

Diagonal isometric form for tensor network states in two dimensions

Benjamin Sappler^{1,2,*}, Masataka Kawano^{1,2}, Michael P. Zaletel^{3,4}, and Frank Pollmann^{1,2}

¹Physics Department, TUM School of Natural Sciences, *Technical University of Munich*, 85748 Garching, Germany

²Munich Center for Quantum Science and Technology (MCQST), Schellingstr. 4, 80799 München, Germany

³Department of Physics, *University of California, Berkeley*, California 94720, USA

⁴Material Science Division, *Lawrence Berkeley National Laboratory*, Berkeley, California 94720, USA



(Received 13 October 2025; accepted 25 February 2026; published 10 April 2026; corrected 23 July 2026)

Isometric tensor network states (isoTNS) generalize the isometric form of the one-dimensional matrix product states to tensor networks in two and higher dimensions. Here, we introduce an alternative isometric form for isoTNS by incorporating auxiliary tensors to represent the orthogonality hypersurface. We implement the time evolving block decimation algorithm on this new isometric form and benchmark the method by computing ground states and the real time evolution of the transverse field Ising model in two dimensions on large square lattices of up to 1250 sites. Our results demonstrate that isoTNS can efficiently capture the entanglement structure of two-dimensional area law states. The short-time dynamics is also accurately reproduced even at the critical point. Our isoTNS formulation further allows for a natural extension to different lattice geometries, such as the honeycomb or kagome lattice.

DOI: [10.1103/56px-qsx](https://doi.org/10.1103/56px-qsx)

I. INTRODUCTION

Quantum many-body systems can exhibit many fascinating phenomena, including spin liquids with fractionalized excitations [1–4], topological phases of matter [5], and high-temperature superconductivity [6,7]. The numerical treatment of these systems is challenging due to the exponential growth of the Hilbert space dimension with system size. In one spatial dimension (1D) this challenge was addressed by the density matrix renormalization group (DMRG) algorithm [8], which can be understood as a variational method over the class of matrix product states (MPS) [9–11]. The success of DMRG is due to the ability of MPS to efficiently capture the entanglement structure of gapped ground states [12,13], which follows a characteristic area law [14]. Additionally, every MPS can be brought into an isometric form [10,11], which greatly improves the computational cost of MPS algorithms. The natural generalization of MPS to higher dimensional systems is given in the form of projected entangled pair states (PEPS) [11,15–21]. PEPS are able to efficiently represent area law states in two or more spatial dimensions [11]. However, a general PEPS cannot be exactly brought into an isometric form similar to MPS due to the presence of closed loops. Consequently, algorithms on PEPS have computational costs scaling with high powers of the bond dimension D , e.g., $\mathcal{O}(D^{10})$ for a full time evolution update and $\mathcal{O}(D^{12})$ for DMRG [22,23].

Additionally, DMRG on PEPS requires solving a generalized eigenvalue problem, which is numerically ill conditioned [23].

More recently, the new class of isometric tensor network states (isoTNS) has been introduced [24–26], generalizing the isometric form of MPS to higher dimensions by enforcing isometry constraints on the tensors of a PEPS. This allows for the efficient computation of local expectation values and can reduce the computational cost of algorithms compared to PEPS. For example, the cost of real and imaginary time evolution is reduced from $\mathcal{O}(D^{10})$ to $\mathcal{O}(D^7)$ [24]. While the generalization of DMRG to isoTNS still scales as $\mathcal{O}(D^{12})$, the generalized eigenvalue problem reduces to a standard eigenvalue problem due to the isometric form, which is much more stable numerically [27]. These advantages come at a cost, namely that the expressional power of isoTNS is lower than that of PEPS. It was however found that isoTNS with finite bond dimension can exactly represent the ground state wave functions of string-net models [28]. This suggests that long-range entanglement does not form an obstruction for isoTNS representations and that the ground states of gapped 2D Hamiltonians with gappable edges can be efficiently represented as an isoTNS. Consequently, the isometric form of isoTNS is compatible with true 2D geometries. In Ref. [29], topological phase transitions were studied with isoTNS, showing that the *Ansatz* is able to represent certain critical states with power-law correlations. isoTNS were also extended to fermionic systems [30], to 2D strips of infinite length [31], and to 3D cubic lattices [32]. They have further been used to compute properties of 2D thermal states [33]. Additionally, there have been works discussing the computational complexity of isoTNS contractions [34] and relating isoTNS to quantum circuits [35,36].

While algorithms based on isoTNS have shown first promising results, there are still open questions. For example,

*Contact author: benjamin.sappler@tum.de

the best way of defining the isometric form is not yet agreed upon. Different isometric forms could lead to reduced errors, more stable algorithms, and reduced computational cost. Recently, a new isometric form for isoTNS was introduced, where the direction of the isometries along columns alternates between pointing up and down [37]. This isometric form improves the ability of isoTNS to represent many-body states. In this work we propose yet another isometric form for isoTNS. The *Ansatz* differs from the original isoTNS introduced in Ref. [24] by rotating the lattice by 45° and introducing auxiliary tensors with no physical degree of freedom. We find that our *Ansatz* yields ground-state energies for the transverse-field Ising model that are comparable to those obtained with the original isoTNS. The advantages of our *Ansatz* arise from the structural difference of the tensor network compared to the original isoTNS. Most isoTNS algorithms require repeatedly shifting a special column of tensors, the *orthogonality hypersurface*, across the network. Because of the structure of our *Ansatz*, this shift becomes conceptionally simpler compared to the original isoTNS. Furthermore, while in our implementation the shift is still performed sequentially along one column of the network, one could also perform it in parallel at each tensor of the column. In that sense, moving the orthogonality hypersurface in our *Ansatz* can be understood as a local operation, compared to the more global shift in the original isoTNS. A further benefit of our *Ansatz* is the ease with which it generalizes to different lattice geometries; in this work, we demonstrate results on both square and honeycomb lattices.

The paper is structured as follows. First, we give a review of MPS and isoTNS in 2D in Sec. II. We discuss how the isometric form is able to speed up algorithms and review the Moses Move (MM) [24], which is a core part of isoTNS algorithms. In Sec. III we introduce the alternative isometric form and the Yang-Baxter (YB) move as the counterpart of the MM. We further implement an approximative algorithm for computing gradients during the disentangling step, giving similar results to the exact computation while being much faster. Additionally, we generalize the time evolving block decimation (TEBD) algorithm [38], which can be used for real and imaginary time evolution, to this new isometric form. In Sec. IV we benchmark the method by performing ground state search and real time evolution of the transverse field Ising model on large square and honeycomb lattices. Our results demonstrate that our *Ansatz* is able to efficiently capture the entanglement structure of 2D area-law states. We conclude with a discussion of our results in Sec. V.

II. ISOMETRIC TENSOR NETWORK STATES

A general quantum many-body state can be written as

$$|\Psi\rangle = \sum_{i_1=1}^{d_1} \sum_{i_2=1}^{d_2} \cdots \sum_{i_N=1}^{d_N} \Psi_{i_1, i_2, \dots, i_N} |i_1\rangle \otimes |i_2\rangle \otimes \cdots \otimes |i_N\rangle, \quad (1)$$

where N is the number of subsystems (e.g., lattice sites or particles), $\{|i_1\rangle \otimes |i_2\rangle \otimes \cdots \otimes |i_N\rangle\}$ with $i_n = 0, 1, \dots, d_n$ is

a set of basis vectors of the full many-body Hilbert space

$$\mathcal{H} = \bigotimes_{n=1}^N \mathcal{H}_n,$$

and $d_n = \dim(\mathcal{H}_n)$ is the dimension of the local Hilbert space $\mathcal{H}_n$ of subsystem n . To simplify the notation we assume that the dimension of all local Hilbert spaces is equal, $d_n = d$. The d^N complex numbers $\Psi_{i_1, i_2, \dots, i_N}$ fully describe the quantum many-body state and one can think of $\Psi \in \mathbb{C}^{d \times d \times \cdots \times d}$ as a tensor of order N . However, due to the number of parameters scaling exponentially with system size, only very small system sizes are accessible computationally.

One can proceed by approximately decomposing Ψ into a tensor network consisting of tensors of smaller order, with the total number of parameters depending polynomially on system size. If one is interested in computing ground state properties of local, gapped Hamiltonians, it is beneficial to choose a tensor network whose connectivity mimics the spatial structure of the physical system. This way, the tensor network is able to efficiently represent the area law entanglement structure present in the system [11]. For 1D chains, the resulting tensor networks are MPS. For lattice systems of two or higher dimensions, PEPS are the natural generalization. We give a brief review of MPS in Sec. II A and of PEPS and isoTNS in 2D in Sec. II B.

A. Matrix product states in 1D

By introducing an order-3 tensor $T^{[n]} \in \mathbb{C}^{d \times \chi_{n-1} \times \chi_n}$ for each subsystem n and contracting the tensors in a chain, the wave function amplitudes in Eq. (1) can be written as an MPS

$$\Psi_{i_1, i_2, \dots, i_N} := \sum_{\alpha_1=1}^{\chi_1} \sum_{\alpha_2=1}^{\chi_2} \cdots \sum_{\alpha_{N-1}=1}^{\chi_{N-1}} T_{1, \alpha_1}^{[1], i_1} T_{\alpha_1, \alpha_2}^{[2], i_2} \cdots T_{\alpha_{N-1}, 1}^{[N], i_N}$$

with *bond dimensions* χ_n , where we have written the physical indices i_n as superscripts, such that the sums are performed over subscripts only. The indices α_n are sometimes referred to as *virtual indices*. Note that in this notation the bond dimensions at the two ends of the chain are $\chi_0 = \chi_N = 1$ and the tensors $T^{[1]}$ and $T^{[N]}$ can be interpreted as tensors of order 2.

Any quantum state can be written as an MPS, where the maximum bond dimension $\max(\{\chi_1, \dots, \chi_{N-1}\})$ is exponential in system size. If the bond dimensions are truncated to a finite value $\chi_n < \chi_{\max}$, the number of parameters of the MPS scales linearly in system size as $\mathcal{O}(N \chi_{\max}^2 d)$. Remarkably, all states with a 1D area law entanglement structure can be represented by an MPS with finite maximum bond dimension [10, 11].

An important property of MPS is the existence of an *isometric form* (also sometimes referred to as *canonical form*), which speeds up most MPS algorithms. To bring an MPS into isometric form, one must turn the tensors $T^{[n]}$ into *isometries*. An isometry is a matrix $W \in \mathbb{C}^{m \times k}$ with $m \geq k$ for which it holds

$$W^\dagger W = \mathbb{1}, \quad WW^\dagger = \mathbb{P}, \quad (2)$$

with $\mathbb{P} = \mathbb{P}^2$ a projector to the smaller vector space.

It is beneficial to introduce a diagrammatic notation, in which tensors are drawn as shapes and indices as lines

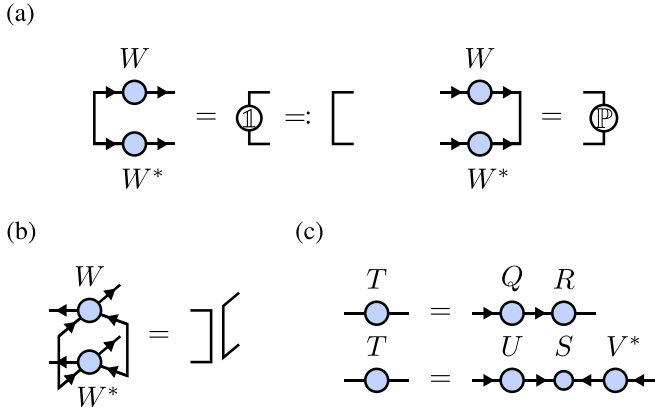


FIG. 1. Tensor network diagrams. (a) Isometry condition (2) for a tensor of order 2. (b) Isometry condition for a tensor of order 4. (c) QR decomposition and singular value decomposition.

emerging from these shapes. Examples of this diagrammatic notation are shown in Fig. 1. Tensor contractions are denoted by connecting the indices over which the contractions are performed. In tensor network diagrams, isometries can be denoted by decorating lines with arrows. In our convention incoming (outgoing) arrows correspond to the larger (smaller) dimension m (k). The definition of isometries can easily be extended to tensors: an isometric tensor is a tensor which becomes an isometry by grouping together all legs with incoming and outgoing arrows, respectively. The isometry condition (2) is drawn in diagrammatic form for a matrix and a tensor of higher order in Fig. 1(a) and Fig. 1(b), respectively.

Several decompositions can be used to decompose arbitrary matrices $A \in \mathbb{C}^{n \times m}$ into isometries. In this work, we use the QR decomposition and the singular value decomposition (SVD), which are depicted diagrammatically in Fig. 1(c).

In the isometric form of MPS, an arbitrary site n is chosen as the *orthogonality center*. All other tensors are isometrized in such a way that all arrows point towards the orthogonality center. As an example, we draw a five-site MPS in isometric form in Fig. 2(a). Any MPS can be exactly brought into isometric form using consecutive QR decompositions or SVDs. Further, the orthogonality center can be moved around easily and without error by contracting two neighboring tensors and splitting them again with a QR decomposition.

The isometric form improves the computational scaling and stability of most MPS algorithms. For example, consider the computation of the expectation value $\langle \Psi | \hat{O}_{ij} | \Psi \rangle$ of a two-site operator $\hat{O}_{ij}$ acting on neighboring sites i and j . This corresponds to the contraction of the tensor network that is created by sandwiching the operator between the bra and the ket. If the orthogonality center is at either site i or site j , the isometry condition can be used to reduce the problem to the contraction of only five tensors; see Fig. 2(b).

B. Isometric tensor network states in 2D

The natural generalization of MPS to 2D is given by PEPS. A PEPS represents the local subsystem on each lattice site j by the physical index i_j of a tensor $T^{[j],i_j}$ and nearest-

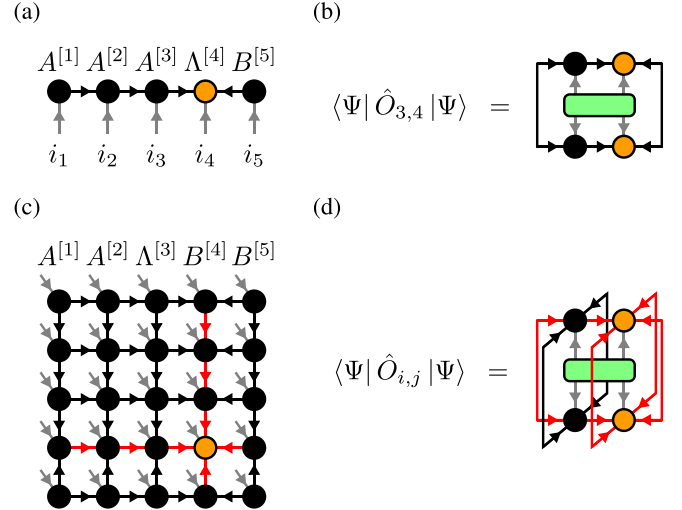


FIG. 2. Isometric tensor network states. (a) An MPS in isometric form representing a five-site chain is depicted as a tensor network diagram. (b) The expectation value of a local two-site operator reduces to a contraction of only five tensors due to the isometry condition (2). (c) isoTNS on a 2D square lattice as introduced in Ref. [24]. The orthogonality hypersurface is drawn in red; the orthogonality center is drawn in orange. (d) Similar to MPS, local two-site expectation values at the orthogonality center reduce to a contraction of only five tensors due to the isometry condition.

neighbor tensors are connected with virtual bonds according to the lattice structure. The maximum bond dimension of the virtual bonds is called D . The wave function amplitudes can be written as

$$\Psi_{i_1, i_2, \dots, i_N} = \mathcal{C}(T^{[1], i_1}, T^{[2], i_2}, \dots, T^{[N], i_N}),$$

where $\mathcal{C}(\dots)$ denotes the contraction of the full network over all virtual bonds. PEPS can efficiently represent states with an area-law entanglement structure in two and higher dimensions [11]. Remarkably, PEPS can even represent certain states with correlations decaying polynomially with separation distance [39], whereas MPS can only handle exponentially decaying correlations. Polynomially decaying correlations are characteristic for critical points. It is not generally possible to bring a PEPS into isometric form without error due to the presence of closed loops. Because of this, already the computation of local expectation values scales exponentially with system size and can only be computed approximately in practice, e.g., using the boundary MPS method [11] or corner transfer matrices [40]. Moreover, algorithms for ground state search and time evolution have computational costs scaling with high powers of the bond dimension. For example, the cost of a full TEBD update is dominated by the contraction of the effective environment, scaling as $\mathcal{O}(D^{10})$ [41].

isoTNS [24–26] generalize the isometric form of MPS to higher dimensions by enforcing isometry constraints. A 2D isoTNS on the square lattice as defined in Ref. [24] is constructed by enforcing the isometry conditions shown in Fig. 2(c). The isometries are chosen in such a way that all arrows point towards a special row and column, called the orthogonality hypersurface of the isoTNS. Along the orthogonality hypersurface all arrows point towards the orthogonality

center. The maximum bond dimension of bonds along the orthogonality hypersurface is increased to $\chi = f \cdot D$, where $f \geq 1$ is a positive integer. In practice, this can produce better results at similar computational costs compared to increasing the maximum bond dimension for all bonds of the lattice [24,27]. Because of the isometry condition, one can think of the contractions of each of the four regions outside the orthogonality hypersurface as orthogonal boundary maps [27]. Local expectation values of operators acting in the vicinity of the orthogonality center can be computed efficiently because most contractions reduce to identity, similar to the computation of local expectation values in MPS; see Fig. 2(d).

For implementing algorithms on isoTNS it is necessary to move the orthogonality center around the lattice. Moving the orthogonality center along the orthogonality hypersurface can be performed similar to MPS via QR decompositions. Moving the entire orthogonality hypersurface is a harder problem and can in general be only done approximately. In analogy to MPS, columns left of the orthogonality hypersurface are called $A^{[n]}$ and columns right of the orthogonality hypersurface are called $B^{[n]}$. Moving the orthogonality hypersurface $\Lambda^{[n]}$ one column to the right can be expressed as solving the problem

$$\Lambda^{[n]} B^{[n+1]} \approx A^{[n]} \Lambda^{[n+1]}, \quad (3)$$

where the notation $\Lambda^{[n]} B^{[n+1]}$ denotes the contraction of the columns $\Lambda^{[n]}$ and $B^{[n+1]}$ along their connecting bonds. Instead of Eq. (3) one can solve the simpler auxiliary problem

$$\Lambda^{[n]} = A^{[n]} \Lambda, \quad (4)$$

where Λ is a column of tensors without physical indices as shown in Fig. 3(a). In a second step this column can then be absorbed into $B^{[n+1]}$ via the standard algorithm of applying a matrix product operator (MPO) to an MPS and compressing the resulting MPS to the maximal bond dimension [10] to form the new orthogonality hypersurface $\Lambda^{[n+1]}$ [24]. One can variationally solve problem Eq. (4) by minimizing the distance $\|\Lambda^{[n]} - A^{[n]} \Lambda\|$, sweeping over all tensors of the columns $A^{[n]}$ and Λ while performing local optimizations, which is known as an Evenbly-Vidal style variational optimization [42,43]. It is however found in [24] that a single unzipping sweep, called the Moses Move (MM), provides a solution very close to the variational one while being far quicker. The MM can also be used as a good initialization for the variational algorithm. We sketch the MM in Fig. 3(b). Starting from the bottom of the orthogonality hypersurface column, the tensors are split one after the other using tripartite decompositions. A single tripartite decomposition of a tensor W is shown in Fig. 3(c). First, W is split into two tensors Q and θ via a truncated SVD $W = U(SV) = Q\theta$, where Q is an isometry. The bond connecting Q and θ is then reshaped into two bonds of bond dimension $\leq D$. Next, it is important to note that the full contraction is invariant under the insertion of a unitary U and its conjugate transpose, $Q\theta = (QU^\dagger)(U\theta)$, with $A = QU^\dagger$ still satisfying the isometry condition. This degree of freedom can be used to disentangle the tensor θ along the vertical direction. Accordingly, U is chosen such that the truncation error or some entanglement measure is minimized for splits of θ along this direction. Choosing a

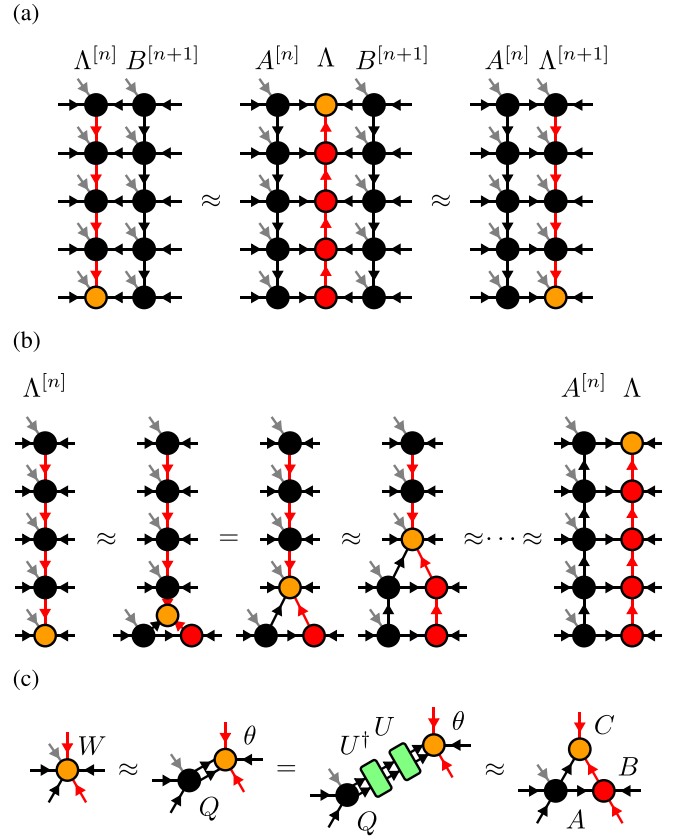


FIG. 3. MM algorithm. (a) The orthogonality hypersurface can be moved in two steps as explained in the text. (b) The MM can be used to iteratively split the column $\Lambda^{[n]}$ into two columns $A^{[n]}$ and Λ via repeated tripartite decompositions. (c) Tensor network diagram of a single tripartite decomposition as explained in the text.

good disentangling unitary is crucial for a successful tripartite decomposition. It is found in Ref. [27] that an Evenbly-Vidal style variational minimization of the Rényi-2 entropy followed by a Riemannian optimization of the Rényi-1/2 entropy gives the best results. It can be shown that the Rényi- α entropy in general bounds the truncation error for a fixed bond dimension if $\alpha < 1$ [13]. After contracting $QU^\dagger$ and $U\theta$, a truncated SVD is used to split $U\theta$ into tensors B and C as shown in the last step of Fig. 3(c), completing the tripartite decomposition. The computational cost of the MM scales with the maximum bond dimension D as $\mathcal{O}(D^7)$ [24,27].

With the MM it is straightforward to generalize certain MPS algorithms to isoTNS. For example, real and imaginary time evolution can be performed via the TEBD² algorithm, which is a nested loop of TEBD performed on the orthogonality hypersurface (which can be thought of as MPS with an enlarged physical dimension), where in one iteration of TEBD² time evolution gates are applied to all nearest-neighbor pairs of tensors. In comparison to PEPS, the cost of TEBD² is reduced from $\mathcal{O}(D^{10})$ to $\mathcal{O}(D^7)$ [24,27].

III. ALTERNATIVE ISOMETRIC FORM

In the following, we introduce an alternative way of defining the isometric form for isometric tensor network states,

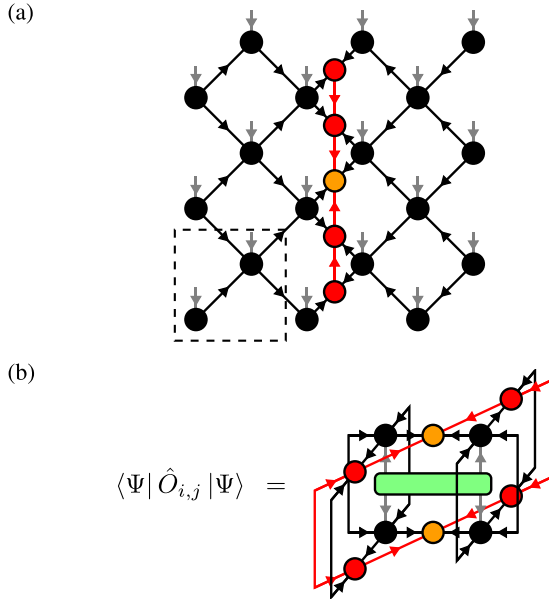


FIG. 4. (a) Tensor network diagram of an isoTNS in the alternative isometric form on a 2D square lattice. The orthogonality hypersurface is composed of auxiliary tensors with no physical degrees of freedom drawn in red. The orthogonality center is drawn in orange. The dashed lines denote a single unit cell of the lattice. (b) The computation of local expectation values at the orthogonality center reduces to a contraction of only a few tensors, similar to the original isoTNS [compare Fig. 2(d)].

which has some important differences to the isoTNS discussed in Sec. II B. We start by introducing the new isometric form in Sec. III A. To move the orthogonality hypersurface, a similar algorithm to the MM is necessary, which we call the Yang-Baxter (YB) move and discuss in Sec. III B. In Sec. III C we generalize the TEBD algorithm to this new isometric form.

A. Network structure

The structure of an isoTNS in the alternative isometric form on a square lattice is shown in Fig. 4(a). It can be constructed in three steps. First, a square PEPS is rotated by 45° . Next, the orthogonality hypersurface is constructed as a column of auxiliary tensors. The auxiliary tensors are connected in a line similar to an MPS and placed between two columns of PEPS tensors. Note that, in contrast to the original isoTNS, the tensors of the orthogonality hypersurface do not carry any physical degrees of freedom and only have virtual indices. Lastly, the isometry condition is enforced such that all arrows point towards the orthogonality hypersurface. Tensors left of the orthogonality hypersurface are thus brought into a left-isometric form and tensors right of the orthogonality hypersurface are brought into a right-isometric form, as shown in Fig. 4(a). The auxiliary tensors making up the orthogonality hypersurface are isometrized such that all arrows point towards a single auxiliary tensor—the orthogonality center.

In the following, we will denote the auxiliary tensors by W_j and the tensors carrying physical degrees of freedom by T_j . The bonds connecting two T tensors or a T tensor and a W tensor are truncated to a maximal bond dimension of D ,

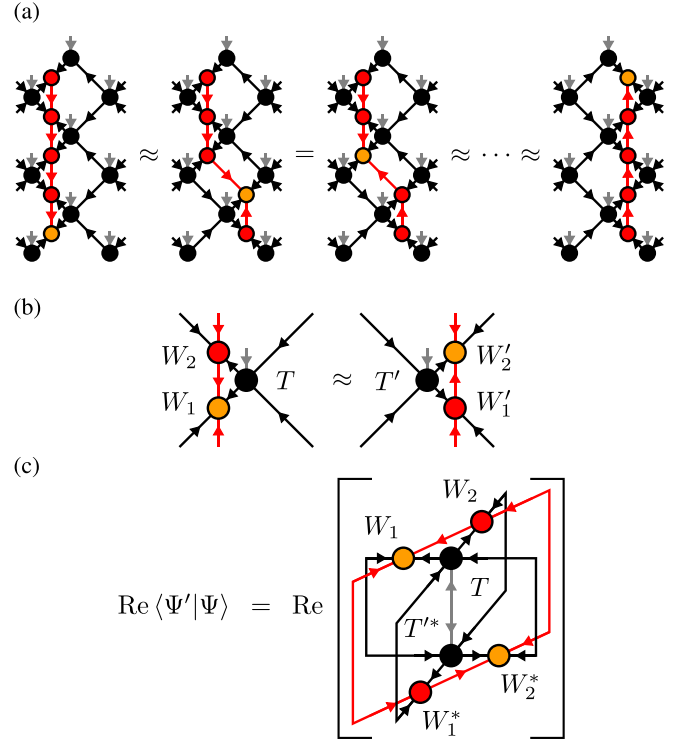


FIG. 5. (a) In the alternative isometric form, the orthogonality hypersurface can be moved one column to the right by iteratively applying YB moves. (b) Tensor network diagram of a single YB move. (c) The isometry condition reduces the overlap of the state before and after a single YB move to a contraction of six tensors.

while the maximal bond dimension between two W tensors is denoted as χ . Similar to the original isoTNS it is found that setting $\chi = f \cdot D$ with an integer $f \geq 1$ produces good results in practice. The alternative isometric form again allows for the fast computation of expectation values of local operators; see Fig. 4(b).

B. Yang-Baxter move

As with the original isoTNS, the orthogonality center can be easily and exactly moved along the orthogonality hypersurface using QR decompositions, but shifting the orthogonality hypersurface to the left and right is a harder problem. In analogy to the MM we look for a procedure to iteratively shift the orthogonality hypersurface through one column of T tensors as shown in Fig. 5(a). A single iteration of this process is shown in Fig. 5(b). The two tensors W_1 and W_2 , which are part of the orthogonality hypersurface, are “pulled through” the tensor T , resulting in updated tensors T' , W_1' , and W_2' . To keep the isometric structure of the network, T' and W_1' must be isometries and (if the wave function is normalized) W_2' must be a tensor of norm one. Due to the visual similarity to the Yang-Baxter equation we call this procedure the Yang-Baxter move. Accordingly, to distinguish the alternative isometric form from the original isoTNS, we will in the following refer to the original isoTNS as MM-isoTNS, whereas we will call isoTNS in the new isometric form YB-isoTNS.

We denote the state represented by the YB-isoTNS before the YB move by $|\Psi\rangle = |\Psi(W_1, W_2, T)\rangle$ and the state after the

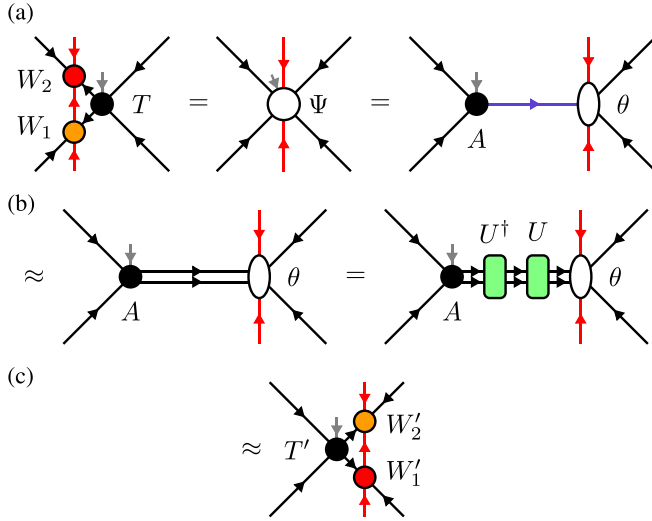


FIG. 6. Tensor network diagram of the tripartite decomposition algorithm that can be used for performing the YB move. The algorithm is explained in detail in the text.

YB move by $|\Psi'\rangle = |\Psi'(W'_1, W'_2, T')\rangle$. One can think of the YB move as an optimization problem

$$(T'_{\text{opt}}, W'_{1,\text{opt}}, W'_{2,\text{opt}}) = \underset{T', W'_1, W'_2}{\operatorname{argmin}} \|\Psi\rangle - |\Psi'\rangle\|$$

under the constraints $T'^{\dagger}T' = \mathbb{1}$, $W_1'^{\dagger}W'_1 = \mathbb{1}$, and $\|W'_2\|_F = 1$ with the Frobenius norm $\|\cdot\|_F$. The error of the YB move can be rewritten as

$$\begin{aligned} \|\Psi\rangle - |\Psi'\rangle\| &= \sqrt{\langle\Psi|\Psi\rangle + \langle\Psi'|\Psi'\rangle - 2\operatorname{Re}\langle\Psi'|\Psi\rangle} \\ &= \sqrt{2 - 2\operatorname{Re}\langle\Psi'|\Psi\rangle}, \end{aligned}$$

where in the second step we used the fact that the wave function is normalized to one: $\langle\Psi|\Psi\rangle = \langle\Psi'|\Psi'\rangle = 1$. It follows that the optimization problem of minimizing the error becomes the problem of maximizing the overlap

$$(T'_{\text{opt}}, W'_{1,\text{opt}}, W'_{2,\text{opt}}) = \underset{T', W'_1, W'_2}{\operatorname{argmax}} \operatorname{Re}\langle\Psi'|\Psi\rangle \quad (5)$$

under the previously defined constraints. Because the only tensors that are changed by the YB move are W_1 , W_2 , and T and the three tensors make up a subregion of the full network with only incoming arrows, we can use the isometry condition to reduce the computation of the overlap $\langle\Psi'|\Psi\rangle$ to a contraction of only six tensors as shown in Fig. 5(c). To solve the optimization problem Eq. (5), we implemented two explicit algorithms. The first algorithm is an Evenbly-Vidal style variational optimization method with iterative local updates. The second algorithm is a tripartite decomposition algorithm with disentangling, similar to the tripartite decomposition used in the MM. In practice we find that the latter performs much better. In the following we will thus only discuss the tripartite decomposition YB move; for more details on the first algorithm see the Appendix.

The tripartite decomposition YB move algorithm is similar to the tripartite decomposition used in the MM and is sketched in Fig. 6. We start by contracting the tensors T , W_1 , and W_2 into a single tensor Ψ . This tensor is then split from left to

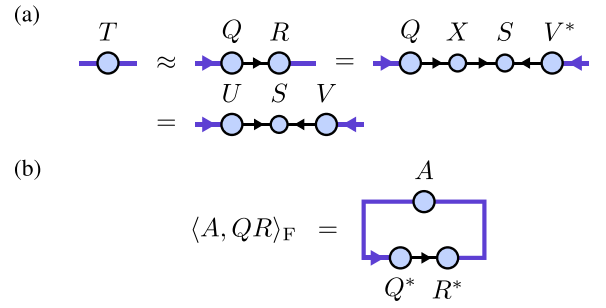


FIG. 7. (a) Approximate truncated SVD based on Ref. [44]. Bonds of bond dimension χ_1 and χ_2 are drawn in purple; bonds of bond dimension χ_{trunc} are drawn in black. (b) Tensor network diagram of the overlap $\langle A, QR \rangle_F$ that can be iteratively maximized to compute the approximate QR decomposition which is the first step of the approximate truncated SVD.

right via a truncated SVD $\Psi = ASV^{\dagger} = A(SV^{\dagger}) =: A\theta$; see Fig. 6(a). Here, the bond connecting A and θ has a bond dimension of $\min(dD^2, \chi^2 D^2)$. The bond dimension is truncated to D^2 . Next, we split the index of the bond connecting A and θ into two indices of dimension D each. There exists a degree of freedom on the bonds connecting A and θ : a unitary U and its adjoint can be inserted without changing the result of the contraction $A\theta = AU^{\dagger}U\theta = (AU^{\dagger})(U\theta) =: T'\theta$; see Fig. 6(b). The unitary U can be chosen to minimize the truncation error of the next step by disentangling the tensor θ . Similarly to MM-isoTNS [24,27] we find that a cheap Rényi-2 disentangling using an Evenbly-Vidal style variational algorithm followed by a Rényi-0.5 disentangling using Riemannian optimization gives the best results in practice. For more details on Evenbly-Vidal style and Riemannian optimization of isometric tensor networks see the Appendix. In the last step, Fig. 6(c), the tensor θ is vertically split into W'_1 and W'_2 using a truncated SVD. Here, the bond dimension is truncated to χ . We end up with the three tensors T' , W'_1 , and W'_2 completing the YB move.

We note that, because of the 45° rotated network structure, the physical tensors involved in shifting the orthogonality surface by one site are not directly connected with each other in the tensor network; see Fig. 5(a). Because of this, one can think of the YB move as an entirely local operation, in contrast to the inherently iterative nature of the unzipping MM. A similar construction to the YB-isoTNS could also be done by inserting the orthogonality hypersurface as a column of auxiliary tensors without performing a 45° rotation of the square lattice. However, as long as the orthogonality hypersurface is not diagonal with respect to the lattice structure, the shift can only be performed iteratively and this locality is lost.

The computational complexity of the YB move using the tripartite decomposition scales as $\mathcal{O}(D^9)$. However, this can be brought down to $\mathcal{O}(D^8)$ by using an approximation of the truncated SVD based on Ref. [44]. Assume we would like to perform the truncated SVD $A \approx USV$ with $A \in \mathbb{C}^{\chi_1 \times \chi_2}$, truncating to a bond dimension $\chi_{\text{trunc}} < \chi_1, \chi_2$. The truncated SVD can be computed by performing a full SVD and keeping only the rows and columns corresponding to the χ_{trunc} largest singular values, scaling as $\mathcal{O}(\chi_1 \chi_2 \min(\chi_1, \chi_2))$. Alterna-

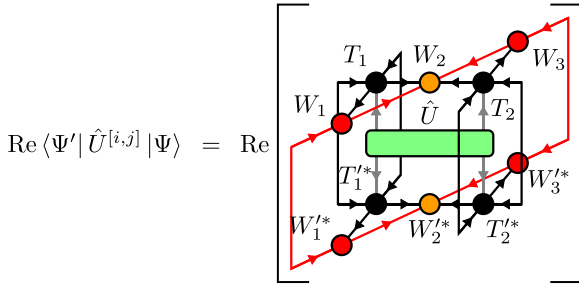


FIG. 8. Overlap of the state before and after applying a local TEBD gate $\hat{U}^{[i,j]}$ is reduced to the contraction of only a few tensors due to the isometry condition.

tively, an approximate iterative procedure can be used, which we sketch in Fig. 7(a). First, an approximate QR decomposition $A = QR$ with $Q \in \mathbb{C}^{\chi_1 \times \chi_{\text{trunc}}}$, $R \in \mathbb{C}^{\chi_{\text{trunc}} \times \chi_2}$ is performed. For this, we first initialize the matrices Q and R , e.g., using random values. We then iteratively maximize the overlap $\langle A, QR \rangle_F$ [see Fig. 7(b)], alternately optimizing the tensors Q and R at a computational complexity of $\mathcal{O}(\chi_1 \chi_2 \chi_{\text{trunc}})$ per iteration. After convergence, a SVD $R = XSV$ is performed at a cost of $\mathcal{O}(\chi_2 \chi_{\text{trunc}}^2)$. Setting $U = QX$ concludes the approximate SVD at an overall cost of $\mathcal{O}(N_{\text{QR}} \chi_1 \chi_2 \chi_{\text{trunc}})$, where N_{QR} denotes the maximum number of iterations for the approximate QR decomposition.

The approximate truncated SVD decomposition can also be used to approximate gradients used in the Riemannian optimization during the disentangling process, again bringing down the computational complexity from $\mathcal{O}(D^9)$ to $\mathcal{O}(D^8)$. This is particularly useful here because during the iterative Riemannian optimization algorithm, the current iterate for the disentanglement unitary changes only very little each iteration. This means that the result of the approximate QR decomposition of the previous iteration can be used as initialization for the current iteration, leading to a very fast convergence (in practice $N_{\text{QR}} \leq 5$ iterations) of the approximate QR decomposition.

C. TEBD²

We will now discuss the generalization of the time evolving block decimation (TEBD) algorithm [38] to YB-isoTNS. The TEBD algorithm can be used for both real and imaginary time evolution. Analogously to MPS we start with a Suzuki-Trotter decomposition, approximating the time evolution operator $\hat{U}(\Delta t) = e^{-i\Delta t \hat{H}}$ by a product of bond operators

$\hat{U}^{[i,j]}(\Delta t) = e^{-i\hat{h}^{[i,j]}\Delta t}$ acting only on neighboring sites i and j of the bond $[i, j]$. These bond operators then must be applied to the state in the correct order, while keeping the YB-isoTNS structure intact. A full update is performed by applying all bond operators, which approximately evolves the state by time Δt . We will first discuss the process of applying a single bond operator $\hat{U}^{[i,j]}(\Delta t)$ to the YB-isoTNS. Let us assume that the orthogonality center is positioned between the two sites on which the bond operator $\hat{U}^{[i,j]}(\Delta t)$ acts. The five tensors around the orthogonality center then make up a subnetwork with only incoming arrows; see Fig. 8. We call these five tensors T_i, T_j, W_1, W_2 , and W_3 . The local TEBD update can then be formulated as the following problem: find tensors T'_i, T'_j, W'_1, W'_2 , and W'_3 satisfying the isometry constraints and minimizing the error

$$\varepsilon_{\text{trunc}} = \|\hat{U}^{[i,j]}(\Delta t) |\Psi\rangle - |\Psi'\rangle\|.$$

Similar to the YB move, we can rewrite this as the problem of maximizing the overlap

$$\begin{aligned} & (T'_{i,\text{opt}}, T'_{j,\text{opt}}, W'_{1,\text{opt}}, W'_{2,\text{opt}}, W'_{3,\text{opt}}) \\ &= \underset{T'_i, T'_j, W'_1, W'_2, W'_3}{\text{argmax}} \text{Re} \langle \Psi' | \hat{U}^{[i,j]}(\Delta t) | \Psi \rangle \end{aligned} \quad (6)$$

under the constraints $T_i'^{\dagger} T'_i = \mathbb{1}$, $T_j'^{\dagger} T'_j = \mathbb{1}$, $W_1'^{\dagger} W'_1 = \mathbb{1}$, $W_3'^{\dagger} W'_3 = \mathbb{1}$, and $\|W_2'\|_F = 1$. Using the isometry condition, the overlap $\langle \Psi' | \hat{U}^{[i,j]}(\Delta t) | \Psi \rangle$ can be computed by contracting the tensor network drawn in Fig. 8. For solving the constrained optimization problem Eq. (6) we use an Evenbly-Vidal style iterative optimization algorithm. We optimize one tensor at a time while keeping all other tensors fixed. This procedure is repeated, sweeping over all five tensors until convergence is achieved. For more details on this optimization method, see Appendix. Since the time step Δt is chosen to be small, the bond operator is close to identity, $\hat{U}^{[i,j]}(\Delta t) \approx \mathbb{1}$. Thus a good initialization for the tensors of the updated wave function $|\Psi'\rangle$ are simply the tensors of the old wave function $|\Psi\rangle$. The computational complexity of applying a local bond operator to a YB-isoTNS with the discussed algorithm scales as $\mathcal{O}(D^6)$. In practice it is observed that the algorithm often converges after only a few iterations.

To apply a global TEBD update, we must apply local TEBD updates at all bonds. Because each YB move introduces an error ε_{YB} , it is beneficial to minimize the number of necessary YB moves. By moving the orthogonality hypersurface once from left to right and back it is possible to apply

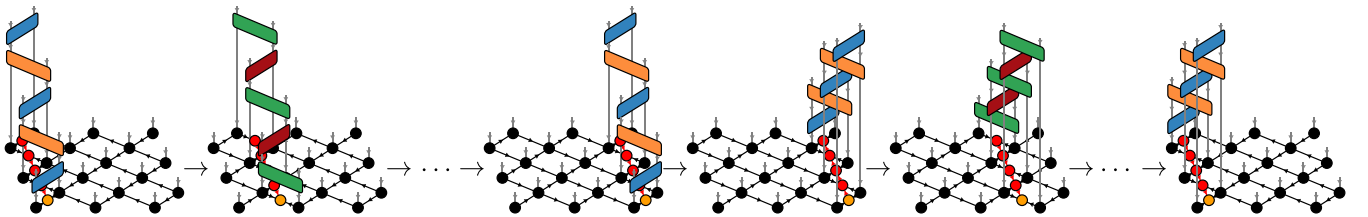


FIG. 9. Full second order TEBD update of time step Δt is performed by applying nearest-neighbor time evolution gates $\hat{U}^{[i,j]}(\Delta t/2)$ along the orthogonality hypersurface, moving from the leftmost to the rightmost position and back. When moving back, the gates are applied in reverse order.

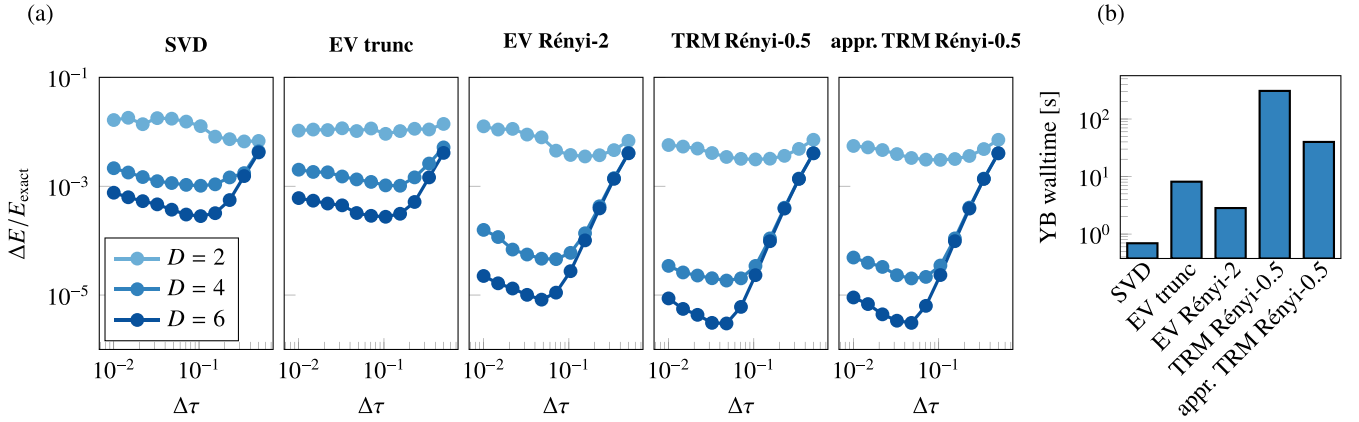


FIG. 10. (a) In this figure we perform ground state search of the TFI model on a 4×4 diagonal square lattice (32 spins) at a transverse field of $g = 3.5$ using the imaginary time TEBD² YB-isoTNS algorithm. The relative energy error is plotted against the time step size $\Delta\tau$. The five plots correspond to five different algorithms used for performing the YB move; details are given in the text. For all iterative algorithms, we used a maximum of $N_{\text{iter,YB}} = 100$ iterations. For all plots, the bond dimension along the orthogonality surface was set to $\chi = 6 \cdot D$ and the number of iterations for local TEBD updates was set to $N_{\text{iter,TEBD}} = 100$. (b) We compare the duration that the different algorithms spend on the YB move during a full TEBD update of time $\Delta\tau$. We compute this accumulated walltime by running TEBD for $N_{\text{TEBD}} = 10$ steps, measuring the time spent performing YB moves, and computing the average walltime per TEBD step.

a second order global TEBD update with a Trotterization error of $\mathcal{O}(\Delta t^3)$; see Fig. 9.

IV. BENCHMARKING

To benchmark the method, we consider the transverse field Ising (TFI) model

$$\hat{H}_{\text{TFI}} = -J \sum_{\langle i,j \rangle} \hat{\sigma}_i^x \hat{\sigma}_j^x - g \sum_i \hat{\sigma}_i^z, \quad (7)$$

where $\langle i, j \rangle$ denotes pairs of nearest-neighbor spin-1/2 particles and $\hat{\sigma}_i^x, \hat{\sigma}_i^z$ are Pauli matrices. On the square lattice the model exhibits a quantum phase transition at a critical transverse field of $g_c \approx 3.04438$ [45].

We first compute the ground state of the TFI model Eq. (7) using YB-isoTNS and TEBD with imaginary time $\Delta t = -i\Delta\tau$, $\Delta\tau \in \mathbb{R}$. As an initial state, we choose the product state $|\Psi_i\rangle = |\uparrow\rangle \otimes \cdots \otimes |\uparrow\rangle$ of all spins pointing up. To use the TEBD algorithm, one must choose a good step size $\Delta\tau$. The total error of a single TEBD step is a sum of three errors,

$$\varepsilon_{\text{TEBD}} = \varepsilon_{\text{Trotter}} + \varepsilon_{\text{trunc}} + \varepsilon_{\text{YB}}.$$

The Trotterization error $\varepsilon_{\text{Trotter}}$ comes from the Suzuki-Trotter decomposition, the truncation error $\varepsilon_{\text{trunc}}$ comes from the local application of the TEBD bond operators, and the YB error ε_{YB} arises when shifting the orthogonality hypersurface. A smaller time step Δt decreases both $\varepsilon_{\text{Trotter}}$ and $\varepsilon_{\text{trunc}}$, while the YB error ε_{YB} is not directly affected. However, since for a smaller time step Δt more TEBD iterations are necessary to approach the limit $t \rightarrow \infty$, YB errors add up and prevent the state from reaching the true ground state. Therefore, we expect, similar to [24,27], that the YB error dominates for small time steps, while the Trotterization and truncation errors dominate for larger time steps. The best results can be achieved when the time step Δt is tuned such that $\varepsilon_{\text{Trotter}} + \varepsilon_{\text{trunc}} \approx \varepsilon_{\text{YB}}$.

In Fig. 10(a) we plot the obtained relative ground state energy error against the time step $\Delta\tau$ on a 4×4 diagonal square lattice at a transverse field of $g = 3.5$. The numerically exact reference energy was computed using an MPS-DMRG simulation using tenpy [46], snaking an MPS through the 2D lattice. We compare the following five different algorithms for performing the YB move.

- (1) *SVD*. Tripartite decomposition via two consecutive SVDs without disentangling.
- (2) *EV trunc*. Direct Evenly-Vidal style optimization of the overlap Eq. (5) (directly minimizing the error ε_{YB}).
- (3) *EV Rényi-2*. Tripartite decomposition with Evenly-Vidal style Rényi-2 disentangling.
- (4) *TRM Rényi-0.5*. Same as *EV Rényi-2*, but with additional disentangling using the Riemannian Trust Region Method, a quasi-Newton optimization method (see Appendix 3 for more details).
- (5) *appr. TRM Rényi-0.5*. Same as *TRM Rényi-0.5*, but using approximate truncated SVD for computing the gradients.

In Fig. 10(b) we additionally plot the YB move walltimes of the different algorithms. We observe that disentangling plays a crucial role in minimizing the error of the YB move. (4) *TRM Rényi-0.5* gives the best results in practice, but has a large computational cost. Using the approximate gradient computation gives essentially the same results, but speeds up the algorithm by almost an order of magnitude. In all following benchmarks we will thus use (5) *appr. TRM Rényi-0.5* for performing the YB move.

In Fig. 11(a) we plot the best obtained ground state energy density against the linear system size L . Since the diagonal square lattice has a unit cell containing two sites, the number of spins is given by $N = 2L^2$. In the limit $L \rightarrow \infty$ we expect the energy density E/N to approach a constant, whereas the energy density of small systems is dominated by finite size effects. We observe that the energy density computed using the reference MPS-DMRG simulation first decreases but then increases again when going to larger system sizes,

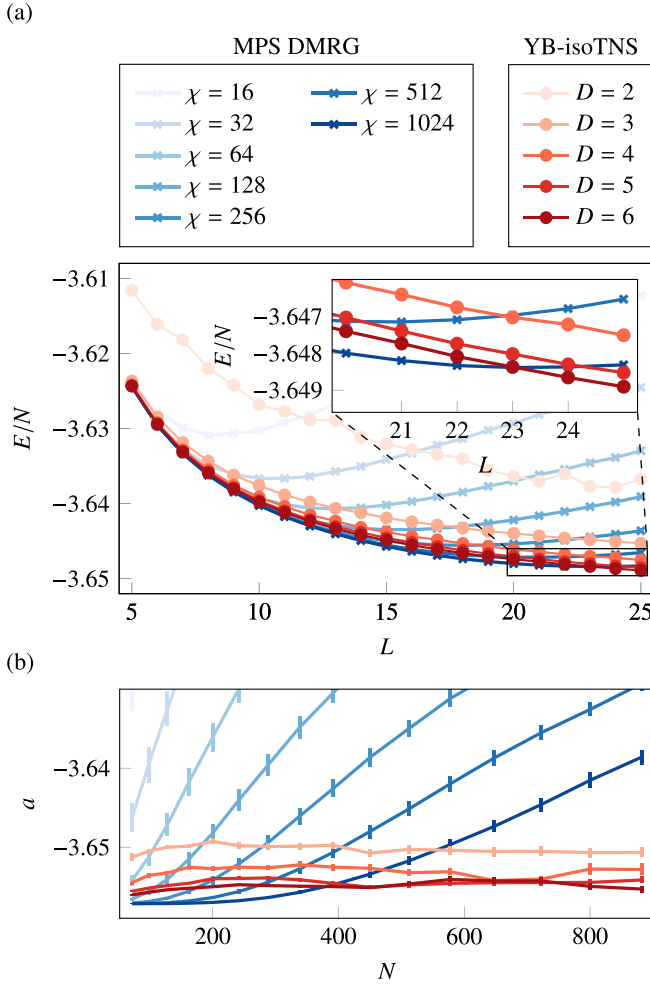


FIG. 11. (a) In this figure we plot the lowest energy densities found using imaginary time TEBD² on the TFI model at $g = 3.5$ [corresponding to the minima in Fig. 10(a)] against linear system size L . The algorithm parameters are the same as in Fig. 10(a), except that we here use only $N_{\text{iters,YB}} = 10$ iterations for the approximate TRM disentangling during the YB move. The reference simulation shown in blue was performed using MPS DMRG. We note that for large systems $L \geq 20$ the reference MPS simulations with $\chi \geq 256$ are only converged up to a ground state energy density error of $\Delta(E/N) \leq 10^{-4}$ due to large computation times and memory costs, which does however not change the results in a qualitative way. (b) Fit parameter a obtained by fitting the function $E = a \cdot N + b \cdot N_{\text{boundary}} + c$ to the data from (a). We used a rolling window of nine data points around each N for the fit. The $D = 2$ data was excluded from this plot due to noise. The error bars are the square root of the variance.

which happens earlier for smaller bond dimensions. The reason for this is that the inherently 1D MPS fails to capture the 2D area law entanglement structure of the ground state. More specifically, certain pairs of sites that are close together in the 2D lattice are far apart in the MPS that is snaked through the lattice, resulting in long-range interactions. In contrast, the energy density computed using YB-isoTNS does not show this effect and approaches a constant energy density already for $D = 2$. We interpret this as the YB-isoTNS being able to correctly capture the 2D area law entanglement structure. For large systems of size $L = 24$ and $L = 25$, YB-

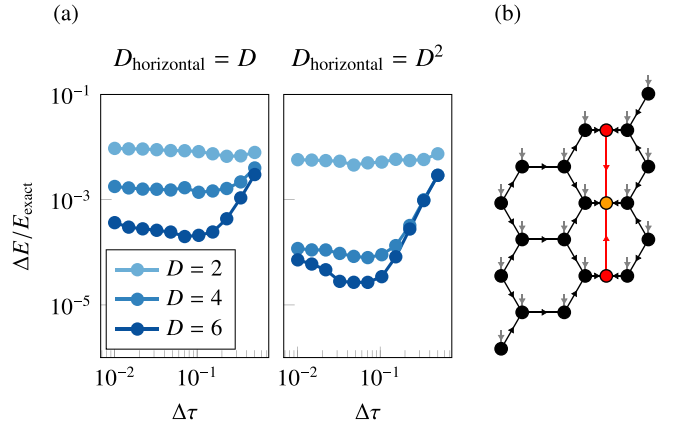


FIG. 12. (a) Imaginary time TEBD results using YB-isoTNS on the honeycomb lattice. The simulation for the left and right plot was computed using $D_{\text{horizontal}} = D$ and $D_{\text{horizontal}} = D^2$, respectively. The bond dimension along the orthogonality hypersurface was chosen as $\chi = 6 \cdot D$. For the YB move we used the approximate Rényi-0.5 disentangler with a maximum of $N_{\text{iter,YB}} = 100$ iterations per YB move. The model is the TFI model on a 4×4 honeycomb lattice at a transverse field of $g = 3.5$. (b) YB-isoTNS network structure on a 3×3 honeycomb lattice.

isoTNS with $D = 5$ and $D = 6$ is able to find lower ground state energies than MPS DMRG with $\chi = 1024$, which can be seen in the inset in Fig. 11(a). This is possible even though the isoTNS has much less variational parameters: for $L = 25$, an MPS with a maximum bond dimension of $\chi = 1024$ needs to store $N_{\text{params,MPS}} \approx 2.6 \times 10^9$ complex numbers, while a YB-isoTNS with $D = 5$, $\chi = 30$ only consists of $N_{\text{params,YB-isoTNS}} \approx 2.7 \times 10^6$ complex numbers, which is a difference of three orders of magnitude. Similar to PEPS we expect the advantage of isoTNS to become even more apparent for even larger systems or closer to the critical point, where capturing the area law entanglement structure is more important.

We now perform a more careful analysis of the data from Fig. 11(a). The ground state energy of a gapped model with open boundary conditions is expected to scale as

$$E = a \cdot N + b \cdot N_{\text{boundary}} + c + \dots, \quad (8)$$

where $N_{\text{boundary}} \propto L$ and the remaining corrections are exponentially small in system size. In Fig. 11(b) we plot the fit parameter a obtained by fitting Eq. (8) over a rolling window around a given N . We observe that the MPS data shows a plateau for small system sizes $N \lesssim 200$ and large bond dimensions $\chi \geq 512$, where the simulation is numerically exact. The parameter a diverges for larger system sizes. In contrast, the isoTNS data plateaus for large system sizes even for small bond dimensions D . However, the numerically exact value for a that can be extracted from the MPS data at $N \lesssim 200$ is not reached even at $D = 6$. We conclude that, for small system sizes, MPS performs better due to the fact that one can choose much larger bond dimensions compared to isoTNS. However, MPS fails to describe the model for large system sizes while the isoTNS simulation correctly captures the area law entanglement structure. We emphasize that the strong performance of MPS in Fig. 11(a) relies on the fact that the correlation

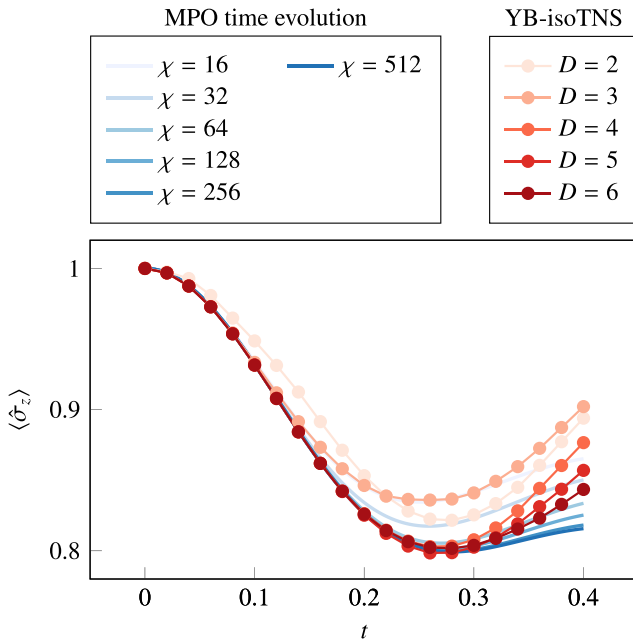


FIG. 13. In this figure we show the time evolution of the $\langle \hat{\sigma}_z \rangle$ expectation value of a spin in the middle of an 8×8 diagonal square lattice (at position $x = y = 4$, $m = 0$, where m is the index into the unit cell), containing in total $N = 128$ spins. We use the TFI model at critical field g_c . We compute the time evolution once with the MPO time evolution algorithm [47] on a MPS and once with TEBD on a YB-isoTNS. We observe good agreement up to times of $t \approx 0.25$, where the two simulations diverge. For the isoTNS simulation we use a time step of $\Delta t = 0.02$, $N_{\text{iter,TEBD}} = 100$ iterations for local TEBD updates, and $N_{\text{iter,YB}} = 100$ iterations of the (v) *appr. TRM Rényi-0.5* disentangler for the YB move. The bond dimension along the orthogonality hypersurface is set to $\chi = 6 \cdot D$. For the MPS reference simulation we use the second order MPO time evolution algorithm with a time step of $\Delta t = 0.001$.

length of the TFI model decreases quickly when moving away from the critical point. We expect the advantages of isoTNS to become more evident closer to criticality or for models that exhibit higher entanglement. To show that the algorithm can be generalized to different lattice structures we perform a ground state search of the TFI model on a honeycomb lattice in Fig. 12(a). The resulting isoTNS has two different kinds of virtual bonds, horizontal and diagonal bonds, and is sketched in Fig. 12(b). Since a single horizontal bond splits into two diagonal bonds it is intuitive to choose a larger bond dimension at the horizontal bonds, e.g., $D_{\text{horizontal}} = D^2$.

We lastly study the capabilities of YB-isoTNS to perform real-time evolution. For this we perform a global quench by initializing the YB-isoTNS in the all-up state $|\Psi\rangle = |\uparrow\rangle \otimes \cdots \otimes |\uparrow\rangle$, which we then evolve in time using TEBD. We work on the 8×8 diagonal square lattice and evolve with the Hamiltonian of the TFI model at critical transverse field $g \approx 3.04438$. At the critical field the entanglement grows very quickly, making this a hard problem. We choose a step size of $\Delta t = 0.02$ and evolve up to time $t = 0.4$, requiring 20 TEBD iterations. For a comparison we use the second order MPO time evolution algorithm [47] defined on MPS, which is able to perform time evolution in

the presence of long range interactions and is implemented in tenpy [46]. Due to the long-range couplings present in the effective Hamiltonian for the MPS a very small time step is necessary, leading to long computation times. We choose a time step of $\Delta t = 0.001$, which is computationally feasible but still not fully converged. We plot the expectation value $\langle \hat{\sigma}_z \rangle$ of a spin in the middle of the lattice against the time in Fig. 13. We observe that the YB-isoTNS results are in good agreement with the reference simulation up to a time of $t \approx 0.25$, at which the two simulations diverge. This happens at earlier times for smaller bond dimensions D . We expect the reason for this to be the fast accumulation of errors during YB moves. We conclude that the short-time behavior of the system is correctly captured by the YB-isoTNS.

V. CONCLUSION

In this work we introduced and benchmarked a new *Ansatz* for defining the isometric form of isometric tensor network states in 2D. In the new *Ansatz* the orthogonality hypersurface is made up of auxiliary tensors without physical degrees of freedom. The YB move was introduced as an algorithm for moving the orthogonality hypersurface. In comparison to the MM used in previous works on isoTNS, the YB move does not need an additional step for MPO-MPS multiplication and compression. It was found, similar to the MM [24,27], that it is crucial to perform a disentangling step during the YB move. The best results were achieved with a Riemannian optimization of the Rényi entropy. To speed up the YB move we implemented an approximate version of this algorithm, bringing down the computational complexity from $\mathcal{O}(D^9)$ to $\mathcal{O}(D^8)$. We finally implemented a TEBD² algorithm that can be used to perform real and imaginary time evolution on YB-isoTNS.

The algorithm was then benchmarked by performing ground state search and a global quench of the transverse field Ising model. It was found that YB-isoTNS are able to produce similar results to MM-isoTNS. The algorithms performing best in the ground state search were found to be the Riemannian optimization algorithms minimizing the Rényi entropy with $\alpha = 1/2$. In a large scale simulation of up to $N = 1250$ spins, numerical evidence was found that YB-isoTNS are able to correctly capture the area law entanglement structure of the ground state. For the largest systems we looked at, YB-isoTNS was able to achieve lower ground state energies while needing to store $\approx 10^3$ times less variational parameters compared to MPS. Additionally, the *Ansatz* was used to compute the ground state energy of the TFI model on a honeycomb lattice, showcasing that YB-isoTNS can be easily generalized to other lattice types. Computing a real time evolution after a global quench was found to be challenging for YB-isoTNS. The reason for this is the fast accumulation of YB move errors. Nevertheless, YB-isoTNS were found to be able to correctly capture short-time behavior of the 2D TFI model with $N = 128$ spins even at criticality. Additionally, a much larger time step Δt can be used compared to the MPO time evolution algorithm on MPS, leading to shorter computation times.

As for the performance, the YB-isoTNS *Ansatz* is able to produce comparable results to MM-isoTNS. While the computational complexity of shifting the orthogonality hypersurface of YB-isoTNS is larger [$\mathcal{O}(D^8)$ compared to $\mathcal{O}(D^7)$] it is still an interesting alternative way of defining the isometry condition. One advantage of YB-isoTNS is the ease with which the *Ansatz* can be generalized to different lattices. For instance, we expect the *Ansatz* to be extendable to the kagome lattice without increasing the cost scaling of the YB move. Moreover, more involved algorithms like DMRG require the contraction of a boundary MPS, which has a computational complexity scaling as $\mathcal{O}(D^{10})$ to $\mathcal{O}(D^{12})$ depending on the implementation. For such algorithms, we expect the increased cost for the YB move compared to the MM to not make a large difference. Furthermore, the algorithm for shifting the orthogonality hypersurface could be adapted to perform all YB moves in parallel by utilizing the canonical form of MPS introduced by Vidal [48]. This is akin to the parallel implementation of TEBD on MPS [49–51]. We anticipate that such a parallelized version would yield results equivalent to those of the sequential algorithm in the regime of low truncation error, while offering significant computational speedups for large system sizes. In contrast, a similar parallel strategy is not feasible for MM-isoTNS, highlighting a fundamental conceptual distinction between the two *Ansätze*: the YB move can be understood as entirely local, whereas the unzipping sweep in MM-isoTNS is a more global operation. Consequently, we anticipate that extending YB-isoTNS to infinite strip geometries, as in Ref. [31], should be straightforward.

It would be interesting to implement a DMRG algorithm on YB-isoTNS and compare its performance to TEBD. Future research could explore alternative algorithms for performing the YB move, which we find to be the main source of errors. Similar to MM-isoTNS, implementing a variational optimization on the complete column when shifting the orthogonality surface may be beneficial, as this was found to be essential for achieving good results when performing real time evolution with MM-isoTNS [27]. The algorithm could also significantly benefit from an implementation on Graphics Processing Units (GPUs), which can greatly accelerate tensor contractions and decompositions while also increasing the power efficiency. Additional promising research directions include the extension of the method to the thermodynamic limit and further investigation of the connection between isoTNS and quantum computing [29,36]. Due to the direct correspondence between sequential quantum circuits and isoTNS [52] one can use the *Ansatz* to classically optimize sequential circuits. Furthermore, it would be interesting to study the limitations of isoTNS compared to unrestrained PEPS more systematically, for example using fermionic Gaussian states [37].

ACKNOWLEDGMENTS

The authors thank S.-H. Lin, S. Anand, and J. Hauschild for helpful discussions. This work was supported by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy EXC-2111-390814868, TRR360-492547816, and the Munich Quantum Valley, which is supported by the Bavarian state government with funds from the Hightech Agenda Bayern Plus. M.P.Z.

was supported by the U.S. Department of Energy, Office of Science, Basic Energy Sciences under Early Career Award No. DE-SC0022716.

DATA AVAILABILITY

Data analysis and simulation codes are available on Zenodo [53]. The YB-isoTNS source code is openly available on github [54].

APPENDIX: OPTIMIZATION OF ISOMETRIC TENSOR NETWORKS

When working with isometric tensor networks, one often needs to find optimal tensors that extremize a given cost function f . In the most general case, f is a function

$$f : \mathbb{C}^{m_1 \times n_1} \times \mathbb{C}^{m_2 \times n_2} \times \dots \times \mathbb{C}^{m_K \times n_K} \rightarrow \mathbb{R},$$

mapping K tensors $T_1, \dots, T_K$ to a scalar cost value. Here, the tensors have already been reshaped into matrices, grouping together legs with incoming arrows and legs with outgoing arrows, respectively. The tensors must satisfy certain constraints. If a tensor T_i possesses both legs with incoming arrows and legs with outgoing arrows, it must satisfy the isometry constraint $T_i^\dagger T_i = \mathbb{1}$. If instead the tensor T_j possesses only legs with incoming arrows (and thus is the orthogonality center), it is constrained to be normalized to one: $\|T_j\|_F = 1$. To summarize, we want to solve the optimization problem

$$T_1^{\text{opt}}, \dots, T_K^{\text{opt}} = \underset{T_1, \dots, T_K}{\operatorname{argmax}} f(T_1, \dots, T_K) \quad (\text{A1})$$

under the constraints

$$T_i^\dagger T_i = \mathbb{1}$$

for isometries T_i and

$$\|T_j\|_F = 1$$

for the orthogonality center T_j .

In the following, we will discuss several approaches for solving optimization problem (A1). We will first assume that the input of the cost function is a single tensor T . If the cost function is linear, the problem is known as the *orthogonal Procrustes problem* [55] and we discuss its closed form solution in Appendix 1. Nonlinear cost functions and functions of multiple tensors can be optimized by using the Evenbly-Vidal algorithm; see Appendix 2.

A different, more involved approach to solving the optimization problem is given by Riemannian optimization, which we discuss in Appendix 3.

1. Orthogonal Procrustes problem

If the cost function is linear, it can be written as

$$f(T) = \sum_{j=1}^m \sum_{k=1}^n [\alpha_{j,k} \operatorname{Re}(T_{j,k}) + \beta_{j,k} \operatorname{Im}(T_{j,k})]$$

with parameters $\alpha_{j,k}, \beta_{j,k} \in \mathbb{R}$. Introducing the *environment tensor* $E \in \mathbb{C}^{m \times n}$ as $E_{j,k} = \alpha_{j,k} + i\beta_{j,k}$, we can write the cost

function as

$$f(T) = \sum_{j=1}^m \sum_{k=1}^n \text{Re}(E_{j,k}^* T_{j,k}) = \text{ReTr}(E^\dagger T) = \text{ReTr}(T^\dagger E).$$

Maximizing $f(T)$ under the isometry constraint $T^\dagger T = \mathbb{1}$ is known as the orthogonal Procrustes problem and permits the closed form solution

$$T_{\text{opt}} = \underset{T^\dagger T = \mathbb{1}}{\text{argmax}} \text{Re Tr}(T^\dagger E) = UV^\dagger, \quad (\text{A2})$$

where the matrices U and V are computed using an SVD $E = USV^\dagger$. To prove this result we insert the SVD into the cost function as

$$\begin{aligned} f(T) &= \text{Re Tr}(ET^\dagger) = \text{Re Tr}(USV^\dagger T^\dagger) \\ &= \text{Re Tr}[(U\sqrt{S})(\sqrt{S}V^\dagger T^\dagger)] \\ &= \text{Re}\langle \sqrt{S}U^\dagger, \sqrt{S}V^\dagger T^\dagger \rangle_F. \end{aligned}$$

We next use the fact that the Frobenius inner product satisfies the Cauchy-Schwarz inequality to obtain the upper bound

$$\begin{aligned} f(T) &= \text{Re}\langle \sqrt{S}U^\dagger, \sqrt{S}V^\dagger T^\dagger \rangle_F \leq \|\sqrt{S}U^\dagger\|_F \|\sqrt{S}V^\dagger T^\dagger\|_F \\ &= \sqrt{\text{Tr}(USU^\dagger)\text{Tr}(TVSV^\dagger T^\dagger)} = \text{Tr}(S), \end{aligned}$$

where in the last step we used $U^\dagger U = \mathbb{1}$, $V^\dagger V = \mathbb{1}$, $T^\dagger T = \mathbb{1}$, and the cyclic property of the trace. This upper bound is reached by the solution

$$F(T_{\text{opt}}) = \text{Re Tr}(USV^\dagger VU^\dagger) = \text{Tr}(S),$$

proving (A2).

If instead of satisfying the isometry condition the tensor T must be normalized to one, the closed form solution can be found as

$$T_{\text{opt}} = \underset{\|T\|_F=1}{\text{argmax}} \text{Re Tr}(T^\dagger E) = E/\|E\|_F. \quad (\text{A3})$$

We arrive at this solution through a similar argument as before. First, we obtain an upper bound

$$f(T) = \text{Re Tr}(T^\dagger E) = \text{Re}\langle T, E \rangle_F \leq \|T\|_F \|E\|_F = \|E\|_F$$

using the Cauchy-Schwarz inequality and the normalization constraint $\|T\| = 1$. We proceed by showing that the upper bound is reached by T^{opt} ,

$$f(T_{\text{opt}}) = \text{Re Tr}(EE^\dagger/\|E\|_F) = \|E\|_F,$$

proving (A3).

2. Evenbly-Vidal style optimization

In general, the cost function $f(T)$ is not linear. For example, a nonlinear cost function is encountered in the disentangling procedure when optimizing a MERA wave function [42]. It was proposed by Evenbly and Vidal [42,43] to linearize the cost function and to update the tensor T iteratively using the closed form solutions from Appendix 1. Let us assume that the cost function $f(T)$ can be written as the contraction of a tensor network, where in general the tensor T may appear multiple times. We contract all tensors except

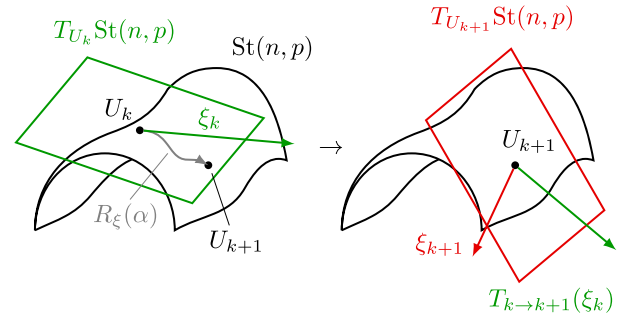


FIG. 14. In this figure, a visualization of optimization on Riemannian manifolds is given. The iterate X_k (left) is updated along the search direction ξ_k , which is an element of the tangent space $T_{X_k} \text{St}(n, p)$. The next iterate X_{k+1} is computed with the retraction $R_{\xi_k}(\alpha)$, where $\alpha_k \in \mathbb{R}$ is the step size. For the computation of the next search direction ξ_{k+1} some algorithms need the previous search direction ξ_k , which must first be brought to the tangent space $T_{X_{k+1}} \text{St}(n, p)$ of the new iterate X_{k+1} via the vector transport $T_{k \rightarrow k+1}(\xi_k)$.

one of the tensors T into an environment tensor $E_T \in \mathbb{C}^{n \times n}$ and the cost function becomes

$$f(T) = \text{Re Tr}(E_T T).$$

We now keep the environment E_T fixed, treating it as if it were independent of T , and update T with the closed form solutions (A2) or (A3). This is repeated until T is converged. If and how fast T converges depends on the details of the cost function, but convergence cannot be guaranteed for arbitrary cost functions.

Cost functions of multiple tensors $T_1, \dots, T_K$ can be optimized iteratively via an algorithm similar to the Evenbly-Vidal algorithm. Let us again assume that the cost function can be written as the contraction of a tensor network containing the tensors $T_1, \dots, T_K$. One can proceed by optimizing one tensor at a time as discussed above while keeping all other tensors fixed. If the cost function is linear in all tensors $T_1, \dots, T_K$ and bounded, $f(T_1, \dots, T_K) \leq c \in \mathbb{R}$, this algorithm is guaranteed to converge, since each local update is optimal and thus the cost function can never decrease.

3. Riemannian optimization

In the following, we give a brief review of Riemannian optimization algorithms over the manifold of isometric matrices $T \in \mathbb{C}^{n \times p}$, $T^\dagger T = \mathbb{1}$, which is called the *Stiefel manifold* $\text{St}(n, p)$. For a more in-depth introduction to the topic we recommend the book in [56]. Riemannian optimization of complex matrix manifolds is discussed in the context of quantum physics and isometric tensor networks in Refs. [57,58]. An implementation of Riemannian optimization on the real Stiefel manifold and other matrix manifolds in python is given in [59]. Some parts of this implementation were also used in our implementation [54].

A typical optimization algorithm iteratively improves an iterate $X_k \in \text{St}(n, p)$, $k = 1, 2, \dots$, until a local minimum of a cost function $f: \text{St}(n, p) \rightarrow \mathbb{R}$ is found. In Riemannian optimization, the gradient $\nabla f(X_k)$ of the cost function is restricted to the tangent space $T_{X_k} \text{St}(n, p)$ of the iterate X_k [56], which we visualize in Fig. 14. The gradient can be computed

either analytically or via automatic differentiation [57,59]. Optimization algorithms typically compute a search direction $\xi_k \in T_{X_k} \text{St}(n, p)$ and a step size $\alpha_k \in \mathbb{R}$ from the gradient. In an optimization algorithm defined on a Euclidean vector space one would then move along this direction as

$$\tilde{X}_{k+1} = X_k + \alpha_k \xi_k.$$

However, $\tilde{X}_{k+1}$ is in general not an element of the manifold. To ensure $X_{k+1} \in \text{St}(n, p)$, one can introduce a *retraction* $R_{\xi_k} : \mathbb{R} \rightarrow \text{St}(n, p)$ [56]. One can think of $R_{\xi_k}(\alpha_k)$ as moving along the direction of ξ_k while staying on the manifold. As α_k increases, we move further along the path defined by the retraction, with $R_{\xi_k}(0) = X_k$; see Fig. 14. Different retractions can be chosen, varying by how well they perform in optimization problems and by how hard they are to compute. A retraction for the Stiefel manifold that is particularly easy to compute while still yielding good results in practice is given by

$$R_{\xi_k}(\alpha_k) = \text{qf}(X_k + \alpha_k \xi_k),$$

where $\text{qf}(A)$ is the Q factor of the QR decomposition $A = QR$.

Many optimization algorithms, e.g., conjugate gradients, require gradients or search directions from previous iterates for computing a search direction at the current iterate. In Riemannian optimization, one must first bring these gradients and search directions from the tangent spaces of previous iterates to the tangent space of the current iterate. This is handled by a so-called *vector transport* $T_{k \rightarrow k+1}(\xi_k)$ [56]; see Fig. 14.

Finally, for optimization algorithms of second order such as the trust-region method, one needs to generalize the notion of the Hessian-vector product to Riemannian manifolds. This generalization is given by the *Riemannian connection* [56]. For the Stiefel manifold the Hessian-vector product is simply given by projecting the Hessian-vector product of the embedding Euclidean vector space $\mathbb{C}^{n \times p}$ to the tangent space [56].

In this work we use two algorithms for performing Riemannian optimization: conjugate gradients (CG) and the trust-region method (TRM); see also our implementation at Ref. [54].

a. Conjugate gradients

CG use the accumulated gradients of previous iterations to compute an improved search direction, trying to achieve faster convergence than simple gradient descent [56,60–62]. For an in-depth explanation of how CG on the Stiefel manifold can be implemented, see Refs. [56,58,63]. In our implementation we additionally use Powell's restart strategy [59,61,64], which can significantly improve the efficiency of CG.

b. Trust region methods

Trust-region methods are a class of second order optimization techniques that are known for their desirable global convergence properties with a local superlinear rate of convergence [56,65]. The main idea of TRM is to locally approximate the cost function f by a quadratic model $m_{X_k}(\eta) \approx f(X_k + \eta)$ with $\eta \in T_{X_k} \text{St}(n, p)$ in a *trust region* $\langle \eta, \eta \rangle \leq \Delta_k^2$ of radius Δ_k around the current iterate X_k . To define this model, one must compute the Hessian-vector product or an approximation thereof. This approximate cost function is then minimized within the trust region using truncated conjugate gradients (tCG), which converge quickly for the quadratic approximation. Depending on the quality of the approximation at the current iterate one can shrink or enlarge the trust region. TRM is able to achieve local superlinear convergence while still retaining global convergence properties [56] and can be thought of as an improved version of Newton's method. For more details on the TRM on Riemannian manifolds, see Refs. [56,59,65].

-
- [1] D. C. Tsui, H. L. Stormer, and A. C. Gossard, Two-dimensional magnetotransport in the extreme quantum limit, *Phys. Rev. Lett.* **48**, 1559 (1982).
 - [2] R. B. Laughlin, Anomalous quantum Hall effect: An incompressible quantum fluid with fractionally charged excitations, *Phys. Rev. Lett.* **50**, 1395 (1983).
 - [3] H. L. Stormer, D. C. Tsui, and A. C. Gossard, The fractional quantum Hall effect, *Rev. Mod. Phys.* **71**, S298 (1999).
 - [4] T.-H. Han, J. S. Helton, S. Chu, D. G. Nocera, J. A. Rodriguez-Rivera, C. Broholm, and Y. S. Lee, Fractionalized excitations in the spin-liquid state of a kagome-lattice antiferromagnet, *Nature (London)* **492**, 406 (2012).
 - [5] X.-G. Wen, *Colloquium*: Zoo of quantum-topological phases of matter, *Rev. Mod. Phys.* **89**, 041004 (2017).
 - [6] J. G. Bednorz and K. A. Müller, Possible high T_c superconductivity in the Ba-La-Cu-O system, *Z. Phys. B* **64**, 189 (1986).
 - [7] P. A. Lee, N. Nagaosa, and X.-G. Wen, Doping a Mott insulator: Physics of high-temperature superconductivity, *Rev. Mod. Phys.* **78**, 17 (2006).
 - [8] S. R. White, Density matrix formulation for quantum renormalization groups, *Phys. Rev. Lett.* **69**, 2863 (1992).
 - [9] J. Dukelsky, M. A. Martín-Delgado, T. Nishino, and G. Sierra, Equivalence of the variational matrix product method and the density matrix renormalization group applied to spin chains, *Europhys. Lett.* **43**, 457 (1998).
 - [10] U. Schollwöck, The density-matrix renormalization group in the age of matrix product states, *Ann. Phys. (N.Y.)* **326**, 96 (2011).
 - [11] R. Orús, A practical introduction to tensor networks: matrix product states and projected entangled pair states, *Ann. Phys. (N.Y.)* **349**, 117 (2014).
 - [12] J. Eisert, M. Cramer, and M. B. Plenio, *Colloquium*: Area laws for the entanglement entropy, *Rev. Mod. Phys.* **82**, 277 (2010).
 - [13] F. Verstraete and J. I. Cirac, Matrix product states represent ground states faithfully, *Phys. Rev. B* **73**, 094423 (2006).
 - [14] M. B. Hastings, An area law for one-dimensional quantum systems, *J. Stat. Mech.* (2007) P08024.
 - [15] T. Nishino and K. Okunishi, Corner transfer matrix renormalization group method, *J. Phys. Soc. Jpn.* **65**, 891 (1996).

- [16] T. Nishino and K. Okunishi, Corner transfer matrix algorithm for classical renormalization group, *J. Phys. Soc. Jpn.* **66**, 3040 (1997).
- [17] N. Maeshima, Y. Hieida, Y. Akutsu, T. Nishino, and K. Okunishi, Vertical density matrix algorithm: A higher-dimensional numerical renormalization scheme based on the tensor product state ansatz, *Phys. Rev. E* **64**, 016705 (2001).
- [18] F. Verstraete and J. I. Cirac, Renormalization algorithms for quantum-many body systems in two and higher dimensions, [arXiv:cond-mat/0407066](https://arxiv.org/abs/cond-mat/0407066).
- [19] M. Levin and C. P. Nave, Tensor renormalization group approach to two-dimensional classical lattice models, *Phys. Rev. Lett.* **99**, 120601 (2007).
- [20] J. Jordan, R. Orús, G. Vidal, F. Verstraete, and J. I. Cirac, Classical simulation of infinite-size quantum lattice systems in two spatial dimensions, *Phys. Rev. Lett.* **101**, 250602 (2008).
- [21] H. C. Jiang, Z. Y. Weng, and T. Xiang, Accurate determination of tensor network state of quantum lattice models in two dimensions, *Phys. Rev. Lett.* **101**, 090603 (2008).
- [22] J. Haegeman, C. Lubich, I. Oseledets, B. Vandereycken, and F. Verstraete, Unifying time evolution and optimization with matrix product states, *Phys. Rev. B* **94**, 165116 (2016).
- [23] M. Lubasch, J. I. Cirac, and M.-C. Bañuls, Algorithms for finite projected entangled pair states, *Phys. Rev. B* **90**, 064425 (2014).
- [24] M. P. Zaletel and F. Pollmann, Isometric tensor network states in two dimensions, *Phys. Rev. Lett.* **124**, 037201 (2020).
- [25] R. Haghshenas, M. J. O'Rourke, and G. K.-L. Chan, Conversion of projected entangled pair states into a canonical form, *Phys. Rev. B* **100**, 054404 (2019).
- [26] K. Hyatt and E. M. Stoudenmire, DMRG approach to optimizing two-dimensional tensor networks, [arXiv:1908.08833](https://arxiv.org/abs/1908.08833).
- [27] S.-H. Lin, M. P. Zaletel, and F. Pollmann, Efficient simulation of dynamics in two-dimensional quantum spin systems with isometric tensor networks, *Phys. Rev. B* **106**, 245102 (2022).
- [28] T. Soejima, K. Siva, N. Bultinck, S. Chatterjee, F. Pollmann, and M. P. Zaletel, Isometric tensor network representation of string-net liquids, *Phys. Rev. B* **101**, 085117 (2020).
- [29] Y.-J. Liu, K. Shtengel, and F. Pollmann, Simulating 2D topological quantum phase transitions on a digital quantum computer *Phys. Rev. Res.* **6**, 043256 (2024).
- [30] Z. Dai, Y. Wu, T. Wang, and M. P. Zaletel, Fermionic isometric tensor network states in two dimensions *Phys. Rev. Lett.* **134**, 026502 (2025).
- [31] Y. Wu, S. Anand, S.-H. Lin, F. Pollmann, and M. P. Zaletel, Two-dimensional isometric tensor networks on an infinite strip, *Phys. Rev. B* **107**, 245118 (2023).
- [32] M. S. J. Tepaske and D. J. Luitz, Three-dimensional isometric tensor networks, *Phys. Rev. Res.* **3**, 023236 (2021).
- [33] W. Kadow, F. Pollmann, and M. Knap, Isometric tensor network representations of two-dimensional thermal states, *Phys. Rev. B* **107**, 205106 (2023).
- [34] D. Malz and R. Trivedi, Computational complexity of isometric tensor network states, *PRX Quantum* **6**, 020310 (2025).
- [35] Z.-Y. Wei, D. Malz, and J. I. Cirac, Sequential generation of projected entangled-pair states, *Phys. Rev. Lett.* **128**, 010607 (2022).
- [36] L. Slattery and B. K. Clark, Quantum circuits for two-dimensional isometric tensor networks, [arXiv:2108.02792](https://arxiv.org/abs/2108.02792).
- [37] Y. Wu, Z. Dai, S. Anand, S.-H. Lin, Q. Yang, L. Wang, F. Pollmann, and M. P. Zaletel, Alternating and Gaussian Fermionic isometric tensor network states, *PRX Quantum* **6**, 040324 (2025).
- [38] G. Vidal, Efficient simulation of one-dimensional quantum many-body systems, *Phys. Rev. Lett.* **93**, 040502 (2004).
- [39] F. Verstraete, M. M. Wolf, D. Perez-Garcia, and J. I. Cirac, Criticality, the area law, and the computational power of projected entangled pair states, *Phys. Rev. Lett.* **96**, 220601 (2006).
- [40] R. J. Baxter and I. G. Enting, Series expansions from corner transfer matrices: The square lattice Ising model, *J. Stat. Phys.* **21**, 103 (1979).
- [41] M. Lubasch, J. I. Cirac, and M.-C. Bañuls, Unifying projected entangled pair state contractions, *New J. Phys.* **16**, 033014 (2014).
- [42] G. Evenbly and G. Vidal, Algorithms for entanglement renormalization, *Phys. Rev. B* **79**, 144108 (2009).
- [43] G. Evenbly and G. Vidal, Algorithms for entanglement renormalization: Boundaries, impurities and interfaces, *J. Stat. Phys.* **157**, 931 (2014).
- [44] J. Unfried, J. Hauschild, and F. Pollmann, Fast time evolution of matrix product states using the QR decomposition, *Phys. Rev. B* **107**, 155133 (2023).
- [45] H. W. J. Blöte and Y. Deng, Cluster Monte Carlo simulation of the transverse Ising model, *Phys. Rev. E* **66**, 066110 (2002).
- [46] J. Hauschild and F. Pollmann, Efficient numerical simulations with tensor networks: Tensor network Python (TeNPy), *SciPost Phys. Lect. Notes*, **5** (2018).
- [47] M. P. Zaletel, R. S. K. Mong, C. Karrasch, J. E. Moore, and F. Pollmann, Time-evolving a matrix product state with long-ranged interactions, *Phys. Rev. B* **91**, 165112 (2015).
- [48] G. Vidal, Efficient classical simulation of slightly entangled quantum computations, *Phys. Rev. Lett.* **91**, 147902 (2003).
- [49] M. Urbanek and P. Soldán, Parallel implementation of the time-evolving block decimation algorithm for the Bose–Hubbard model, *Comput. Phys. Commun.* **199**, 170 (2016).
- [50] V. Volokitin, I. Vakulchyk, E. Kozinov, A. Liniov, I. Meyerov, M. Ivanchenko, T. Lapyeva, and S. Denisov, Propagating large open quantum systems towards their steady states: Cluster implementation of the time-evolving block decimation scheme, *J. Phys.: Conf. Ser.* **1392**, 012061 (2019).
- [51] R.-Y. Sun, T. Shirakawa, and S. Yunoki, Improved real-space parallelizable matrix-product state compression and its application to unitary quantum dynamics simulation, *Phys. Rev. B* **110**, 085149 (2024).
- [52] S.-H. Lin, R. Dilip, A. G. Green, A. Smith, and F. Pollmann, Real-and imaginary-time evolution with compressed quantum circuits, *PRX Quantum* **2**, 010342 (2021).
- [53] B. Sappller, M. Kawano, M. Zaletel, and F. Pollmann, Diagonal isometric form for tensor network states in two dimensions [Data set], Zenodo (2025), <https://doi.org/10.5281/zenodo.17311440>.
- [54] B. Sappller, YB-isoTNS (2025), <https://github.com/SnackerBit/YB-isoTNS>.

- [55] J. C. Gower and G. B. Dijksterhuis, *Procrustes Problems* (Oxford University Press, New York, 2004), Vol. 30.
- [56] P.-A. Absil, R. Mahony, and R. Sepulchre, *Optimization Algorithms on Matrix Manifolds* (Princeton University Press, Princeton, NJ, 2008).
- [57] I. A. Luchnikov, M. E. Krechetov, and S. N. Filippov, Riemannian geometry and automatic differentiation for optimization problems of quantum physics and quantum technologies, *New J. Phys.* **23**, 073006 (2021).
- [58] M. Hauru, M. Van Damme, and J. Haegeman, Riemannian optimization of isometric tensor networks, *SciPost Phys.* **10**, 040 (2021).
- [59] J. Townsend, N. Koep, and S. Weichwald, Pymanopt: A Python toolbox for optimization on manifolds using automatic differentiation, *J. Mach. Learn. Res.* **17**, 1 (2016).
- [60] J. R. Shewchuk, *et al.*, *An Introduction to the Conjugate Gradient Method without the Agonizing Pain* (Carnegie-Mellon University, Pittsburgh, PA, 1994).
- [61] W. W. Hager and H. Zhang, A survey of nonlinear conjugate gradient methods, *Pacific J. Optimization* **2**, 35 (2006).
- [62] W. W. Hager and H. Zhang, Algorithm 851: CG_DESCENT, a conjugate gradient method with guaranteed descent, *ACM Trans. Math. Software* **32**, 113 (2006).
- [63] X. Zhu, A Riemannian conjugate gradient method for optimization on the Stiefel manifold, *Comput. Optim. Appl.* **67**, 73 (2017).
- [64] C.-X. Kou, W.-H. Zhang, W.-B. Ai, and Y.-F. Liu, On the use of Powell's restart strategy to conjugate gradient methods, *Pacific J. Optimization* **10**, 85 (2014).
- [65] P.-A. Absil, C. G. Baker, and K. A. Gallivan, Trust-region methods on Riemannian manifolds, *Found. Comput. Math.* **7**, 303 (2007).
- Correction:* A missing funding support statement has been added.