$$\hat{H} = J_1 \sum_{\langle i,j \rangle} \hat{\boldsymbol{S}}_i \cdot \hat{\boldsymbol{S}}_j + J_2 \sum_{\langle\langle i,j \rangle\rangle} \hat{\boldsymbol{S}}_i \cdot \hat{\boldsymbol{S}}_j \quad (1)$$

where $\hat{\boldsymbol{S}}_i = (S_i^x, S_i^y, S_i^z)$ is the $S = 1/2$ spin operator at site i and J_1 and J_2 are nearest- and next-nearest-neighbour antiferromagnetic couplings, respectively. Its ground-state properties have been the subject of many studies over the years, often with conflicting results[14,32,37]. In particular, several works focused on the highly frustrated regime, which turns out to be challenging for numerical methods[14,16,21–23,28–32,37–42] and for this reason it is widely recognized as the benchmark model for validating new approaches. Here, we will focus on the particularly challenging case with $J_2/J_1 = 0.5$ on the 10×10 cluster, where the exact solution is not known.

Within variational methods, one of the main difficulties comes from the fact that the sign structure of the ground state is not known for $J_2/J_1 > 0$. Indeed, the Marshall sign rule[43] gives the correct signs (for every cluster size) only when $J_2 = 0$. However, in order to stabilize the optimizations, many previous works imposed the Marshall sign rule as a first approximation for the sign structure (see *Marshall prior* in Table 1). By contrast, within the present approach, we do not need to use any prior knowledge of the signs, thus defining a very general and flexible variational *Ansatz*.

In the following, we first show the alternative SR formulation, then discuss the Deep Transformer architecture, recently introduced by some of us[33,34] as a variational state, and finally present the results obtained by combining the two techniques on the J_1–J_2 Heisenberg model.

Results

Stochastic reconfiguration

Finding the ground state of a quantum system with the variational principle involves minimizing the variational energy $E_\theta = \langle \Psi_\theta | \hat{H} | \Psi_\theta \rangle / \langle \Psi_\theta | \Psi_\theta \rangle$, where $|\Psi_\theta\rangle$ is a variational state parametrized through a vector $\boldsymbol{\theta}$ of P real parameters; in case of complex parameters, we can treat their real and imaginary parts separately[44]. For a system of N 1/2-spins, $|\Psi_\theta\rangle$ can be expanded in the computational basis $|\Psi_\theta\rangle = \sum_{\{\sigma\}} \Psi_\theta(\sigma) |\sigma\rangle$, where $\Psi_\theta(\sigma) = \langle \sigma | \Psi_\theta \rangle$ is a map from spin configurations of the Hilbert space, $\{|\sigma\rangle = |\sigma_1^z, \sigma_2^z, \cdots, \sigma_N^z\rangle,\ \sigma_i^z = \pm 1\}$, to complex numbers. In a gradient-based optimization approach, the fundamental ingredient is the evaluation of the gradient of the loss, which in this case is the variational energy, with respect to the parameters θ_α for $\alpha = 1, \ldots, P$. This gradient can be expressed as a correlation function[44]

$$F_\alpha = -\frac{\partial E_\theta}{\partial \theta_\alpha} = -2\Re\left[\langle (\hat{H} - \langle \hat{H} \rangle)(\hat{O}_\alpha - \langle \hat{O}_\alpha \rangle) \rangle\right], \quad (2)$$

which involves the diagonal operator $\hat{O}_\alpha$ defined as $O_\alpha(\sigma) = \partial \mathrm{Log}[\Psi_\theta(\sigma)]/\partial\theta_\alpha$. The expectation values $\langle \ldots \rangle$ are computed with respect to the variational state. The SR updates[10,20,44] are constructed according to the geometric structure of the landscape:

$$\delta\boldsymbol{\theta} = \tau\left(S + \lambda \mathbb{I}_P\right)^{-1} \boldsymbol{F}, \quad (3)$$

where τ is the learning rate and λ is a regularization parameter to ensure the invertibility of the S matrix. The matrix S has shape $P \times P$ and it is defined in terms of the $\hat{O}_\alpha$ operators[44]

$$S_{\alpha,\beta} = \Re\left[\langle (\hat{O}_\alpha - \langle \hat{O}_\alpha \rangle)^\dagger (\hat{O}_\beta - \langle \hat{O}_\beta \rangle) \rangle\right]. \quad (4)$$

Eq. (3) defines the standard formulation of the SR, which involves the inversion of a $P \times P$ matrix, being the bottleneck of this approach when the number of parameters is larger than $O(10^4)$. To address this problem, we start reformulating Eq. (3) in a more convenient way. For a given sample of M spin configurations $\{\sigma_i\}$ (sampled according to $|\Psi_\theta(\sigma)|^2$), the stochastic

Table 1 | Ground-state energy on the 10 × 10 square lattice at $J_2/J_1 = 0.5$

Energy per site	Wave function	# parameters	Marshall prior	Reference	Year
−0.48941(1)	MLP	893994	Not available	42	2023
−0.494757(12)	CNN	Not available	No	23	2020
−0.4947359(1)	Shallow CNN	11009	Not available	22	2018
−0.49516(1)	Deep CNN	7676	Yes	21	2019
−0.495502(1)	PEPS + Deep CNN	3531	No	41	2021
−0.495530	DMRG	8192 SU(2) states	No	32	2014
−0.495627(6)	aCNN	6538	Yes	40	2023
−0.49575(3)	RBM-fermionic	2000	Yes	16	2019
−0.49586(4)	CNN	10952	Yes	39	2023
−0.4968(4)	RBM ($p = 1$)	Not available	Yes	38	2022
−0.49717(1)	Deep CNN	106529	Yes	29	2022
−0.497437(7)	GCNN	67548	No	28	2023
−0.497468(1)	Deep CNN	421953	Yes	31	2022
−0.4975490(2)	VMC ($p = 2$)	5	Yes	14	2013
−0.497627(1)	Deep CNN	146320	Yes	30	2023
−0.497629(1)	RBM + PP	13132	Yes	37	2021
−0.497634(1)	Deep ViT	267720	No	Present work	2023

estimate of F_α can be obtained as:

$$\bar{F}_\alpha = -\Re\left[\frac{2}{M}\sum_{i=1}^{M}[E_{Li} - \bar{E}_L]^*[O_{\alpha i} - \bar{O}_\alpha]\right]. \tag{5}$$

Here, $E_{Li} = \langle\sigma_i|\hat{H}|\Psi_\theta\rangle/\langle\sigma_i|\Psi_\theta\rangle$ defines the local energy for the configuration $|\sigma_i\rangle$ and $O_{\alpha i} = O_\alpha(\sigma_i)$; in addition, $\bar{E}_L$ and $\bar{O}_\alpha$ denote sample means. Throughout this work, we adopt the convention of using latin and greek indices to run over configurations and parameters, respectively. Equivalently, Eq. (4) can be stochastically estimated as

$$\bar{S}_{\alpha,\beta} = \Re\left[\frac{1}{M}\sum_{i=1}^{M}\left[O_{\alpha i} - \bar{O}_\alpha\right]^*\left[O_{\beta i} - \bar{O}_\beta\right]\right]. \tag{6}$$

To simplify further, we introduce $Y_{\alpha i} = (O_{\alpha i} - \bar{O}_\alpha)/\sqrt{M}$ and $\varepsilon_i = -2[E_{Li} - \bar{E}_L]^*/\sqrt{M}$, allowing us to express Eq. (5) in matrix notation as $\bar{\boldsymbol{F}} = \Re[Y\boldsymbol{\varepsilon}]$ and Eq. (4) as $\bar{S} = \Re[YY^\dagger]$. Writing $Y = Y_R + iY_I$ we obtain:

$$\bar{S} = Y_R Y_R^T + Y_I Y_I^T = XX^T \tag{7}$$

where $X = \text{Concat}(Y_R, Y_I) \in \mathbb{R}^{P\times 2M}$, the concatenation being along the last axis. Furthermore, using $\varepsilon = \varepsilon_R + i\varepsilon_I$, the gradient of the energy can be recast as

$$\bar{\boldsymbol{F}} = Y_R\boldsymbol{\varepsilon}_R - Y_I\boldsymbol{\varepsilon}_I = X\boldsymbol{f}, \tag{8}$$

with $\boldsymbol{f} = \text{Concat}(\boldsymbol{\varepsilon}_R, -\boldsymbol{\varepsilon}_I) \in \mathbb{R}^{2M}$. Then, the update of the parameters in Eq. (3) can be written as

$$\boldsymbol{\delta\theta} = \tau(XX^T + \lambda\mathbb{I}_P)^{-1}X\boldsymbol{f}. \tag{9}$$

This reformulation of the SR updates is a crucial step, which allows the use of the well-known push-through identity $(AB + \lambda\mathbb{I}_n)^{-1}A = A(BA + \lambda\mathbb{I}_m)^{-1}$[45,46], where A and B are respectively matrices with dimensions $n\times m$ and $m\times n$ (see "Methods" for a derivation). As a result, Eq. (9) can be rewritten as

$$\boldsymbol{\delta\theta} = \tau X(X^TX + \lambda\mathbb{I}_{2M})^{-1}\boldsymbol{f}. \tag{10}$$

This derivation is our first result: it shows, in a simple and transparent way, how to exactly perform the SR with the inversion of a $2M\times 2M$ matrix and, therefore, without allocating a $P\times P$ matrix. We emphasize that the last formulation is very useful in the typical deep learning setup, where $P \gg M$. Employing Eq. (10) instead of Eq. (9) proves to be more efficient in terms of both computational complexity and memory usage. The required operations for this new formulation are $O(M^2P) + O(M^3)$ instead of $O(P^3)$, and the memory usage is only $O(MP)$ instead of $O(P^2)$. For deep neural networks with n_l layers the memory usage can be further reduced roughly to $O(MP/n_l)$ (see ref. 47). We developed a memory-efficient implementation of SR that is optimized for deployment on a multi-node GPU cluster, ensuring scalability and practicality for real-world applications (see "Methods"). Other methods, based on iterative solvers, require $O(nMP)$ operations, where n is the number of steps needed to solve the linear problem in Eq. (3). However, this number increases significantly for ill-conditioned matrices (the matrix S has a number of zero eigenvalues equal to $P - M$), leading to many non-parallelizable iteration steps and consequently higher computational costs[48]. Our proof also highlights that the diagonal-shift regularization of the S matrix in parameter space [see Eq. (3)] is equivalent to the same diagonal shift in sample space [see Eq. (10)]. Furthermore, it would be interesting to explore the applicability of regularization schemes with parameter-dependent diagonal shifts in the sample space[28,49]. In contrast, for the MinSR update[30], a pseudo-inverse regularization is applied in order to truncate the effect of vanishing singular values during inversion.

The variational wave function

The architecture of the variational state employed in this work is based on the Deep ViT introduced and described in detail in ref. 34. In the Deep ViT, the input configuration σ of shape $L\times L$ is initially split into $b\times b$ square patches, which are then linearly projected in a d-dimensional vector space in order to obtain an input sequence of L^2/b^2 vectors, i.e., $(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{L^2/b^2})$ with $\boldsymbol{x}_i \in \mathbb{R}^d$. This sequence is then processed by an encoder block employing Multi-Head Factored Attention (MHFA) mechanism[33,50–52]. This produces another output sequence of vectors $(\boldsymbol{A}_1, \ldots, \boldsymbol{A}_{L^2/b^2})$ with $\boldsymbol{A}_i \in \mathbb{R}^d$ and can be formally implemented as follows:

$$A_{i,p} = \sum_{q=1}^{d} W_{p,q}\sum_{j=1}^{L^2/b^2}\alpha_{i,j}^{\mu(q)}\sum_{r=1}^{d} V_{q,r}x_{j,r}, \tag{11}$$

where $\mu(q) = \lceil q\,h/d\rceil$ select the correct attention weights of the corresponding head, being h the total number of heads. The matrix $V \in \mathbb{R}^{d\times d}$ linearly transforms each input vector identically and independently. Instead, the attention matrices $\alpha^\mu \in \mathbb{R}^{L^2/b^2\times L^2/b^2}$ combine the different input vectors and the linear transformation $W \in \mathbb{R}^{d\times d}$ mixes the representations of the different heads.

Due to the global receptive field of the attention mechanism, its computational complexity scales quadratically with respect to the length of the input sequence. Note that in the standard dot product self-attention[3], the attention weights $\alpha_{i,j}$ are function of the inputs, while in the factored case employed here, the attention weights are input-independent variational parameters. This choice is the only custom modification that we employ with respect to the standard Vision Transformer architecture[53], which is also supported by numerical simulations and analytical arguments suggesting that queries and keys do not improve the performance in these problems[52]. Finally, the resulting vectors $\boldsymbol{A}_i$ are processed by a two layers fully connected network (FFN) with hidden size $2d$ and ReLu activation function. Skip connections and pre-layer normalization are implemented as described in refs. 34,54. Generally, a total of n_l such blocks are stacked together to obtain a Deep Transformer. This architecture operates on an input sequence, yielding another sequence of vectors $(\boldsymbol{y}_1, \ldots, \boldsymbol{y}_{L^2/b^2})$, where each $\boldsymbol{y}_i \in \mathbb{R}^d$. At the end, a complex-valued fully connected output layer, with $\log[\cosh(\cdot)]$ as activation function, is applied to $\boldsymbol{z} = \sum_i \boldsymbol{y}_i$ to produce a single number $\text{Log}[\Psi_\theta(\sigma)]$ and predict both the modulus and the phase of the input configuration (refer to Algorithm 1 in "Methods" for a pseudocode of the neural-network architecture).

To enforce translational symmetry, we define the attention weights as $\alpha_{i,j}^\mu = \alpha_{i-j}^\mu$, thereby ensuring translational symmetry among patches. This choice reduces the computational cost during the restoration of the full translational symmetry through quantum number projection[39,55]. Specifically, only a summation involving b^2 terms is required. Under the specific assumption of translationally invariant attention weights, the factored attention mechanism can be technically implemented as a convolutional layer with d input channels, d output channels and a very specific choice of the convolutional kernel: $K_{p,r,k} = \sum_{q=1}^{d} W_{p,q}\alpha_k^{\mu(q)}\sum_{r=1}^{d} V_{q,r} \in \mathbb{R}^{d\times d\times L^2/b^2}$. However, it is well-established that weight sharing and low-rank factorizations in learnable tensors within neural networks can lead to significantly different learning dynamics and, consequently, different final solutions[56–58]. Other symmetries of the Hamiltonian in Eq. (1) [rotations, reflections (C_{4v} point group) and spin parity] can also be restored within quantum number projection. As a result, the symmetrized wave function reads:

$$\tilde{\Psi}_\theta(\sigma) = \sum_{i=0}^{b^2-1}\sum_{j=0}^{7}\left[\Psi_\theta(T_iR_j\sigma) + \Psi_\theta(-T_iR_j\sigma)\right]. \tag{12}$$

In the last equation, T_i and R_j are the translation and the rotation/reflection operators, respectively. Furthermore, due to the $SU(2)$ spin symmetry of the J_1–J_2 Heisenberg model, the total magnetization is also conserved and the Monte Carlo sampling (see "Methods") can be limited in the $S^z = 0$ sector for the ground-state search.

Numerical calculations

Our objective is to approximate the ground state of the J_1–J_2 Heisenberg model in the highly frustrated point $J_2/J_1 = 0.5$ on the 10×10 square lattice. We use the formulation of the SR in Eq. (10) to optimize a variational wave function parametrized through a Deep ViT, as discussed above. The result in Table 1 is achieved with the symmetrized Deep ViT in Eq. (12) using $b = 2$, $n_l = 8$ layers, embedding dimension $d = 72$, and $h = 12$ heads per layer. This variational state has in total 267,720 real parameters (the complex-valued parameters of the output layer are treated as couples of independent real-valued parameters). Regarding the optimization protocol, we choose the learning rate $\tau = 0.03$ (with cosine decay annealing) and the number of samples is fixed to be $M = 6000$. We emphasize that using Eq. (9) to optimize this number of parameters would be infeasible on available GPUs: the memory requirement would be more than $O(10^3)$ gigabytes, one order of magnitude bigger than the highest available memory capacity. Instead, with the formulation of Eq. (10), the memory requirement can be easily handled by available GPUs (see "Methods"). The simulations took 4 days on twenty A100 GPUs. Remarkably, as illustrated in Table 1, we are able to obtain the state-of-the-art ground-state energy using an architecture solely based on neural networks, without using any other regularization than the diagonal shift reported in Eq. (10), fixed to $\lambda = 10^{-4}$. We stress that a completely unbiased simulation, without assuming any prior for the sign structure, is performed, in contrast to other cases where the Marshall sign rule is used to stabilize the optimization[21,29–31,37] (see Table 1). Furthermore, we verified with numerical simulations that the final results are not affected by the Marshall prior. This is an important point since a simple sign prior is not available for the majority of the models (e.g., the Heisenberg model on the triangular or Kagome lattices). We would like also to mention that the definition of a suitable architecture is fundamental to take advantage of having a large number of parameters. Indeed, while a stable simulation with a simple regularization scheme (only defined by a finite value of λ) is possible within the Deep ViT wave function, other architectures require more sophisticated regularizations. For example, to optimize Deep GCNNs it is necessary to add a temperature-dependent term to the loss function[28] or, for Deep CNNs, a process of variance reduction and reweighting[30] helps in escaping local minima. We also point out that physically inspired wave functions, as the Gutzwiller-projected states[14], which give a remarkable result with only a few parameters, are not always equally accurate in other cases.

In Fig. 1 we show a typical Deep ViT optimization on the 10×10 lattice at $J_2/J_1 = 0.5$. First, we optimize the Transformer having translational invariance among patches (blue curve). Then, starting from the previously optimized parameters, we restore sequentially the full translational invariance (green curve), rotational symmetry (orange curve) and lastly, reflections and spin parity symmetry (red curve). Whenever a new symmetry is restored, the energy reliably decreases[55]. We stress that our optimization process, which combines the SR formulation of Eq. (10) with a real-valued Deep ViT followed by a complex-valued fully connected output layer[34], is highly stable and insensitive to the initial seed, ensuring consistent results across multiple optimization runs.

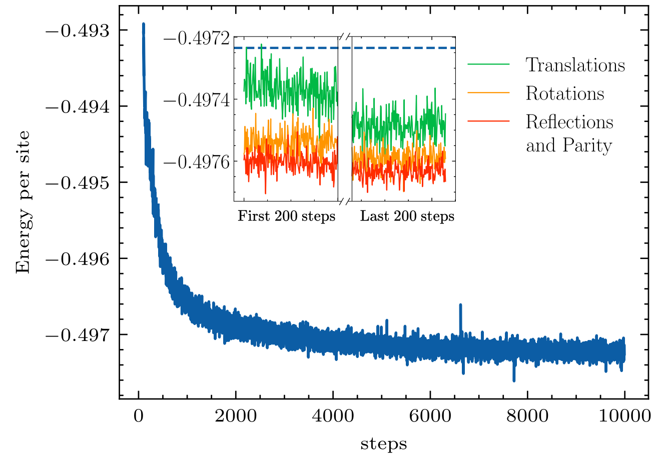

Fig. 1 | Variational energy optimization. Optimization of the Deep ViT with patch size $b = 2$, $n_l = 8$ layers, embedding dimension $d = 72$ and $h = 12$ heads per layer, on the J_1–J_2 Heisenberg model at $J_2/J_1 = 0.5$ on the 10×10 square lattice. The first 200 optimization steps are not shown for better readability. Inset: first and last 200 optimization steps when recovering sequentially the full translational (green curve), rotational (orange curve), and reflections and parity (red curve) symmetries. The total number of steps after restoring the symmetries is 5000 for translations, 5000 for rotations, and 4000 for reflections and parity. The mean energy obtained without quantum number projection is also reported for comparison (blue dashed line).

Discussion

We have introduced a formulation of the SR that excels in scenarios where the number of parameters (P) significantly outweighs the number of samples (M) used for stochastic estimations. Exploiting this approach, we attained the state-of-the-art ground-state energy for the J_1–J_2 Heisenberg model at $J_2/J_1 = 0.5$, on a 10×10 square lattice, optimizing a Deep ViT with $P = 267{,}720$ parameters and using only $M = 6000$ samples. It is essential to note that this achievement highlights the remarkable capability of deep learning in performing exceptionally well even with a limited sample size relative to the overall parameter count. This also challenges the common belief that a large amount of Monte Carlo samples are required to find the solution in the exponentially large Hilbert space and for precise SR optimizations[31].

Our results have important ramifications for investigating the physical properties of challenging quantum many-body problems, where the use of the SR is crucial to obtain accurate results. The use of large-scale neural-network quantum states can open new opportunities in approximating ground states of quantum spin Hamiltonians, where other methods fail. Additionally, making minor modifications to the equations describing parameter updates within the SR framework [see Eq. (10)] enables us to describe the unitary time evolution of quantum many-body systems according to the time-dependent variational principle[44,59,60]. Extending our approach to address time-dependent problems stands as a promising avenue for future works. Furthermore, this formulation of the SR can be a key tool for obtaining accurate results also in quantum chemistry, especially for systems that suffer from the sign problem[61]. Typically, in this case, the standard strategy is to take $M > 10 \times P$[44], which may be unnecessary for deep learning-based approaches.

Methods

Linear algebra identity

The key point of the method is the transformation from Eq. (9) to Eq. (10), which uses the matrix identity

$$(AB + \lambda \mathbb{I}_n)^{-1} A = A(BA + \lambda \mathbb{I}_m)^{-1} , \quad (13)$$

where A and B are $n \times m$ and $m \times n$ matrices, respectively. This identity can be proved starting from

$$\mathbb{I}_m = (BA + \lambda \mathbb{I}_m)(BA + \lambda \mathbb{I}_m)^{-1} , \quad (14)$$

then, multiplying from the left by A, we get

$$A = A(BA + \lambda \mathbb{I}_m)(BA + \lambda \mathbb{I}_m)^{-1} , \quad (15)$$

and exploiting the fact that $A\mathbb{I}_m = \mathbb{I}_n A$, we obtain

$$A = (AB + \lambda \mathbb{I}_n) A (BA + \lambda \mathbb{I}_m)^{-1} . \quad (16)$$

At the end, multiplying from the left by $(AB + \lambda \mathbb{I}_n)^{-1}$, we recover Eq. (13).

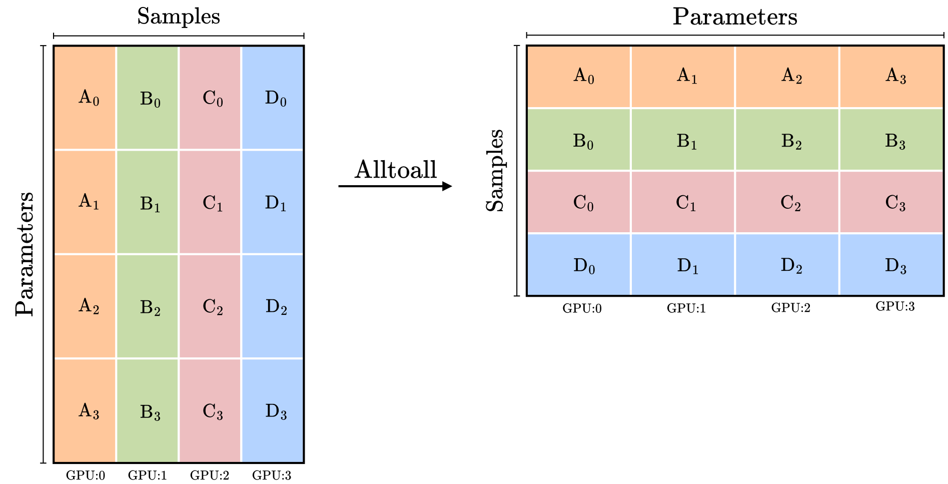

Fig. 2 | Message passing interface (MPI) *alltoall* operation. Graphical representation of MPI *alltoall* operation to transpose the X matrix distributed across multiple GPUs. For example, GPU:0 initially contains sub-matrices A_0, A_1, A_2, A_3, while following the transposition, GPU:0 contains sub-matrices A_0, B_0, C_0, D_0.

The identity in Eq. (13) is also used in the *kernel trick*, which is at the basis of kernel methods which have applications in many areas of machine learning[62], including many-body quantum systems[63].

Distributed SR computation

The algorithm proposed in Eq. (10) can be efficiently distributed, both in terms of computational operations and memory, across multiple GPUs. To illustrate this, we consider for simplicity the case of a real-valued wave function, where $X = Y_R \equiv Y$. Given a number M of configurations, they can be distributed across n_G GPUs, facilitating parallel simulation of Markov chains. In this way, on the gth GPU, the elements $i \in [gM/n_G, (g+1)M/n_G)$ of the vector $\boldsymbol{f}$ are obtained, along with the columns $i \in [gM/n_G, (g+1)M/n_G)$ of the matrix X, which we indicate using $X_{[:,g]}$. To efficiently apply Eq. (10), we employ the message passing interface (MPI) *alltoall* collective operation to transpose X, yielding the sub-matrix $X_{[g,:]}$ on gth GPU. This sub-matrix comprises the rows elements in $[gP/n_G, (g+1)P/n_G)$ of the original matrix X (see Fig. 2). Consequently, we can express:

$$X^T X = \sum_{g=0}^{n_G-1} X^T_{[g,:]} X_{[g,:]} \,. \tag{17}$$

The inner products can be computed in parallel on each GPU, while the outer sum is performed using the MPI primitive *reduce* with the *sum* operation. The *master* GPU performs the inversion, computes the vector $\boldsymbol{t} = (X^T X + \lambda \mathbb{I}_{2M})^{-1} \boldsymbol{f}$, and then scatters it across the other GPUs. Finally, after transposing again the matrix X with the MPI *alltoall* operation, the parameter update can be computed as follows:

$$\delta\boldsymbol{\theta} = \tau \sum_{g=0}^{n_G-1} X_{[:,g]} \boldsymbol{t}_g \,. \tag{18}$$

This procedure significantly reduces the memory requirement per GPU to $O(MP/n_G)$, enabling the optimization of an arbitrary number of parameters using the SR approach. In Fig. 3, we report the memory usage and the computational time per optimization step.

Systematic energy improvement

Here, we discuss the impact of the number of samples M and parameters P on the variational energy of the ViT wave function, showing the results in Fig. 4. In the right panel, we show the variational energy as a function of the number of parameters for a fixed number of layers $n_l = 8$, performing the optimizations with $M = 6000$ samples. The number of parameters is increased by enlarging the width of each layer. In particular, we take the following architectures: ($h = 10, d = 40$), ($h = 10, d = 60$), ($h = 12, d = 72$), and ($h = 14, d = 140$) with $P = 85400$, 187100, 267720, and 994,700 parameters, respectively. A similar analysis for a fixed width but varying the number of layers is discussed in ref. 34. Instead, in the left panel, we fix an architecture ($h = 12, d = 72$) with $n_l = 8$ and increase the number of samples M up to 10^4. Both analyses are performed without restoring the symmetries by quantum number projection; for comparison, we report in the left panel the energy obtained after restoring the symmetries [see Eq. (12)]. The latter one coincides with the ViT wave function used to obtain the energy reported in Table 1. We point out that the energy curves depicted in Fig. 4 are obtained from unbiased simulations, without the utilization of Marshall sign prior. The final value of the energy, as well as the convergence to it, are qualitatively similar regardless of its inclusion.

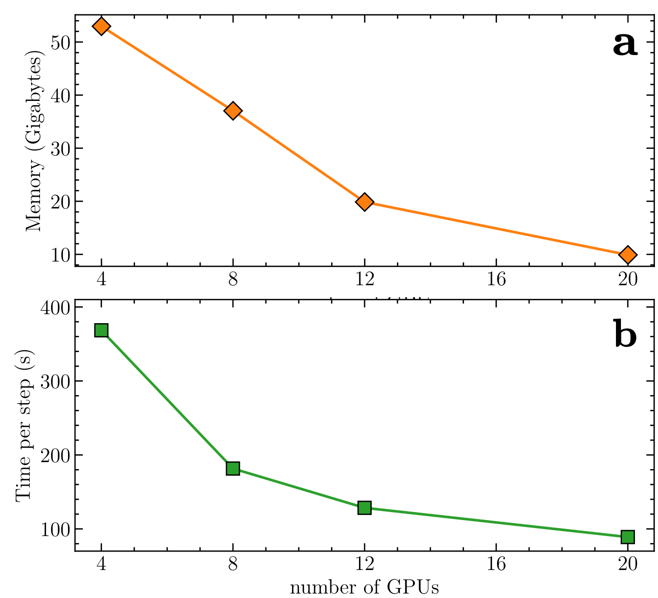

Fig. 3 | Memory and time usage. Memory usage in gigabytes (**a**) and computational time per optimization step in seconds (**b**) as a function of the number of GPUs. The reported values are related to a ViT architecture with $h = 12$, $d = 72$, $n_l = 8$, fully symmetrized [see Eq. (12)] and optimized with $M = 6000$ samples.

Details on the Transformer architecture

In this section, we provide a pseudocode (see Algorithm 1) describing the steps for the implementation of the Vision Transformer architecture

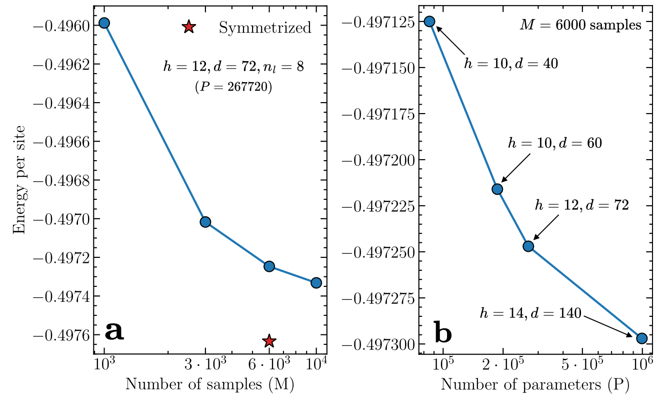

Fig. 4 | Variational energy scalings. a Energy per site as a function of the number of samples M for a ViT with $n_l = 8$ layers, embedding dimension $d = 72$ and $h = 12$ heads per layer. **b** Energy per site as a function of the number of parameters P, increased by adding heads h and taking larger embedding dimensions d, for a fixed number of layers $n_l = 8$. For both panels, the energy values (blue circles) are obtained without restoring the symmetries; for comparison, we also show the energy corresponding to the fully symmetrized state in Eq. (12) (red star) which is the one reported in Table 1.

employed in this work and described in the "Results" section. In particular, we emphasize that skip connections and layer normalization are implemented as described in ref. 54.

Algorithm 1. **Vision Transformer wave function**

Input configuration $\sigma \in \{-1, 1\}^{L\times L}$
Patch and Embed: $\mathcal{X} \leftarrow (\boldsymbol{x}_1, \ldots \boldsymbol{x}_{L^2/b^2}) \in \mathbb{R}^{L^2/b^2 \times d}$
for $i = 1, n_l$ **do**
$\mathcal{X} \leftarrow \mathcal{X} +$ MHFA(LayerNorm($\mathcal{X}$))
$\mathcal{X} \leftarrow \mathcal{X} +$ FFN(LayerNorm($\mathcal{X}$))
end for
$(\boldsymbol{y}_1, \ldots \boldsymbol{y}_{L^2/b^2}) \leftarrow$ LayerNorm$(\mathcal{X})$
$\boldsymbol{z} \leftarrow \sum_{i=1}^{d} \boldsymbol{y}_i$
$\mathrm{Log}[\Psi_\theta(\sigma)] \leftarrow \sum_{\alpha=1}^{d} g(b_\alpha + \boldsymbol{w}_\alpha \cdot \boldsymbol{z})$

Monte Carlo sampling

The expectation value of an operator $\hat{A}$ on a given variational state $|\Psi_\theta\rangle$ can be computed as

$$\langle \hat{A} \rangle = \frac{\langle \Psi_\theta | \hat{A} | \Psi_\theta \rangle}{\langle \Psi_\theta | \Psi_\theta \rangle} = \sum_{\{\sigma\}} P_\theta(\sigma) A_L(\sigma)\,, \tag{19}$$

where $P_\theta(\sigma) \propto |\Psi_\theta(\sigma)|^2$ and $A_L(\sigma) = \langle \sigma | \hat{A} | \Psi_\theta \rangle / \langle \sigma | \Psi_\theta \rangle$ is the so-called local estimator of $\hat{A}$

$$A_L(\sigma) = \sum_{\{\sigma'\}} \langle \sigma | \hat{A} | \sigma' \rangle \frac{\Psi_\theta(\sigma')}{\Psi_\theta(\sigma)}\,. \tag{20}$$

For any local operator (e.g., the Hamiltonian) the matrix $\langle \sigma | \hat{A} | \sigma' \rangle$ is sparse, then the calculation of $A_L(\sigma)$ is at most polynomial in the number of spins. Furthermore, if it is possible to efficiently generate a sample of configurations $\{\sigma_1, \sigma_2, \ldots, \sigma_M\}$ from the distribution $P_\theta(\sigma)$ (e.g., by performing a Markov chain Monte Carlo), then Eq. (19) can be used to obtain a stochastic estimation of the expectation value

$$\bar{A} = \frac{1}{M} \sum_{i=1}^{M} A_L(\sigma_i)\,, \tag{21}$$

where $\bar{A} \approx \langle \hat{A} \rangle$ and the accuracy of the estimation is controlled by a statistical error which scales as $O(1/\sqrt{M})$.

Note added to the proof. During the revision process, we became aware of an updated version of Ref. 30 where a variational energy per site of -0.4976921(4) has been obtained.

Data availability

The data that support the findings of this study are available from the corresponding author upon reasonable request.

Code availability

The variational quantum Monte Carlo and the Deep ViT architecture were implemented in JAX[64]. The parallel implementation of the stochastic reconfiguration was implemented using mpi4jax[65], and it is available on NetKet[48], under the name of `VMC_SRt`. The ViT architecture will be made available from the authors upon reasonable request.

Received: 23 October 2023; Accepted: 3 July 2024;
Published online: 02 August 2024

References

1. Krizhevsky, A., Sutskever, I. & Hinton, G. E. Imagenet classification with deep convolutional neural networks. in *Advances in Neural Information Processing Systems* Vol. 25 (eds Pereira, F. et al.) (Curran Associates, Inc., 2012). https://proceedings.neurips.cc/paper_files/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf.
2. He, K., Zhang, X., Ren, S. & Sun, J. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* 770–778 (IEEE, 2016).
3. Vaswani, A. et al. Attention is all you need. In *Advances in Neural Information Processing Systems 30* (Curran Associates, Inc., 2017).
4. Devlin, J., Chang, M.-W., Lee, K. & Toutanova, K. Bert: pre-training of deep bidirectional transformers for language understanding. In *Proc. 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies* (Association for Computational Linguistics, 2019).
5. Brown, T., Mann, B., Ryder, N. et al. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, Vol. 33, 1877–1901 (Curran Associates, Inc., 2020). https://proceedings.neurips.cc/paper_files/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf.
6. Kingma, D. P. & Ba, J. Adam: a method for stochastic optimization. In *Proc. International Conference on Learning Representations* (ICLR) (ICLR, 2017).
7. Loshchilov, I. & Hutter, F. Decoupled weight decay regularization. In *Proc. International Conference on Learning Representations (ICLR)* (OpenReview.net, 2019).
8. Amari, S. & Douglas, S. Why natural gradient? In *Proc. 1998 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP '98 (Cat. No. 98CH36181)*, Vol. 2, 1213–1216 (IEEE, 1998).
9. Amari, S., Karakida, R. & Oizumi, M. Fisher information and natural gradient learning in random deep networks. In *Proc. Twenty-Second International Conference on Artificial Intelligence and Statistics, Proceedings of Machine Learning Research* (eds. Chaudhuri, K. & Sugiyama, M.) Vol. 8, 9694–702 (PMLR, 2019). https://proceedings.mlr.press/v89/amari19a.html.
10. Sorella, S. Green function Monte Carlo with stochastic reconfiguration. *Phys. Rev. Lett.* **80**, 4558–4561 (1998).
11. Sorella, S. Wave function optimization in the variational Monte Carlo method. *Phys. Rev. B* **71**, 241103 (2005).
12. Park, C. Y. & Kastoryano, M. J. Geometry of learning neural quantum states. *Phys. Rev. Res.* **2**, 023232 (2020).
13. Capello, M., Becca, F., Fabrizio, M., Sorella, S. & Tosatti, E. Variational description of Mott insulators. *Phys. Rev. Lett.* **94**, 026406 (2005).
14. Hu, W.-J., Becca, F., Parola, A. & Sorella, S. Direct evidence for a gapless Z_2 spin liquid by frustrating néel antiferromagnetism. *Phys. Rev. B* **88**, 060402 (2013).

15. Carleo, G. & Troyer, M. Solving the quantum many-body problem with artificial neural networks. *Science* **355**, 602–606 (2017).
16. Ferrari, F., Becca, F. & Carrasquilla, J. Neural Gutzwiller-projected variational wave functions. *Phys. Rev. B* **100**, 125131 (2019).
17. Nomura, Y., Darmawan, A. S., Yamaji, Y. & Imada, M. Restricted Boltzmann machine learning for solving strongly correlated quantum systems. *Phys. Rev. B* **96**, 205152 (2017).
18. Viteritti, L., Ferrari, F. & Becca, F. Accuracy of restricted Boltzmann machines for the one-dimensional J_1–J_2 Heisenberg model. *SciPost Phys.* **12**, 166 (2022).
19. Park, C.-Y. & Kastoryano, M. J. Expressive power of complex-valued restricted Boltzmann machines for solving nonstoquastic Hamiltonians. *Phys. Rev. B* **106**, 134437 (2022).
20. Nomura, Y. Boltzmann machines and quantum many-body problems. *J. Phys. Condens. Matter* **36**, 073001 (2023).
21. Choo, K., Neupert, T. & Carleo, G. Two-dimensional frustrated J_1–J_2 model studied with neural network quantum states. *Phys. Rev. B* **100**, 125124 (2019).
22. Liang, X. et al. Solving frustrated quantum many-particle models with convolutional neural networks. *Phys. Rev. B* **98**, 104426 (2018).
23. Szabó, A. & Castelnovo, C. Neural network wave functions and the sign problem. *Phys. Rev. Res.* **2**, 033075 (2020).
24. Hibat-Allah, M., Ganahl, M., Hayward, L. E., Melko, R. G. & Carrasquilla, J. Recurrent neural network wave functions. *Phys. Rev. Res.* **2**, 023358 (2020).
25. Roth, C. Iterative retraining of quantum spin models using recurrent neural networks. Preprint at https://doi.org/10.48550/arXiv.2003.06228 (2020).
26. Hibat-Allah, M., Melko, R. G. & Carrasquilla, J. Supplementing recurrent neural network wave functions with symmetry and annealing to improve accuracy. In *Fourth Workshop on Machine Learning and the Physical Sciences* (NeurIPS, 2022).
27. Hibat-Allah, M., Melko, R. G. & Carrasquilla, J. Investigating topological order using recurrent neural networks. *Phys. Rev. B* **108**, arXiv:2303.11207 (2023).
28. Roth, C., Szabó, A. & MacDonald, A. H. High-accuracy variational monte carlo for frustrated magnets with deep neural networks. *Phys. Rev. B* **108**, 054410 (2023).
29. Li, M. et al. Bridging the gap between deep learning and frustrated quantum spin system for extreme-scale simulations on new generation of Sunway supercomputer. *IEEE Trans. Parallel Distrib. Syst.* **33**, 2846–2859 (2022).
30. Chen, A. & Heyl, M. Empowering deep neural quantum states through efficient optimization. *Nat. Phys.* https://doi.org/10.1038/s41567-024-02566-1 (2023).
31. Liang, X. et al. Deep learning representations for quantum many-body systems on heterogeneous hardware. *Mach. Learn.: Sci. Technol.* **4**, 015035 (2022).
32. Gong, S.-S., Zhu, W., Sheng, D. N., Motrunich, O. I. & Fisher, M. P. A. Plaquette ordered phase and quantum phase diagram in the spin-$\frac{1}{2}J_1$–J_2 square Heisenberg model. *Phys. Rev. Lett.* **113**, 027201 (2014).
33. Viteritti, L. L., Rende, R. & Becca, F. Transformer variational wave functions for frustrated quantum spin systems. *Phys. Rev. Lett.* **130**, 236401 (2023).
34. Viteritti, L. L., Rende, R., Parola, A., Goldt, S. & Becca, F. Transformer wave function for the Shastry-Sutherland model: emergence of a spin-liquid phase. Preprint at https://doi.org/10.48550/arXiv.2311.16889 (2023).
35. Sprague, K. & Czischek, S. Variational Monte Carlo with large patched transformers. *Commun. Phys.* https://doi.org/10.1038/s42005-024-01584-y (2023).
36. Luo, D. et al. Gauge-invariant and anyonic-symmetric autoregressive neural network for quantum lattice models. *Phys. Rev. Res.* **5**, 013216 (2023).
37. Nomura, Y. & Imada, M. Dirac-type nodal spin liquid revealed by refined quantum many-body solver using neural-network wave function, correlation ratio, and level spectroscopy. *Phys. Rev. X* **11**, 031034 (2021).
38. Chen, H., Hendry, D., Weinberg, P. & Feiguin, A. Systematic improvement of neural network quantum states using Lanczos. in *Advances in Neural Information Processing Systems* Vol. 35 (eds. Koyejo, S. et al.) 7490–7503 (Curran Associates, Inc., 2022). https://proceedings.neurips.cc/paper_files/paper/2022/file/3173c427cb4ed2d5eaab029c17f221ae-Paper-Conference.pdf.
39. Reh, M., Schmitt, M. & Gärttner, M. Optimizing design choices for neural quantum states. *Phys. Rev. B* **107**, 195115 (2023).
40. Wang, J.-Q., He, R.-Q. & Lu, Z.-Y. Variational optimization of the amplitude of neural-network quantum many-body ground states. *Phys. Rev. B* **109**, 245120 (2023).
41. Liang, X., Dong, S.-J. & He, L. Hybrid convolutional neural network and projected entangled pair states wave functions for quantum many-particle states. *Phys. Rev. B* **103**, 035138 (2021).
42. Ledinauskas, E. & Anisimovas, E. Scalable imaginary time evolution with neural network quantum states. *SciPost Phys.* **15**, 229 (2023)
43. Marshall, W. Antiferromagnetism. *Proc. R. Soc. Lond. Ser. A Math. Phys. Sci.* **232**, 48–68 (1955).
44. Becca, F. & Sorella, S. Quantum Monte Carlo approaches for correlated systems (Cambridge University Press, 2017).
45. Henderson, H. V. & Searle, S. R. On deriving the inverse of a sum of matrices. *SIAM Rev.* **23**, 53–60 (1981).
46. Petersen, K. B. & Pedersen, M. S. The matrix cookbook. http://www2.compute.dtu.dk/pubdb/pubs/3274-full.html (2012).
47. Novak, R., Sohl-Dickstein, J. & Schoenholz, S. S. Fast finite width neural tangent kernel. In *Proc. 39th International Conference on Machine Learning*, *Proceedings of Machine Learning Research* Vol. 162, 17018–17044 (PMLR, 2022). https://proceedings.mlr.press/v162/novak22a.html.
48. Vicentini, F. et al. NetKet 3: machine learning toolbox for many-body quantum systems. *SciPost Phys. Codebases*. https://doi.org/10.21468/scipostphyscodeb.7 (2022)
49. Lovato, A., Adams, C., Carleo, G. & Rocco, N. Hidden-nucleons neural-network quantum states for the nuclear many-body problem. *Phys. Rev. Res.* **4**, 043178 (2022).
50. Rende, R., Gerace, F., Laio, A. & Goldt, S. Mapping of attention mechanisms to a generalized Potts model. *Phys. Rev. Res.* **6**, 023057 (2023).
51. Bhattacharya, N. et al. Interpreting potts and transformer protein models through the lens of simplified attention. *Pac. Symp. Biocomput.* **27**, 34–45 (2020).
52. Rende, R. & Viteritti, L. L. Are queries and keys always relevant? A case study on transformer wave functions. Preprint at https://doi.org/10.48550/arXiv.2405.18874 (2024).
53. Dosovitskiy, A. et al. An image is worth 16x16 words: transformers for image recognition at scale. In *Proc. International Conference on Learning Representations* (OpenReview.net, 2021).
54. Xiong, R. et al. On layer normalization in the transformer architecture. In *Proc. Machine Learning Research* (JMLR.org, 2020).
55. Nomura, Y. Helping restricted Boltzmann machines with quantum-state representation by restoring symmetry. *J. Phys. Condens. Matter* **33**, 174003 (2021).
56. Urban, G. et al. Do deep convolutional nets really need to be deep and convolutional? In *International Conference on Learning Representations* (2017). https://openreview.net/forum?id=r10FA8Kxg.
57. d'Ascoli, S., Sagun, L., Biroli, G. & Bruna, J. Finding the needle in the haystack with convolutions: on the benefits of architectural bias. in *Advances in Neural Information Processing Systems* Vol. 32 (eds. Wallach, H. et al.) (Curran Associates, Inc., 2019). https://proceedings.neurips.cc/

paper_files/paper/2019/file/124c3e4ada4a529aa0fedece80bb42ab-Paper.pdf.
58. Ingrosso, A. & Goldt, S. Data-driven emergence of convolutional structure in neural networks. *Proc. Natl Acad. Sci. USA* **119**, e2201854119 (2022).
59. Mendes-Santos, T., Schmitt, M. & Heyl, M. Highly resolved spectral functions of two-dimensional systems with neural quantum states. *Phys. Rev. Lett.* **131**, 046501 (2023).
60. Schmitt, M. & Heyl, M. Quantum many-body dynamics in two dimensions with artificial neural networks. *Phys. Rev. Lett.* **125**, 100503 (2020).
61. Nakano, K. et al. TurboRVB: a many-body toolkit for <i>ab initio</i> electronic simulations by quantum Monte Carlo. *J. Chem. Phys.* **152** https://doi.org/10.1063/F5.0005037 (2020).
62. Bishop, C. M. *Pattern Recognition and Machine Learning (Information Science and Statistics)* (Springer-Verlag, Berlin, Heidelberg, 2006).
63. Giuliani, C., Vicentini, F., Rossi, R. & Carleo, G. Learning ground states of gapped quantum Hamiltonians with Kernel methods. *Quantum* **7**, 1096 (2023).
64. Bradbury, J. et al. JAX: composable transformations of Python+NumPy programs. http://github.com/google/jax (2018).
65. Häfner, D. & Vicentini, F. mpi4jax: Zero-copy mpi communication of jax arrays. *J. Open Source Softw.* **6**, 3419 (2021).

Acknowledgements
We thank R. Favata for critically reading the manuscript and M. Imada for stimulating us with challenging questions. We also acknowledge G. Carleo, F. Vicentini, Y. Nomura, A. Szabó, J. Carrasquilla, and A. Chen for useful discussions. The variational quantum Monte Carlo and the Deep ViT architecture were implemented in JAX[64]. The parallel implementation of the stochastic reconfiguration was implemented using mpi4jax[65], and it is available on NetKet[48], under the name of `VMC_SRt`. R.R. and L.L.V. acknowledge the CINECA award under the ISCRA initiative, for the availability of high-performance computing resources and support. S.G. acknowledges co-funding from Next Generation EU, in the context of the National Recovery and Resilience Plan, Investment PE1 – Project FAIR "Future Artificial Intelligence Research". This resource was co-financed by the Next Generation EU [DM 1555 del 11.10.22].

Author contributions
R.R., L.L.V., and L.B. devised the algorithm with input from F.B. and S.G. and performed the numerical simulations. R.R., L.L.V., L.B., F.B., and S.G. wrote the manuscript.

Competing interests
The authors declare no competing interests.

Additional information
Correspondence and requests for materials should be addressed to Riccardo Rende, Luciano Loris Viteritti or Sebastian Goldt.

Peer review information *Communications Physics* thanks the anonymous reviewers for their contribution to the peer review of this work.

Reprints and permissions information is available at http://www.nature.com/reprints

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.