UNIVERSITAT DE BARCELONA

MASTER IN QUANTUM SCIENCE AND TECHNOLOGY

Master's thesis

Long-time evolution of quantum systems with Tensor Network techniques

---

Marc Farreras Bartra

Supervisor: Stefano Carignano

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

Long-time evolution of quantum systems with Tensor Network techniques

Marc Farreras

Supervised by: Stefano Carignano

Barcelona Supercomputing Centre, Plaça Eusebi Güell, 08034 Barcelona
25 August 2024

One of the most significant challenges in simulating the dynamics of many-body quantum systems is the exponential increase in computational complexity, driven by the linear growth of entanglement during time evolution. In this work, we explore a novel algorithm designed to mitigate this complexity by identifying and trading the entanglement by mixture, i.e. depurifying the originally pure closed-system state. This approach preserves the essential local information necessary for computing expectation values, while effectively reducing the computational resources. Additionally, we propose a method for performing time evolution with the resulting mixed states, highlighting both the strengths and limitations of this approach. We also discuss potential avenues for future improvements to enhance the efficiency and applicability of this method.

Keywords: Tensor Networks, time evolution, out-of-equilibrium, quantum many body,TFM

Acknowledgements

I would like to express my deepest gratitude to my supervisor, Dr. Stefano Carignano, for his invaluable time and guidance throughout this project. His knowledge and unwavering support have been of immense value during the many months we have worked together. I would also like to extend my thanks to the entire Quantic group for their warm welcome, continuous support, and the insightful discussions that have greatly contributed to my work. Additionally, I am sincerely grateful to Dr. Luca Tagliacozzo and Carlos Ramos for their teachings and assistance, which have enriched my knowledge and equipped me with the tools necessary to undertake this project and future endeavors.

Marc Farreras: mfarreba27@alumnes.ub.edu

Contents

1 Introduction

The study of quantum many-body systems has long posed significant challenges. Analytically, solutions are known for only a limited number of cases, difficulting further progress. Numerically, the primary obstacle is the curse of dimensionality, which refers to the exponential growth of the Hilbert space as the number of particles in the system increases.

However, in recent decades, it has been recognized that physically relevant states, such as the ground states of gapped Hamiltonians, exhibit much less entanglement than theoretically possible, typically satisfying an area law for entanglement[ECP10]. This suggests that physically relevant states possess additional local structures that set them apart from random states in the Hilbert space, indicating that they occupy a much smaller region.

In this context, tensor networks (TN) have emerged as powerful tools for simulating quantum many-body systems. Their effectiveness stems largely from their ability to capture the mentioned local structures, for instance, by satisfying an area law for entanglement[EV11, CPGSV21], thereby enabling an efficient representation of low-energy states. Moreover, tensor networks can be easily adapted to different geometries and symmetries.

In recent years, many numerical methods, particularly for 1-D systems, have been developed within the Matrix Product State (MPS) framework. Among these methods, the Density Matrix Renormalization Group (DMRG) algorithm[Whi92, Whi93], along with its higher-dimensional counterparts[Sch11], stands out as the most accurate method for studying ground states of quantum systems.

Contrasting with equilibrium systems, studying the dynamical properties of out-of-equilibrium systems has proven challenging. Although efficient methods exist for simulating the time evolution of MPS wave functions, such as TEBD[Vid03] and t-DMRG[WF04], in out-of-equilibrium scenarios, during such evolution, initially localized correlations can extend over arbitrarily large distances[CC05]. As a result, the entropy of these systems grows linearly in time, leading to an exponential increase in the computational resources required to simulate them. Consequently, these methods typically yield reliable results only for short timescales[TCF+12].

Despite these challenges, the research community has pushed the boundaries of standard TN, developing new approaches to manage linear entanglement growth. These algorithms generally focus on accurately computing the time evolution of local operator expectation values rather than fully capturing the entire quantum state. Several strategies have been explored, such as emphasizing entanglement in the temporal direction and the light-cone structure of interactions [FPBn22, BnHVC09], or replacing the quantum state with a mixed state to capture only the locally relevant information[HLB+18, WZMR18].

In this work, we review and characterize an algorithm that falls into the second category of approaches, as proposed by Frias, Tagliacozzo, and Bañuls in Ref.[FPTBn24]. Building on concepts from Ref.[SPT19], they developed a tensor network algorithm capable of identifying and eliminating the contributions of long-range entanglement within a system by replacing the quantum state with a local mixture. This innovative approach allows for the removal of long-range entanglement, enabling the accurate benchmarking of the time evolution of local operators over significantly longer times than was previously possible.

Our objective has been to thoroughly characterize the algorithm, faithfully describing the various steps involved and the challenges encountered along the way. We have also highlighted the strengths and weaknesses of the algorithm and suggested further steps that could be taken to better understand its reliability and the systems and conditions under which it achieves optimal efficiency.

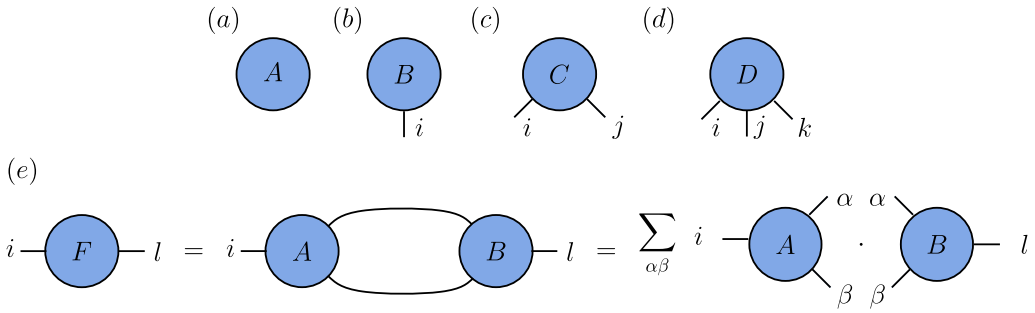

Figure 1: Diagrammatic representation of (a) a scalar A, (b) a vector B, (c) a matrix C, (d) an order-3 tensor D, and (e) the contraction between two order-3 tensors to form an order-2 tensor.

The rest of the work is organized as follows: In Section 2, we define the formalism of MPS, reviewing the foundational concepts with a particular focus on the infinite system variant and the methods for performing time evolution. Section 3 details the main ideas underlying the algorithm, explains how these ideas are implemented within a TN framework, and proposes our method for time evolution. In Section 4, we present the results of our simulations, and finally, in Section 5, we discuss these results and suggest potential avenues for further improvement.

2 Formalism

2.1 Basics of TN

Tensors can be defined in various ways depending on the context. In this work, we adopt a practical approach from a computer science perspective, where tensors are understood as multidimensional arrays of complex numbers. The order of a tensor is determined by the number of indices required to uniquely identify each element within it. For instance, a scalar, with no indices, is an order-0 tensor; a vector, with one index, is an order-1 tensor; a matrix, with two indices, is an order-2 tensor, and higher-order tensors have correspondingly more indices.

While tensor calculus can sometimes appear obscure and complex, one of the key advantages of Tensor Networks (TN) is their intuitive and straightforward pictorial notation. This notation simplifies the study and development of algorithms by making the general properties of tensors more visually accessible. The structure of the tensor network itself often reveals essential insights into the underlying properties of the tensors being studied.

In diagrammatic notation, tensors are depicted as geometric shapes with legs extending from them, where each leg corresponds to an index of the tensor, see Fig.1. Standard operations involving tensors can also be intuitively represented. For additional examples of tensor operations and their diagrammatic representations, refer to Appendix A.

Another essential operation in Tensor Networks is the Singular Value Decomposition (SVD) [KL80]. For an arbitrary matrix M, its SVD is expressed as

$$M = USV^\dagger, \tag{1}$$

where U and $V^\dagger$ are isometries (matrices with orthonormal columns) and S is a diagonal matrix containing the singular values, which are non-negative real numbers.

SVD plays a crucial role in tensor networks due to its connection with the Schmidt decomposition and because it offers an optimal method for approximating large tensors by truncating small singular values. See Appendix B for more details.

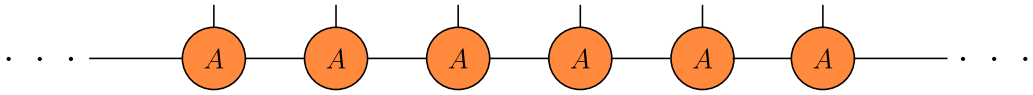

Figure 2: Diagrammatic representation of an iMPS.

2.2 Matrix Product States

Matrix Product States (MPS) can be defined in various ways, each highlighting different aspects of the MPS structure. In this work, we adopt a practical approach by viewing the MPS as a wavefunction ansatz.

Consider a 1-D chain with N sites, where each site corresponds to a d-dimensional Hilbert space spanned by an orthonormal basis $\{|s_i\rangle\}$. To describe an arbitrary quantum state $|\Psi\rangle \in \mathbb{C}^{d^{\otimes N}}$, we employ the Matrix Product State (MPS) ansatz:

$$|\Psi\rangle = \sum_{s_1,...,s_N} A^{[1]}_{s_1} A^{[2]}_{s_2} \cdots A^{[N]}_{s_N} |s_1, ..., s_N\rangle, \tag{2}$$

where each $A^{[i]}$ is a tensor with dimensions $d \times m_i \times m_{i+1}$, where m_i and m_{i+1} are virtual indices that are contracted sequentially to form a product of tensors. The physical index s_i corresponds to the local Hilbert space at each site. Notice that the tensors at the boundaries of the chain, $A^{[1]}$ and $A^{[N]}$, are treated as order-3 tensors, where the initial and final virtual indices satisfy $m_1 = m_N = 1$. The dimension of the virtual indices, commonly referred to as the bond dimension, is denoted by D.

The MPS representation drastically reduces the number of parameters needed to describe a quantum state. Instead of requiring 2^N coefficients, which scale exponentially with N, the MPS form requires only $N \times d \times D^2$ parameters, scaling linearly with N.

One can construct a MPS via performing several SVD, as described in Appendix C.1.

2.3 Infinite MPS

The MPS formalism can also be extended to the thermodynamic limit for translationally invariant systems. Since our work is centered on the simulation of infinite systems, we will briefly explain the infinite case and focus exclusively on this scenario from here onwards.

In the infinite case, the states are constructed by repeating a single tensor, or a group of tensors (depending on the system's unit cell), infinitely across all lattice sites. These MPS are referred to as infinite MPS (iMPS)[Vid07, OV08] or uniform MPS (uMPS)[VHV19] in the literature.

To illustrate this concept, consider an infinite 1-D lattice where each site $r \in \mathbb{Z}$ is associated with a Hilbert space of dimension d. An infinite MPS on this chain can be described by

$$|\Psi(A)\rangle = \sum_{\{s\}} \prod_{r\in\mathbb{Z}} A^{s_r} |\{s_r\}\rangle, \tag{3}$$

where the unit cell consists of a single order-3 tensor A with dimensions $d \times D \times D$, which is repeated infinitely. The notation $\{s\}$ represents the set of all sites in the lattice, and, as before, the contraction of the virtual indices is implied but not explicitly shown. This state is depicted diagrammatically in Fig.2.

It is important to note that there is a gauge freedom in representing the state: one can insert an identity XX^{-1}, being X an invertible matrix, between two tensors A, resulting in a new MPS that, while different, still represents the same physical state. This gauge

freedom is typically partially fixed by working in canonical forms, which are representations where the leading left and right vectors are associated with identities. These canonical forms are essential for ensuring numerical stability and efficiency. For a more in-depth explanation of canonical forms in iMPS, refer to Appendix C.2.

2.4 Time evolution in iMPS

Time evolution is essential to studying the dynamics of out-of-equilibrium systems. Various algorithms have been developed for this purpose, with comprehensive reviews available in Refs.[Bañ23, Sch11, PKS+19]. In this work, we focus on the time-evolving block decimation (TEBD) algorithm [Vid03], with particular emphasis on its infinite variant (iTEBD)[Vid07, OV08], which is specifically designed for iMPS.

The time evolution of an initial state $|\psi(0)\rangle$ to a state $|\psi(t)\rangle$ at time t is governed by

$$|\Psi(t)\rangle = e^{-it\hat{H}}|\psi(0)\rangle = U(t)|\psi(0)\rangle, \tag{4}$$

where $|\Psi(t)\rangle$ represents the state at time t and $U(t) = e^{-it\hat{H}}$ is the time evolution operator corresponding to the Hamiltonian $\hat{H}$. To implement this time evolution, the total time interval is divided into smaller steps, such that $t = N_t\delta_t$, where δ_t is the time step. We then apply a first-order Suzuki-Trotter decomposition[Suz85], which breaks down the evolution operator into a sequence of simpler, local operators. Specifically, the Hamiltonian $\hat{H}$ is split into H_e and H_o, corresponding to interactions on even and odd bonds, respectively. The time evolution operator is thus decomposed into two-body operators $U_1 = e^{-i\delta_t H_e}$ and $U_2 = e^{-i\delta_t H_o}$.

The iTEBD leverages the translational invariance of the system's unit cell. Although applying $e^{-i\delta_t H_{o(e)}}$ disrupts translational invariance under single-site shifts, it preserves it under two-site shifts. Thus, only the translationally invariant unit cell needs updating, achievable through simple matrix multiplications and SVD.

Before starting the algorithm, the infinite MPS must be in what we refer to as the Vidal form, where the unit cell consists of two tensors $\{\Gamma, \Lambda\}$. In this representation, Γ is an order-3 tensor, and Λ is a diagonal matrix containing the singular values associated with the bipartition of the state at the bond where Λ is located, see Appendix C.2 for a more detailed explanation. Due to the partial breaking of translational invariance, the unit cell size increases to $\{\Gamma^A, \Gamma^B, \Lambda^{AB}, \Lambda^{BA}\}$, where A and B represent tensors on even and odd sites, respectively.

The algorithm begins by applying the evolution operator $U_1(\delta_t)$ to all even bonds. This involves computing a new tensor $\Theta^{[1]}$, defined as

$$\Theta^{[1]\sigma_r,\sigma_{r+1}}_{\alpha,\beta} = (\Lambda^{BA}\Gamma^A\Lambda^{AB}\Gamma^B\Lambda^{BA})^{\sigma_r,\sigma_{r+1}}_{\alpha,\beta}, \tag{5}$$

where contraction over internal virtual indices is implied. Absorbing Λ^{BA} ensures that the state remains in the Schmidt basis, enabling controlled truncation during SVD.

Next, we apply the two-body operator U_1 to $\Theta^{[1]}$, yielding

$$\tilde{\Theta}^{[1]\sigma_r,\sigma_{r+1}}_{\alpha,\beta} = \sum_{\sigma'_r,\sigma'_{r+1}} U^{\sigma'_r,\sigma'_{r+1}}_{[1]\sigma_r,\sigma_{r+1}} \Theta^{[1]\sigma'_r,\sigma'_{r+1}}_{\alpha,\beta}, \tag{6}$$

as illustrated in Fig. 3 (a).

After reshaping $\tilde{\Theta}^{[1]}$, we perform SVD and reshape the matrices back to obtain

$$\tilde{\Theta}^{[1]\sigma_r\sigma_{r+1}} = U^{[A]\sigma_r}\tilde{\Lambda}^{AB}V^{[B]\sigma_{r+1}}, \tag{7}$$

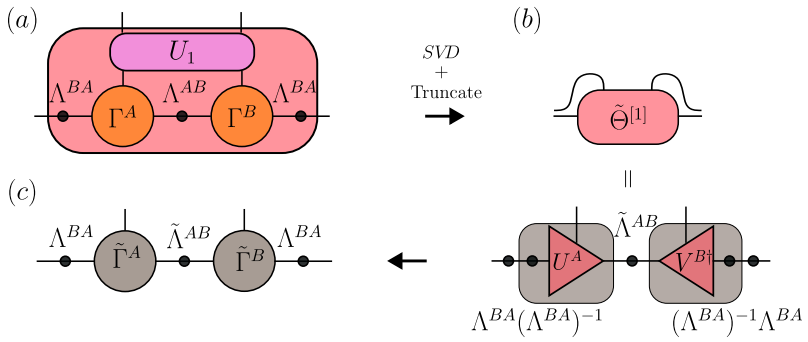

Figure 3: Diagrammatic representation of the update of the even bond in the iTEBD. (a) Application of the two-body evolution operator $U_1(\delta_t)$ to the tensor $\Theta^{[1]}$ in an iMPS, showing the contraction with even-site tensors. (b) Singular value decomposition of the evolved tensor $\tilde{\Theta}^{[1]}$, followed by truncation to retain the largest D singular values. (c) Restoration of the Vidal form by updating the $\Gamma^{[A]}$ and $\Gamma^{[B]}$ tensors, ensuring the state remains in the canonical form suitable for further time evolution.

where $U^{[A]}$ and $V^{[B]\dagger}$ are isometries, and $\tilde{\Lambda}^{AB}$ contains the singular values. We truncate by retaining only the largest D singular values, as shown in Fig. 3(b).

To restore the Vidal form, given that $U^{[A]}$ and $V^{[B]\dagger}$ are left-normalized and right-normalized tensors, respectively, we can use the relationship from Eq.(47). By inserting the identity $\lambda^{BA}(\lambda^{BA})^{-1}$ where necessary and then absorbing $(\lambda^{BA})^{-1}$, we obtain the updated $\Gamma^{[A]}$ and $\Gamma^{[B]}$ tensors, effectively restoring the Vidal form. This final step is depicted in Fig. 3(c).

The next step is to update the odd bonds, following the same procedure: contract the tensors to form $\Theta^{[2]}$, apply the gate U_2, perform SVD, truncate, and recover the new tensors Γ^A and Γ^B. This process is repeated N_t times to obtain the final state $|\Psi(t)\rangle$.

3 Algorithm

3.1 Physical motivation

In this work, we address the challenges of simulating out-of-equilibrium dynamics, specifically those induced by quantum quenches [Mit18, PSSV11] in 1-D infinite systems. Prior to the quench, the system resides in equilibrium, in the ground state $|\Psi\rangle$ of a local Hamiltonian H.

The quench drives the initial state $|\Psi\rangle$ into a highly excited state under the new Hamiltonian $\tilde{H}$, which acts as a source of entangled quasi-particle (QP) excitations[CC05]. These entangled QPs possess well-defined momenta and propagate in opposite directions with a finite velocity v_k, spreading entanglement across the system as they separate over time. This dynamic results in a linear growth of entanglement [SLRD13, LC08].

This concept is illustrated in Fig. 4 for an infinite system. If we divide the system into two semi-infinite subsystems, A and B, the number of QPs moving from A to B increases linearly with time, leading to a corresponding linear increase in entanglement.

This framework reveals that faster QPs rapidly increase the system's entanglement. If one can identify these fast modes or QPs and eliminate them without affecting the local information needed for calculating expectation values, a more efficient description of the evolved state becomes possible.

This is precisely the approach taken by the proposed TN algorithm. It is important to note that this algorithm is designed for infinite systems, where the absence of boundary

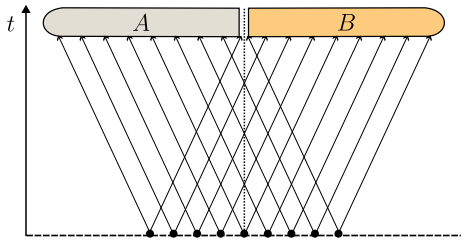

Figure 4: Space-time diagram illustrating the linear entanglement growth in an infinite system partitioned into subsystems A and B. Black points represent sources of entangled QP pairs moving in opposite directions. Movement is shown along the light cone for clarity.

conditions allows QPs to travel indefinitely without encountering obstacles that could affect their behavior.

3.2 Identifying long-distance entanglement

To detect the contributions of fast and slow modes, we focus on a reduced section of our system, referred to as subsystem S, which consists of l neighboring spins. The remaining semi-infinite subsystem to the left is denoted as L, and the one to the right as R.

In the QP picture, the entanglement across a bipartition can be understood as the presence of correlated particles on both sides of a given cut. For example, consider the bipartition between subsystems L and SR. If a pair of correlated QPs is entirely contained within L, there is no entanglement between the bipartitions. However, if one QP is in L and the other in SR, entanglement is present between these regions.

With this setup in mind, we define long-distance entanglement as the entanglement contribution generated by QPs that have support in both L and R but not in S. As time progresses, the distance between the QPs increases, eventually causing them to leave the S subsystem, thereby creating long-distance entanglement. At this point, the entanglement is only between the L and R subsystems, with S remaining in a product state with respect to the other two.

In a general scenario, complete factorization of S is not always possible. This is due not only to potential additional sources of entanglement between S and LR but also because, even if we assume that the only entanglement arises from the QP picture, QPs with different velocities could lead to one pair of QPs having support only in L and R, while another pair has support in L and S. At this stage, even though the subsystem S cannot be completely factorized from LR, parts of the subsystems that do factorize from S can still be identified.

Mathematically, this can be understood by finding a slow-fast factorization of our Hilbert space such that $L \equiv L_s \otimes L_f$ and $R \equiv R_s \otimes R_f$, where the subsystem $L_f \otimes R_f$ is no longer entangled with S. In the simple QP scenario, $L_f \otimes R_f$ represents the fast degrees of freedom that generate long-distance entanglement, or equivalently, the fast mode QP pairs.

Once we have identified the fast degrees of freedom, we can factorize our state $|\Psi\rangle$ as

$$|\Psi\rangle = |\psi^{slow}_{LSR}\rangle \otimes |\phi^{fast}_{L_f R_f}\rangle, \tag{8}$$

where $|\psi^{slow}_{LSR}\rangle$ represents the part of the state that retains the slow mode QPs, and $|\phi^{fast}_{L_f R_f}\rangle$ captures the fast mode QPs.

In a translationally invariant system, we can revisit the LSR cut picture for each block of l spins. Suppose we are interested in computing the expectation value of a local observable. In this scenario, we need to trace out all degrees of freedom except those

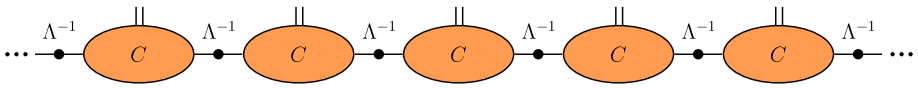

Figure 5: Diagrammatic representation of the initial state for the factorization algorithm, the unit cell is formed by the tensors $\{C, \Lambda^{-1}\}$.

involved in the operator. Since fast-mode QPs are separated by at least l sites, tracing out the non-involved degrees of freedom will result in discarding at least one QP from the fast-mode QP pair, leaving the remaining QP in a mixed state.

Therefore, when computing expectation values of local observables, it is not essential to preserve the complete description of the entire system. Instead, the system can be effectively represented as a mixed state by discarding the information linked to the fast degrees of freedom, thereby eliminating long-range entanglement.

To achieve this, we begin by constructing the density matrix ρ for our state $|\Psi\rangle$. Using Eq.(8), we obtain

$$\rho = \left|\psi^{slow}_{LSR}\right\rangle \left\langle\psi^{slow}_{LSR}\right| \otimes \left|\phi^{fast}_{L_f R_f}\right\rangle \left\langle\phi^{fast}_{L_f R_f}\right| = \rho^{slow}_{LSR} \otimes \rho^{fast}_{L_f R_f}, \tag{9}$$

where $\rho^{fast}_{L_f R_f}$ represents the density matrix of the fast modes, and ρ^{slow}_{LSR} represents the density matrix of the slow modes.

From Eq.(9), the density matrix $\rho^{fast}_{L_f R_f}$ still retains the long-distance entanglement. To eliminate this contribution without affecting the relevant local information needed for computing our expected values, we replace $\rho^{fast}_{L_f R_f}$ with a mixed state obtained by tracing out one of the entangled pairs in the fast modes

$$\rho^{fast}_{L_f R_f} = \left|\phi^{fast}_{L_f R_f}\right\rangle \left\langle\phi^{fast}_{L_f R_f}\right| \;\rightarrow\; \rho^{fast}_{L_f} \otimes \rho^{fast}_{R_f}, \tag{10}$$

where $\rho^{fast}_{L_f(R_f)} = tr_{R(L)}(\rho^{fast}_{L_f R_f})$ is the reduced density matrix obtained by tracing out one of the subsystems.

The obtained mixed state is globally different from the original state ρ. Nevertheless, concerning the expectation value of a local operator, both states should be indistinguishable because they carry the same local information needed for the computation. Moreover, the mixed state has removed the long-distance entanglement components, thereby providing a more efficient local description in terms of correlations. Consequently, this mixed state could enhance methods for simulating the evolution of expected values, particularly in cases where entanglement growth presents a major computational barrier, such as with tensor network (TN) methods.

The next step is to translate the above idea into a TN algorithm to leverage the benefits of this mixed state. Focusing on the time evolution of infinite 1-D systems, we describe how this QP picture can be integrated into the iMPS framework to obtain a more efficient description for local observables.

We start by assuming that our system is described by an iMPS in Vidal form, where the unit cell is composed of $\{\Gamma, \Lambda\}$, with Γ representing a subsystem S that consists of l neighboring spins. This configuration can always be achieved by grouping l tensors, such as when each tensor initially corresponds to a single site. For simplicity, we will assume $l = 2$ throughout this work, although the approach applies to any l. Next, we absorb the singular values at both ends of Γ into Γ itself, resulting in a new unit cell composed of $\{C, \Lambda^{-1}\}$, as illustrated in Fig. 5. This step enables us to directly utilize the orthogonality properties of the canonical forms within the tensor C.

In this form, when singling out the subsystem S, the state can be described by

$$|\Psi\rangle = \sum_{\alpha,s_l,\beta} C^{s_l}_{\alpha,\beta} \left|\Phi^L_\alpha\right\rangle |s_l\rangle \left|\Phi^R_\beta\right\rangle, \tag{11}$$

where $|s_l\rangle$ represents the basis for the subsystem S, while $\{\left|\Phi^L\right\rangle\}$ and $\{\left|\Phi^R\right\rangle\}$ are the Schmidt basis representing the semi-infinite chains to the left and right of the subsystem S, respectively. These can be directly understood as the basis representing the subsystems L and R.

Therefore, by using this representation, we can efficiently analyze and manage the long-distance entanglement structure within the system at the level of the virtual indices. Specifically, the fast degrees of freedom that decouple from the subsystem S will correspond to a new tensor ϕ_{fast}, which connects the left and right bond dimensions and forms a product state with respect to the tensor containing the slow degrees of freedom ψ_{slow}.

If such a form exists, there would be a basis transformation $U_L^\dagger$ and $V_R^\dagger$ acting on the subsystems L and R respectively, enabling us to factorize them into $L_f \otimes L_s$ and $R_f \otimes R_s$. Applying these transformations to our initial state, we obtain

$$U_L^\dagger \otimes \mathbb{1}_S \otimes V_R^\dagger |\Psi\rangle = |\psi^{slow}_{LSR}\rangle \otimes |\phi^{fast}_{L_f R_f}\rangle. \tag{12}$$

To determine these unitary transformations, assuming they exist, we need to find them variationally. The simplest approach involves minimizing the overlap between the left and right sides of the equality in Eq. (12). This leads to the following minimization problem:

$$\max_{U_L,V_R,|\psi^{slow}_{LSR}\rangle,|\phi^{fast}_{L_f R_f}\rangle} |\langle\psi^{slow}_{LSR}| \otimes \langle\phi^{fast}_{L_f R_f}| U_L^\dagger \otimes \mathbb{1}_S \otimes V_R^\dagger |\Psi\rangle|, \tag{13}$$

where the optimization variables are the unitaries U_L and V_R, and the final factorized states $\left|\psi^{slow}_{LSR}\right\rangle$ and $|\phi^{fast}_{L_f R_f}\rangle$.

Although complete optimization is a complex problem, optimizing each component individually is quadratic and can be solved analytically, as first noted in Ref. [KRB+18]. Therefore, the approach is to start with initial random states and unitaries of fixed dimensions, then iteratively optimize each component while keeping the others fixed until a fixed point is reached.

First, let's review the optimization of the states. For simplicity, we will demonstrate the optimization of $|\psi^{slow}_{LSR}\rangle$, noting that the optimization of $|\phi^{fast}_{L_f R_f}\rangle$ follows equivalently by interchanging the states. Our new optimization problem becomes

$$\max_{|\psi^{slow}_{LSR}\rangle} |\langle\psi^{slow}_{LSR}| \otimes \langle\phi^{fast}_{L_f R_f}| U_L^\dagger \otimes \mathbb{1}_S \otimes V_R^\dagger |\Psi\rangle| = \max_{|\psi^{slow}_{LSR}\rangle} |\langle\psi^{slow}_{LSR}||\varphi\rangle|, \tag{14}$$

where $|\varphi\rangle \equiv \langle\phi^{fast}_{L_f R_f}| U_L^\dagger \otimes \mathbb{1}_S \otimes V_R^\dagger |\Psi\rangle$. The solution to this optimization problem is simply $|\psi^{opt}_{slow}\rangle \propto |\varphi\rangle$. This new state in tensor network notation is constructed as shown in Fig. 6 (a). It is worth to highlight that due to the quadratic nature of this optimization problem, it can equivalently be solved using a DMRG scheme, as discussed in Appendix D.

For optimizing the unitaries, we will follow a similar scheme. We fix all other tensors while optimizing the one of interest. Here, we will show the optimization for the U_L unitary, noting that the process is equivalent for V_R. Our new optimization function becomes

$$\begin{aligned}\max_{U_L} |\langle\psi^{slow}_{LSR}| \otimes \langle\phi^{fast}_{L_f R_f}| U_L^\dagger \otimes \mathbb{1}_S \otimes V_R^\dagger |\Psi\rangle| &= \max_{U_L} |\langle\psi^{slow}_{LSR}| \otimes \langle\phi^{fast}_{L_f R_f}| U_L^\dagger |\tilde{\psi}\rangle| \\ = \max_{U_L} |\operatorname{tr}(U_L^\dagger |\tilde{\psi}\rangle\langle\psi^{slow}_{LSR}| \otimes \langle\phi^{fast}_{L_f R_f}|)| &= \max_{U_L} |\operatorname{tr}_L(U_L^\dagger \rho_L)|,\end{aligned} \tag{15}$$

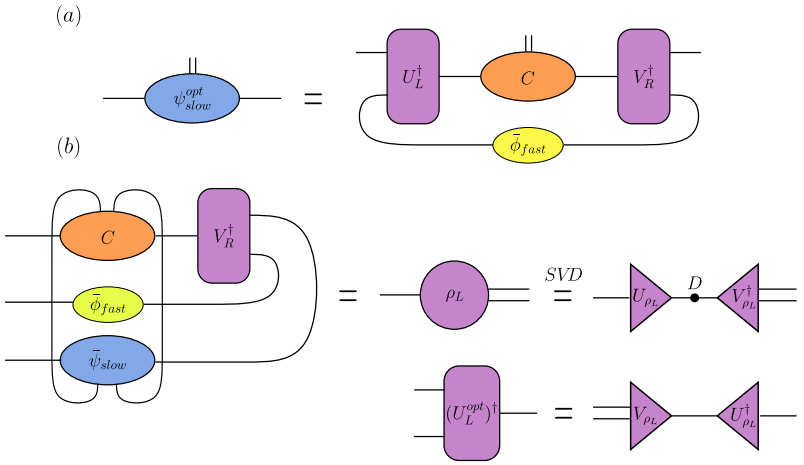

Figure 6: (a) Representation of how the new state $|\psi^{opt}_{slow}\rangle$ can be achieved using tensor networks. In this representation, the tensor C represents the original state $|\Psi\rangle$, the tensor $\bar{\phi}_{fast}$ represents $\langle\phi^{fast}_{L_f R_f}|$, and $U_L^\dagger$ and $V_R^\dagger$ are the unitaries that achieve the factorization.(b)Representation of the process to obtain the new U_L^{opt} tensor. The ρ_L corresponds to the environment tensor. The optimal U_L^{opt} is achieved by combining the isometries obtained in the SVD of the environment tensor ρ_L.

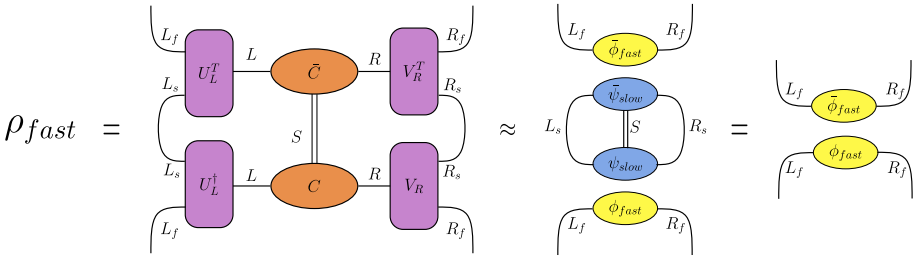

Figure 7: Representation of the calculation of the ρ_{fast} tensor. The symbol $\approx$ is used to indicate that due to the numerical optimization not being perfect, there may be some difference between the two cases. Additionally, in the last equality, we use that the tensor ψ_{slow} represents a physical state, so $Tr(\rho_{slow}) = 1$.

where $|\tilde{\psi}\rangle \equiv \mathbb{1}_S \otimes V_R^\dagger |\Psi\rangle$ and $\rho_L = \mathrm{tr}_{S,R}(|\tilde{\psi}\rangle\langle\psi^{slow}_{LSR}| \otimes \langle\phi^{fast}_{L_f R_f}|)$.

The final expression in Eq. (15) is closely related to the variational form of the trace norm or Schatten 1-norm [Wil13]. Consequently, the maximum value of the cost function corresponds to the sum of the singular values of ρ_L, which can be found using the singular value decomposition $\rho_L = U_{\rho_L} D V^\dagger_{\rho_L}$. The optimal unitary that achieves the maximum value of the cost function is $U_L^{opt} = U_{\rho_L} V^\dagger_{\rho_L}$. This process is depicted in Fig. 6(b).

After solving the optimization problem, we need to verify whether we have successfully acquired the desired structure from Eq.(12). A useful measure for this purpose is the von Neumann entropy of the reduced state $S(\rho_{fast})$. To create the reduced state, we trace out the subsystem $L_s \otimes S \otimes R_s$, obtaining $\rho_{fast} = \mathrm{tr}_{L_s \otimes S \otimes R_s}(|\Psi\rangle\langle\Psi|)$. In Fig. 7, the calculation of the reduced density matrix ρ_{fast} within the TN framework is shown.

If we have indeed obtained a factorized state as in Eq.(12), our state $\rho_{fast} = |\phi^{fast}_{L_f R_f}\rangle\langle\phi^{fast}_{L_f R_f}|$ corresponds to a pure state. Consequently, the entropy $S(\rho_{fast}) = 0$ because we have a product state that is completely disentangled from the subsystem $L_s \otimes S \otimes R_s$. Conversely, if the entropy is not zero, this indicates that some entanglement remains between the subsystems, and the factorization has not achieved a perfect separation.

There are two caveats to this optimization process. The first is that the dimension d_{fast}

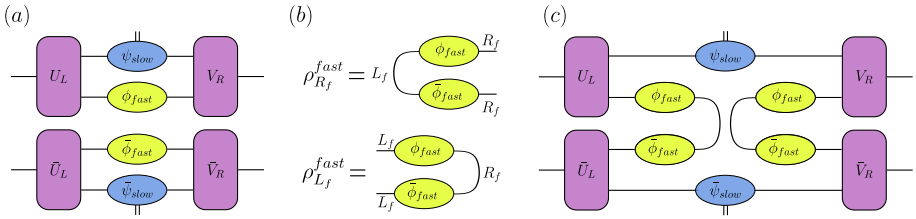

Figure 8: Diagrammatic representation of the simple truncation process. (a) The density matrix of the full system before truncation. (b) The reduced density matrices $\rho^{fast}_{L_f}$ and $\rho^{fast}_{R_f}$. (c) The new density matrix after simple truncation.

for the state $|\phi^{fast}_{L_f R_f}\rangle$ is not known a priori. To find the exact decomposition, one should experiment with different values of d_{fast} and select the best result. The second caveat is that there are infinitely many solutions to the decomposition problem. Numerically, we have observed solutions where, for example, the entanglement between the subsystem $L_f \otimes R_f$ is zero, indicating that the state supposed to contain the long-range entanglement actually retains none. To avoid such cases, we start our optimization process using a maximally entangled state as our fast state:

$$|\phi^{fast}_{L_f R_f}\rangle = \frac{1}{\sqrt{d_{fast}}} \sum_{i=1}^{d_{fast}} |i, i\rangle. \tag{16}$$

This ensures that the initial state $|\phi^{fast}_{L_f R_f}\rangle$ contains maximum entanglement, helping to guide the optimization toward a more meaningful factorization.

3.3 Simple truncation algorithm

Once the factorization is achieved, we obtain the desired state $|\psi^{slow}_{L_s S R_s}\rangle \otimes |\phi^{fast}_{L_f R_f}\rangle$. The next step involves replacing the pure state with a mixed state as described in Eq.(10).

The method proposed by Miguel Frias et al. in Ref.[FPTBn24] involves substituting the density operator of the entire system $|\Psi\rangle\langle\Psi|$, as illustrated in Fig. 8(a), with a new mixed state that leaves the subsystem $L_s \otimes S \otimes R_s$ unchanged while replacing the density operator of the subsystem $L_f \otimes R_f$ with its reduced density matrices $\rho^{fast}_{L_f} \otimes \rho^{fast}_{R_f}$, as depicted in Fig. 8$(b)$. In the tensor network picture, this process translates to keeping the tensors ψ_{slow} unchanged and replacing the tensors ϕ_{fast} with the tensor product of their reduced density matrices, as shown in Fig. 8(c). Then we exchange the original density matrix with the newly found one in all blocks simultaneously, as shown in Fig. 9.

When we have achieved an exact factorization, the newly found mixed state presents the same reduced density matrices as the original system, ensuring the correct expectation values for at least observables acting on l sites. For a more detailed explanation, see Appendix E.

3.4 Heuristic Truncation

To ensure all the desirable properties mentioned above, we rely on achieving a complete decomposition of the fast and slow degrees of freedom. Nevertheless, there is always a residual entropy $S(\rho_{fast})$, indicating imperfect factorization of the fast and slow modes. This results in a systematic error with each truncation, still, the results show that this error does not change the qualitative behavior of the dynamics, as we will show in the following.

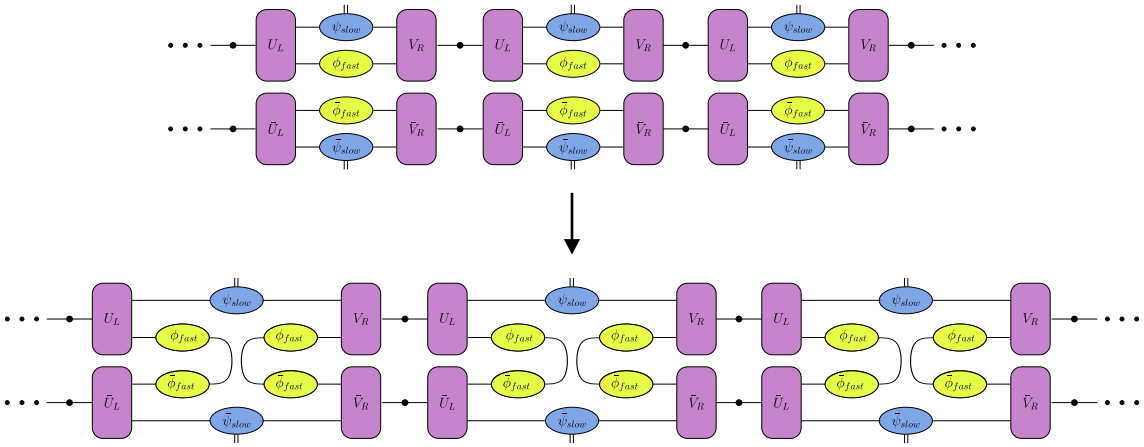

Figure 9: Conversion of the pure state representation of the spin chain to a mixed state by performing simultaneous simple truncation on each block.

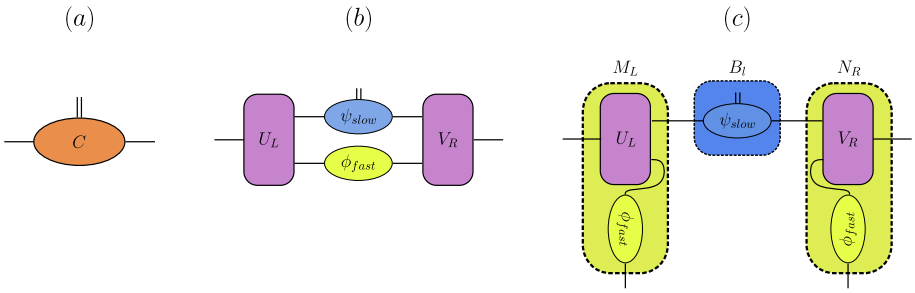

Figure 10: (a) Tensor C from the infinite MPS, represents the initial state before truncations.(b) Pure state representation after the decomposition of the fast and slow degrees of freedom.(c) Ansatz for the heuristic truncation, described how are formed the tensors M_L, B_l and N_R

To mitigate the error due to imperfect factorization between fast and slow degrees of freedom, the authors of Ref. [FPTBn24] proposed a heuristic algorithm. The approach is as follows: first, perform simple truncation, as described in the previous section, until $S(\rho_{fast})$ is below a predetermined threshold ν_s. Once the factorization falls below this threshold and the initial truncation is performed, we use the obtained state as a purification ansatz. We then variationally optimize our state to accurately capture the marginals for the subsystems LS and RS.

The ansatz used consists of three tensors named M_L, B_l, and N_R. They are built from the tensors obtained during the simple truncation, in Fig.10 we depict how these initial tensors are constructed. Then, we optimize them by minimizing the sum of the Euclidean distance between the reduced density matrix of the subsystems $LS(SR)$ from the original state $\rho_{LS(SR)}$ and the truncated state $\tilde{\rho}_{LS(SR)}$. Therefore, the cost function we need to minimize is

$$\min_{M_L,B_l,N_R} \left(||\rho_{LS} - \tilde{\rho}_{LS}||^2 + ||\rho_{SR} - \tilde{\rho}_{SR}||^2 \right), \tag{17}$$

where our optimization variables are the tensors M_L, B_l, and N_R. These tensors form the matrices $\tilde{\rho}$ and include terms that are fourth-order in our optimization tensor, hence it can not be solved using DMRG optimization[CM23]. For this reason, we have used a gradient descent method to minimize the cost function, for further details see Appendix F. The state after this second optimization has the form depicted in Fig. 11.

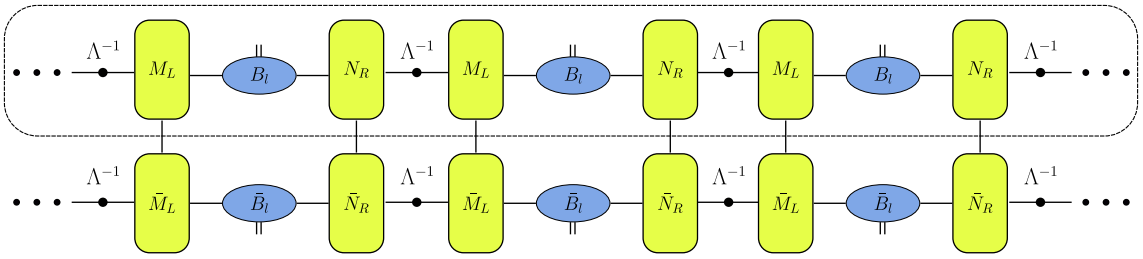

Figure 11: The density matrix for the full state after successfully performing the heuristic truncation. The section highlighted by the dashed rectangle represents the purification MPS form of the state.

3.5 Time evolution

Once we have a new mixed state that accurately captures the local information of our original state, we can proceed with the time evolution. However, starting from this mixed state, it is not immediately clear how to perform the time evolution efficiently.

In the literature, when dealing with mixed states, purification MPS are commonly used [PKS+19]. This technique has been employed in various contexts, such as simulating master equations [ZV04] and finite temperature states [FW05, VGRC04].

Generally, the approach begins by obtaining a Matrix Product Density Operator (MPDO) tensor, which extends the MPS framework from pure states to mixed states [VGRC04]. Subsequently, the purification MPS is recovered by combining the purification and physical legs of the MPDO. The TEBD is then performed by applying the time evolution operator to the physical legs, while the identity operator is applied to the purification legs.

Mathematically, this can be understood as using a purification state to represent the mixed state. For any mixed state $\rho_A \in \mathcal{H}_A$, there exists a pure state $|\Psi\rangle_{AB}$ in an expanded Hilbert space $\mathcal{H}_A \otimes \mathcal{H}_B$, such that tracing out the ancillary system B recovers the original state:

$$\rho_A = Tr_B\{|\Psi\rangle_{AB} \langle\Psi|\}. \tag{18}$$

Thus, when we unify the purification leg with the physical leg, we effectively move to an extended Hilbert space $\mathcal{H}_A \otimes \mathcal{H}_B$. The resulting MPS represents our purified state $|\Psi\rangle_{AB}$.

Regarding time evolution using TEBD, unifying the physical and purification legs is equivalent to working with the MPDO tensor, and applying the time evolution operator only to the physical legs.

To apply this approach in our setup, we first need to create the MPDO for our case. There are several alternatives for constructing the MPDO. As we have stated in the non-heuristic truncation, the mixed state we have obtained correctly captures the local information from the reduced left (right) density matrix $\rho_{L(R)}$ of the system. Additionally, the M_L (N_R) tensor is the only purification tensor involved in creating the left (right) density matrix. This implies that M_L contains the long-range entanglement for the left subsystem, while N_R contains the long-range entanglement for the right subsystem.

Consequently, it appears that both tensors already encapsulate the relevant information in each purification leg. Therefore, we only need to incorporate the local physical information contained in the Bl tensor to each one of them. The most natural way to do this without altering the existing structure of the mixed state is to perform an SVD on Bl. The result of the SVD consists of two isometries, U_A and $V_B^\dagger$, and a diagonal matrix Λ_{AB}. By absorbing the singular values into the isometries, we obtain the tensors $\hat{A}$ and $\hat{B}$, separated by the inverse Λ_{AB}^{-1}.

Finally, we contract the ML and A tensors to yield a new tensor A', which now contains both the purification leg and the physical leg. Similarly, we contract B and NR to produce

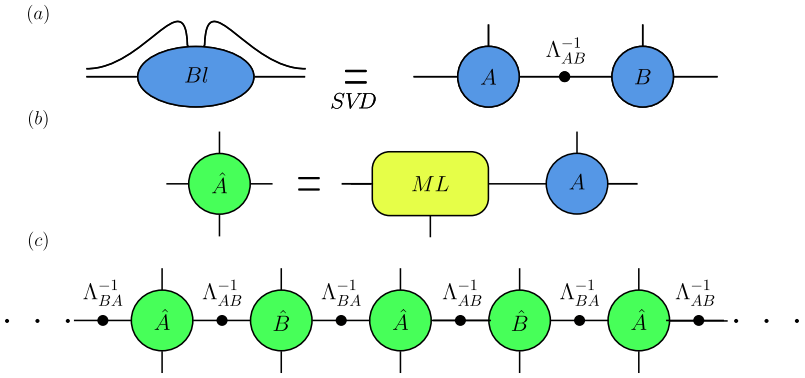

Figure 12: Diagrammatic scheme illustrating the creation of the infinite Matrix Product Density Operator (iMPDO) from the initial mixed state. In panel (a), we show the singular value decomposition of the tensor Bl into two new tensors, A and B, each of which has absorbed the singular values Λ_{AB}. In panel (b), we demonstrate the construction of the new tensor $\hat{A}$, which forms one of the building blocks of the final MPDO. The construction of $\hat{B}$ is analogous, replacing M_L and A with N_R and B, respectively. Panel (c) presents the final iMPDO.

B', which also contains one physical leg and one purification leg. The resulting MPDO is composed of a unit cell containing the tensors $\{\hat{A}, \hat{B}, \Lambda_{AB}^{-1}, \Lambda_{BA}^{-1}\}$. In Fig.12 we show diagrammatically the main steps described above.

Once we have the iMPDO, we can perform the iTEBD. In this case, as we have already stated, the iTEBD is equivalent to the MPS case but in an expanded Hilbert space of dimension $\mathcal{H}_d \otimes \mathcal{H}_{d_{pur}}$. To see the details of how iTEBD is performed refer to Appendix G.1.

After several iterations of the TEBD algorithm, we will again factorize our evolved state's slow and fast degrees of freedom. When the entropy $S(\rho_{fast})$ of our factorized state falls below a predefined threshold ν_s, we will start the gradient descent optimization to ensure that the local information on both the left and right sides is accurately captured.

However, to apply the slow and fast decomposition, we need to revert to the original structure with the tensors ML, NR, and Bl to isolate and factorize the relevant local information encapsulated in the Bl tensor. To accomplish this, we must transform our iMPDO unit cell from the form $\hat{A}, \hat{B}, \Lambda_{AB}, \Lambda_{BA}$ back to the desired structure. The simplest approach is to reshape $\hat{A}$ and perform an SVD to separate the left bond dimension and the purification leg from the right bond dimension and the physical leg. In the final step of the SVD, we absorb the singular values Λ_A into the isometry $V_A^\dagger$ to form the new tensor A, which now only contains the physical leg and not the purification leg. The isometry U_A becomes the desired tensor M_L. Using the same method on $\hat{B}$, we can obtain a new tensor B and recover the desired purification tensor N_R. Lastly, we can contract the new tensors A and B with Λ_{AB}^{-1} to yield the last tensor we needed, the B_l tensor. The steps are summarized in Fig. 13. Once the state has been successfully factorized and its entropy is below the chosen threshold ν_s, we have the original M_L and N_R tensors, plus the new tensors obtained from the decomposition, named M_L', N_R', and B_l. We absorb the new M_L' and N_R' tensors into the original ones, causing the purification legs to grow exponentially after each truncation. To address this issue, we introduce a cut-off after a few truncations, setting a maximum dimension $d_{\text{pur}} = 32$. The truncation is performed on the iMPDO by doing an SVD between the purification leg and the left ones, keeping only the largest singular values to achieve the desired dimension. This process is illustrated schematically in Fig. 14.

The described algorithm, which we will refer to as iMPDO$_{v1}$, works well for shorter

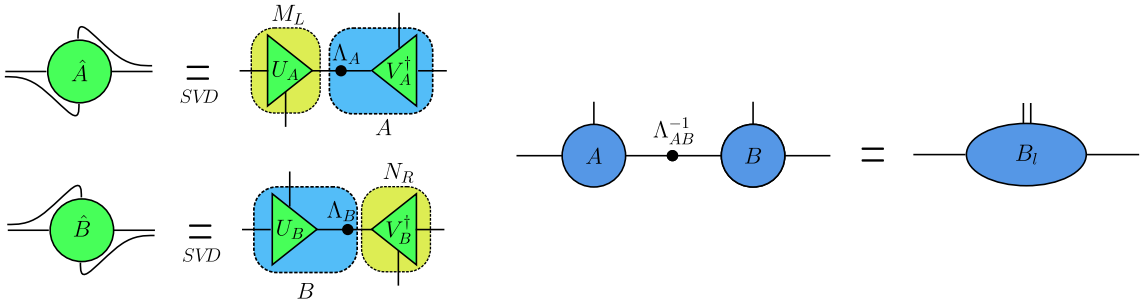

Figure 13: Diagrammatic scheme illustrating the procedure to recover the initial unit cell with the tensors M_L,N_R, and B_l from our iMPDO.

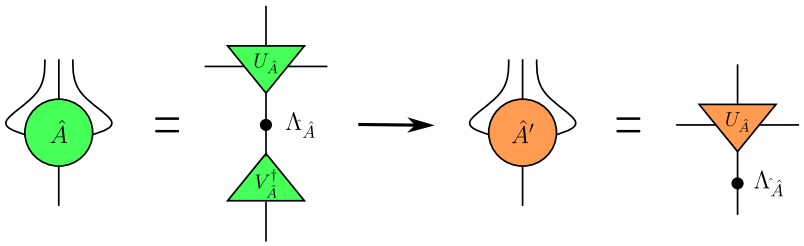

Figure 14: Diagrammatic scheme illustrating the procedure to truncate the purification legs from our iMPDO. The isometry $V_{\hat{A}}^\dagger$ is discarded because the purification leg will always be contracted with its complex conjugate, ensuring that $V_{\hat{A}}^\dagger V_{\hat{A}} = \mathbb{1}$. This redundancy make $V_{\hat{A}}^\dagger$ unnecessary for the purification leg. The same procedure is also applied to the tensor $\hat{B}$ of our iMPDO.

times; however, several aspects suggest that it is not the optimal method to fully leverage the factorization algorithm's advantages. One key issue is that when we create the iMPDO, as shown in Fig. 12, we contract the tensors $M_L(N_R)$ and $A(B)$ to form $\hat{A}(\hat{B})$. This approach does not utilize the reduced bond dimension obtained from the factorization algorithm during the initial step of the TEBD. As a result, the first SVD becomes very computationally expensive and does not provide significant benefits over the standard TEBD evolution. Additionally, numerical implementation shows that the larger bond dimension grows exponentially. To address this issue, we have developed a second method for the iTEBD that aims to utilize the reduced bond dimension better, which we will refer to as iMPDO$_{v2}$ in the following.

The key difference between this new method and the previous one lies in how we perform the first iTEBD step. In the new approach, we start by reshaping the M_L tensor, combining the left leg with the purification leg. We then perform an SVD, resulting in two isometries, U_L and $V_L^\dagger$, and the singular values Λ_L. The isometry U_L becomes our new purification tensor M_L', while the remaining tensors are absorbed into the B_l tensor. Subsequently, we apply the same procedure to the N_R tensor. The resulting tensor B_l' encapsulates all the entanglement from our physical state, ensuring that the truncation minimizes the global error. The schematic representation of this method is illustrated in Fig. 15.

Then, the first time evolution operator U_1 is directly applied to the new B_l' tensor, ensuring that we are taking advantage of the reduced bond on the factorization. The rest of the iTEBD is performed similarly to the first method. For a more detailed explanation, we refer to Appendix G.2

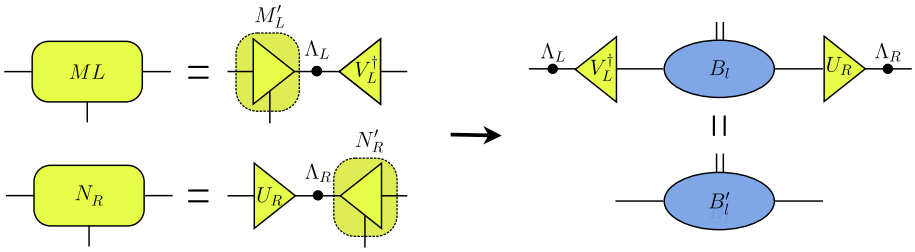

Figure 15: Diagrammatic scheme illustrating the procedure for obtaining the new tensor B'_l, designed to minimize global error during truncation in the first iTEBD step.

4 Numerical results

In this section, we benchmark the algorithm described in the previous section using the Transverse Field Ising Model (TFIM), governed by the Hamiltonian

$$H = -\sum_i \left(J\sigma_i^z \sigma_{i+1}^z + g\sigma_i^x \right), \tag{19}$$

where σ_i^z and σ_i^x are Pauli matrices acting on the ith site, J determines the interaction strength between neighboring spins, and g represents the strength of the external transverse magnetic field.

The initial state used in our simulation is a product state $|X\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$, which corresponds to the ground state in the deep paramagnetic phase $g \to \infty$.

We selected the Transverse Field Ising Model for benchmarking the algorithm for two primary reasons. First, it is one of the rare models with an analytical solution, and this solution explicitly involves quasiparticles. This makes the TFIM an ideal system for testing the underlying concepts of our algorithm, particularly those related to quasiparticle dynamics. Second, the TFIM was used by Frias et al. to benchmark their algorithm, allowing us to directly compare our results with theirs.

For the simulations, we employed the numpy and scipy libraries for various linear algebra tasks, including SVDs and diagonalizing Hermitian matrices. Tensor contractions were handled using the ncon package. However, all core routines and more advanced methods, such as bringing tensors into canonical form and performing the iTEBD algorithm, were implemented from scratch. Additionally, we utilized the jax Python package for gradient descent, taking advantage of its automatic differentiation features to efficiently optimize the system. The complete code for these simulations is available in the GitHub repository cited in Ref. [Far24].

4.1 Factorization results

We began by analyzing the results from the factorization of the system into fast and slow modes. To assess the effectiveness of this factorization, we simulated a standard TEBD time evolution, applying the factorization procedure described in Section 3.2 at each time step. For the factorization, we set the dimension $d_{fast} = 2$, consistent with the dimension used in the time evolution. If the bond dimension of our MPS was not a multiple of two, we skipped the factorization for that time step and proceeded to the next.

Our initial study involved starting with random tensors U_L, V_R, ψ_{slow}, and ϕ_{fast}. After each factorization, we computed the von Neumann entropy $S(\rho_{fast})$ to quantify the entanglement in the fast modes. Additionally, we applied an SVD to the tensor ψ_{slow} and examined the quantity $1 - \lambda_{max}$, where λ_{max} is the largest singular value. This was done to determine whether the state retained long-range entanglement, as expected.

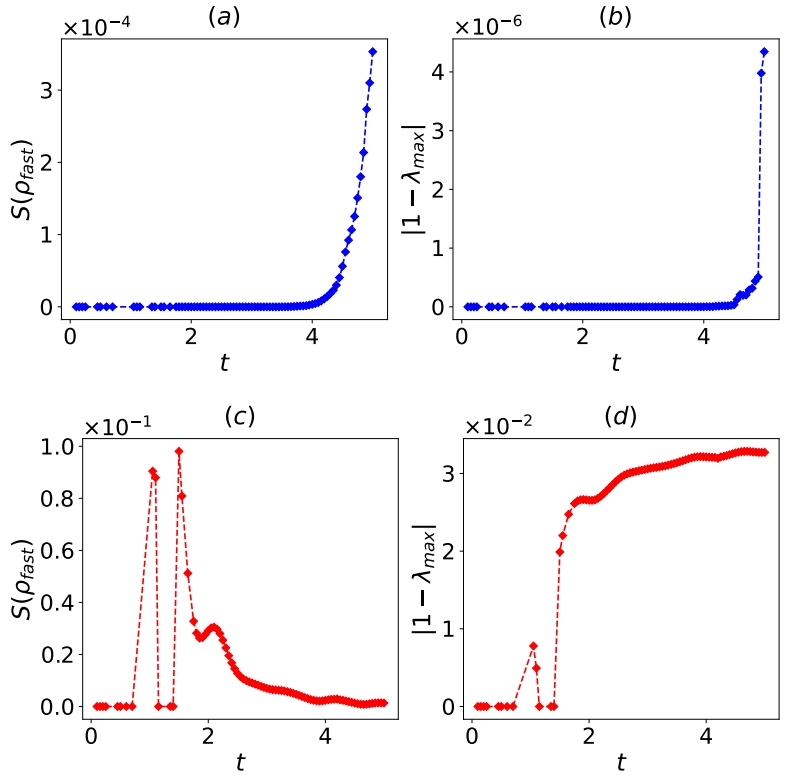

Figure 16: (a,b) Results from the optimization without initializing the fast mode as a maximally entangled state. (a) Shows the von Neumann entropy of the reduced density matrix $S(\rho_{fast})$. (b) Shows the proximity of the maximum singular value of the state ϕ_{fast} to one. (c-d) Results from the optimization starting with the fast mode as a maximally entangled state. (c) Shows the von Neumann entropy of the reduced density matrix $S(\rho_{fast})$. (d) Shows the proximity of the maximum singular value of the state ϕ_{fast} to one.

We then compared these results with those obtained by initializing the factorization with a maximally entangled state for the fast mode. In this scenario, we allowed several iterations of factorization with the fast tensor fixed as the maximally entangled state before enabling the standard optimization. The results are shown in Fig.16.

As observed, when we do not start with a maximally entangled state in the fast component, we achieve a perfect factorization where the largest singular value between the cut $L_f R_f$ is one. This outcome indicates that the subsystems L_f and R_f are not entangled, which is the opposite of our intended goal. However, when the optimization is constrained to start with a maximally entangled state, the maximum singular value begins to deviate significantly from 1 over time, indicating the presence of long-range entanglement. Correspondingly, the entropy $S(\rho_{fast})$ also deviates from 0, indicating that perfect factorization is not achieved.

It is important to note that the factorization process often gets stuck in local minima. To mitigate this, we typically repeat the factorization multiple times with different initial conditions at the same time step and retain the best result. Another effective strategy is to start the optimization from the tensors obtained in the previous time step rather than from a random state. However, it is advisable to combine this approach with some factorizations from random tensors to better explore the optimization landscape and avoid biased results.

Next, we investigated the performance of the factorization algorithm across different

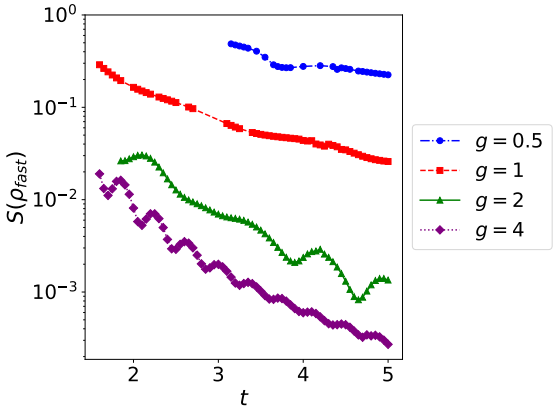

Figure 17: The von Neumann entropy of the fast degrees of freedom $S(\rho_{fast})$ found by the factorization algorithm as a function of time t for the Transverse Ising model.

values of g, corresponding to various phases of the Transverse Ising model. For this optimization, we employed the variant where the factorization begins with a maximally entangled state. We focused on the behavior after $t \approx 2$, the time at which the factorization begins to identify fast modes with long-range entanglement.

As illustrated in Fig.17, the efficiency of the factorization strongly depends on the specific system under consideration. In all cases, the entropy decreases exponentially over time, indicating that the factorization effectively captures the quasi-particle (QP) dynamics. As time progresses, the number of QPs that completely disentangle from the subsystem S increases. However, perfect disentanglement is not achieved in most cases, although the results consistently improve as the parameter g increases.

The improvement in factorization with increasing g is closely related to the intrinsic behavior of the system, particularly the distribution of QPs in the initial state. As studied in Ref. [FPTBn24], the velocity group and occupation number of QPs vary depending on its momentum k in the Transverse Ising model. In the state under consideration, for larger values of g, the maximum occupation number aligns with modes that have the highest velocity group, whereas for smaller g, it coincides with modes of zero velocity group. This suggests that for larger g, the QPs radiate more effectively and move faster, leading to quicker disentanglement from the subsystem S. These findings indicate that the algorithm's effectiveness is strongly dependent on the specific system and its intrinsic dynamics.

Another factor affecting the algorithm's effectiveness is the complexity of the optimization landscape, which can easily trap the factorization process in local minima. The goal of the factorization is to maximize the overlap between the original state and the desired factorized state. However, there is no assurance that this factorization will result in a clear separation between fast and slow modes. As seen in cases where the factorization does not begin with a maximally entangled state, alternative factorizations with similar forms can exist, which may not align with the intended separation of modes.

4.2 Characterization of the algorithm: time evolution

First, let's discuss the results obtained using the simple truncation algorithm, which relies solely on the factorization of fast and slow modes without employing gradient descent optimization. In this study, we simulated the Transverse Ising model with $g = 4$ and compared the results against those from the standard iTEBD algorithm. For the simulation,

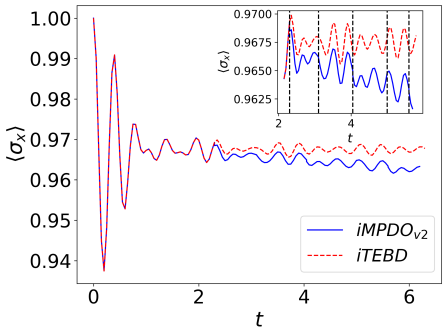

Figure 18: Transverse magnetization σ_x as a function of time t for the Transverse Ising model with $J = 1$ and $g = 4$ without gradient descent optimization. The red dashed line corresponds to the standard iTEBD, and the blue continuous line corresponds to iMPDO$_{v2}$. The inset provides a more detailed view of the truncations, which are marked by the horizontal dashed lines.

we set an entropy threshold of $\nu_s = 0.008$. After each truncation, we performed at least 10 time steps with $\delta_t = 0.05$, retaining only the singular values greater than $\epsilon > 10^{-5}$, before reapplying the factorization.

The results, shown in Fig. 18, reveal that each truncation introduces a systematic error. However, the qualitative behavior of the expectation value is preserved, indicating that the mixed state effectively captures the same expectation values as the pure state.

Moreover, we observed that reducing the entropy threshold ν_s diminishes the systematic error, highlighting that this error is closely linked to the degree of disentanglement between the fast and slow modes. As the threshold decreases, the approximation becomes more accurate, leading to improved results in the simulation.

We now proceed to characterize the algorithm, with a primary focus on the iMPDO$_{\text{v2}}$ method. Our analysis shows that the results obtained at short times are consistent across both iMPDO$_{\text{v1}}$ and iMPDO$_{\text{v2}}$ methods, with the bond dimension behavior being nearly identical. However, the computational complexity scales differently between the two.

In the iMPDO$_{\text{v1}}$ method, the most computationally expensive operation is the SVD performed after applying the U_1 operator. The complexity of this step is $O((D_{\text{big}} d_{\text{fast}} d)^3)$, where D_{big} is the bond dimension that remains unaffected by factorization, d_{fast} is the dimension that increases up to a predefined value (16 in our short-time simulations), and d is the physical dimension, which is 2 in this case. In contrast, the iMPDO$_{\text{v2}}$ method's most expensive operation is the SVD performed after applying the U_2 operator, with a complexity of $O((D_{\text{small}} d_{\text{fast}} d)^3)$, where D_{small} is the bond dimension reduced through factorization.

To provide a comparative perspective, we have also included results from the iMPDO$_{\text{v1}}$ method in Fig. 19. These results demonstrate similar behavior, albeit with simulation times that are two to three times longer than those required by the improved algorithm. As anticipated, for short times, iMPDO$_{\text{v1}}$ recovers the same solutions as the standard iTEBD. The bond dimension behavior aligns with expectations: D_{big} exhibits the same behavior as the bond dimension in iTEBD, while D_{small}, due to the factorization, is progressively reduced, leading to a slower increase over time.

To thoroughly evaluate the features of the algorithm, we applied the iMPDO$_{\text{v2}}$ algorithm with various entropy thresholds ν_s during the factorization and different cost function thresholds f_{cost} for the gradient descent. These were then compared with a standard iTEBD for the Transverse Ising model with parameters $J = 1$ and $g = 4$. In all cases, we retained only the singular values from the TEBD truncation that were greater than 10^{-5},

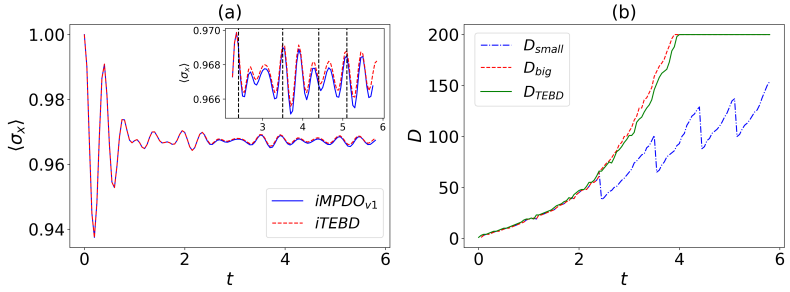

Figure 19: Time evolution of the Transverse Ising model with $J = 1$ and $g = 4$ using the iMPDO$_{v1}$ time evolution. (a) Transverse magnetization σ_x as a function of time t. The red dashed line corresponds to the standard iTEBD, and the blue continuous line corresponds to iMPDO$_{v1}$. The inset provides a more detailed view of the truncations, which are marked by the vertical dashed lines. (b) Bond dimension D as a function of time. The dashed blue line corresponds to the bond dimension reduced by the factorization D_{small}, the continuous green line corresponds to the bond dimension unaffected by the factorization D_{big}, and the red dashed line represents the bond dimension in the standard iTEBD algorithm D_{iTEBD}.

with a maximum bond dimension of $D = 200$. Figure 20(a) presents the evolution of the expected value σ_x. The simulation results systematically improve as the threshold f_{cost} is reduced, although this leads to longer intervals between truncations and consequently, the bond dimension tends to increase. In Fig. 20(b), the different values of the cost function are shown at each factorization step, after the gradient descent. Two key points emerge: first, when the spectrum of the cost function is similar, the simulation results are identical, regardless of the starting entropy thresholds ν_s. This suggests that the behavior of the simulation is primarily dictated by the cost function values during the heuristic truncation. However, setting an overly high entropy threshold combined with a very low gradient descent threshold is suboptimal, as it leads to unnecessary gradient descent optimizations over many time steps, significantly increasing the simulation time.

Second, there are instances where the cost function yields values as low as 10^{-11}. In such cases, after factorization, the reduced bond dimension D_{small} quickly returns to its pre-factorization value after evolving one time step $\delta_t = 0.05$. This indicates that the fast state identified in this factorization may not contain any meaningful entanglement, rendering it ineffective.

Figures 20(c) and 20(d) illustrate the behavior of the non-reduced bond dimension D_{big} and the reduced bond dimension D_{small}, respectively. As with iMPDO$_{v1}$, the non-reduced bond dimension D_{big} mirrors the behavior seen in the standard iTEBD case, while the reduced bond dimension D_{small} increases more slowly during the evolution. However, it appears that after the factorization, D_{small} experiences a significant increase during the first time evolution step δ_t, followed by a slower growth. This initial jump could suggest a potential issue in the method, but when compared to cases where the cost function was extremely small and the bond quickly recovered its initial value, it seems that these jumps are intrinsically linked to the degree of entanglement captured by the fast tensors. The more entanglement that is captured, the smaller the jumps; conversely, if less entanglement is captured, the jumps are larger, potentially restoring the original bond dimension.

5 Conclusion

In this work, we have reviewed and characterized an algorithm designed to overcome the entanglement barrier in the simulation of time evolution in out-of-equilibrium quantum

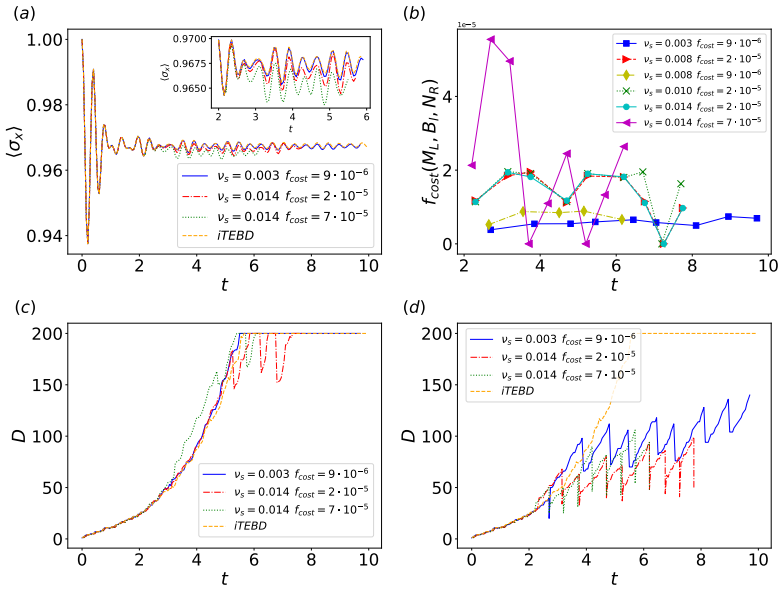

Figure 20: Time evolution of the Transverse Ising model with $J = 1$ and $g = 4$ using the iMPDO$_{v2}$ time evolution for different values of the entropy threshold ν_s and the cost function threshold during the gradient descent f_{cost}. (a) Transverse magnetization σ_x as a function of time t. The inset provides a more detailed view of the expectation values during the truncations. (b) Cost function values at each truncation after the gradient descent. (c) Non-reduced bond dimension D_{big} as a function of time. (d) Reduced bond dimension D_{small} as a function of time. The jumps correspond to the times at which the factorization is made.

systems. The core idea of this algorithm is to mitigate the effects of long-range entanglement by replacing the system's coherence with a mixed state representation that retains the necessary local information for computing the evolution of local expectation values.

The algorithm draws its inspiration from the QP picture, aiming to identify and separate the contributions of fast modes, which are responsible for propagating long-range entanglement across the system. By replacing these fast modes with their corresponding reduced density matrices, the algorithm simplifies the representation of the state, focusing on the essential degrees of freedom required for accurate time evolution. Once the factorization is achieved, the state is evolved, allowing the modes to propagate until it becomes possible to factorize the state again. We proposed two methods for evolving this mixed state, each designed to exploit the factorization to improve the efficiency of the time evolution process.

We tested the algorithm against the Transverse Ising model to evaluate its behavior and efficiency. The algorithm demonstrates good performance at short times, successfully capturing the key features of the system's evolution. However, as time progresses, certain challenges emerge that suggest areas for further improvement.

A primary observation is that the major bottleneck of the algorithm is its computational time. Although the factorization reduces the working bond dimension, the gradient descent steps required to achieve this add significant computational overhead, accounting for more than two-thirds of the total computing time. In our tests, we were able to simulate up to $T/J \approx 12$ over 24 hours on a single GPU-accelerated node in the MareNostrum5 supercomputer by imposing a cutoff $D = 200$ on the bond dimension. Extending this cutoff to larger dimensions and further optimizing the gradient descent parameters could potentially extend the algorithm's applicability to longer timescales.

Another important observation is that, although it does not directly impact the time evolution step, the non-reduced bond dimension increases at the same rate as in the standard iTEBD algorithm. Ideally, a more effective method would not only control the growth of the reduced bond dimension but also mitigate the increase in the non-reduced bond dimension. One approach we considered, but did not have the opportunity to fully explore, involves using the iMPDO$_{\text{v1}}$ algorithm while alternating the bonds where the factorization is performed. This strategy could potentially achieve a more balanced reduction in both bond dimensions.

Additionally, in this study, the purification leg of the state was left untouched during the time of evolution. However, the purification leg offers additional gauge freedom that could be exploited to search for a purification basis that further reduces the system's entanglement. This approach has been explored in other works, such as in Ref. [HLB+18], and could be an avenue for future improvement.

Another area of potential enhancement is the factorization process itself. In this work, the decomposition was used to provide an initial ansatz for the gradient descent optimization. One possible improvement could be to integrate the factorization and optimization processes into a single, more cohesive method. For example, cost functions that lend themselves to DMRG optimization, which is naturally suited for tensor networks, could be explored. Alternatively, the gradient descent process could be enhanced using more advanced optimization algorithms, such as ADAM [KB17] or L-BFGS-B [ZBLN97], which are known to perform better than simple gradient descent in many cases.

Finally, after optimizing and successfully characterizing the algorithm for long-time evolution in the transverse Ising model, the next natural step would be to test its performance on non-integrable models. In these systems, the simple picture of quasiparticles freely traveling through the system no longer applies, making it an ideal scenario to evaluate the algorithm's adaptability and robustness in more complex settings.

Bibliography

[Bañ23] M. C. Bañuls. Tensor network algorithms: A route map. *Annual Review of Condensed Matter Physics*, 14(Volume 14, 2023):173–191, 2023.

[BnHVC09] M. C. Bañuls, M. B. Hastings, F. Verstraete, and J. I. Cirac. Matrix product states for dynamical simulation of infinite chains. *Phys. Rev. Lett.*, 102:240603, Jun 2009.

[CC05] P. Calabrese and J. Cardy. Evolution of entanglement entropy in one-dimensional systems. *Journal of Statistical Mechanics: Theory and Experiment*, 2005(04):P04010, apr 2005.

[CM23] G. Catarina and B. Murta. Density-matrix renormalization group: a pedagogical introduction. *The European Physical Journal B*, 96(8), August 2023.

[CPGSV21] J. Ignacio C., D. Pérez-García, N. Schuch, and F. Verstraete. Matrix product states and projected entangled pair states: Concepts, symmetries, theorems. *Rev. Mod. Phys.*, 93:045003, Dec 2021.

[ECP10] J. Eisert, M. Cramer, and M. B. Plenio. Colloquium: Area laws for the entanglement entropy. *Rev. Mod. Phys.*, 82:277–306, Feb 2010.

[EV11] G. Evenbly and G. Vidal. Tensor network states and geometry. *Journal of Statistical Physics*, 145(4):891–918, June 2011.

[Far24] M. Farreras. Tn_factorization_algorithm. https://github.com/Marc-Farreras/TN_Factorization_Algorithm, 2024. Accessed: 2024-08-23.

[FPBn22] M. Frías-Pérez and M. C. Bañuls. Light cone tensor network and time evolution. *Phys. Rev. B*, 106:115117, Sep 2022.

[FPTBn24] M. Frías-Pérez, L. Tagliacozzo, and M. C. Bañuls. Converting long-range entanglement into mixture: Tensor-network approach to local equilibration. *Phys. Rev. Lett.*, 132:100402, Mar 2024.

[FW05] A. E. Feiguin and S. R. White. Finite-temperature density matrix renormalization using an enlarged hilbert space. *Phys. Rev. B*, 72:220401, Dec 2005.

[HLB+18] J. Hauschild, E. Leviatan, J. H. Bardarson, E. Altman, M. P. Zaletel, and F. Pollmann. Finding purifications with minimal entanglement. *Phys. Rev. B*, 98:235163, Dec 2018.

[KB17] Diederik P. K. and J. Ba. Adam: A method for stochastic optimization, 2017.

[KL80] V. Klema and A. Laub. The singular value decomposition: Its computation and some applications. *IEEE Transactions on Automatic Control*, 25(2):164–176, 1980.

[KRB+18] T. Kraft, C. Ritz, N. Brunner, M. Huber, and O. Gühne. Characterizing genuine multilevel entanglement. *Physical Review Letters*, 120(6), February 2018.

[LC08] A. M Läuchli and C.Kollath. Spreading of correlations and entanglement after a quench in the one-dimensional bose–hubbard model. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(05):P05018, may 2008.

[Mit18] A. Mitra. Quantum quench dynamics. *Annual Review of Condensed Matter Physics*, 9(Volume 9, 2018):245–259, 2018.

[OV08] R. Orús and G. Vidal. Infinite time-evolving block decimation algorithm beyond unitary evolution. *Phys. Rev. B*, 78:155117, Oct 2008.

[PKS+19] S. Paeckel, T. Köhler, A. Swoboda, Salvatore R. M., U. Schollwöck, and C. Hubig. Time-evolution methods for matrix-product states. *Annals of Physics*, 411:167998, 2019.

[PSSV11] A. Polkovnikov, K. Sengupta, A. Silva, and M. Vengalattore. Colloquium: Nonequilibrium dynamics of closed interacting quantum systems. *Rev. Mod. Phys.*, 83:863–883, Aug 2011.

[Sch11] U. Schollwöck. The density-matrix renormalization group in the age of matrix product states. *Annals of Physics*, 326(1):96–192, 2011. January 2011 Special Issue.

[SLRD13] J. Schachenmayer, B. P. Lanyon, C. F. Roos, and A. J. Daley. Entanglement growth in quench dynamics with variable range interactions. *Phys. Rev. X*, 3:031015, Sep 2013.

[SPT19] J. Surace, M. Piani, and L. Tagliacozzo. Simulating the out-of-equilibrium dynamics of local observables by trading entanglement for mixture. *Phys. Rev. B*, 99:235115, Jun 2019.

[Suz85] M. Suzuki. Decomposition formulas of exponential operators and Lie exponentials with some applications to quantum mechanics and statistical physics. *Journal of Mathematical Physics*, 26(4):601–612, 04 1985.

[TCF+12] S. Trotzky, Y.-A. Chen, A. Flesch, I. P. McCulloch, U. Schollwöck, J. Eisert, and I. Bloch. Probing the relaxation towards equilibrium in an isolated strongly correlated one-dimensional bose gas. *Nature Physics*, 8(4):325–330, 2012.

[VGRC04] F. Verstraete, J. J. García-Ripoll, and J. I. Cirac. Matrix product density operators: Simulation of finite-temperature and dissipative systems. *Phys. Rev. Lett.*, 93:207204, Nov 2004.

[VHV19] L. Vanderstraeten, J. Haegeman, and F. Verstraete. Tangent-space methods for uniform matrix product states. *SciPost Phys. Lect. Notes*, page 7, 2019.

[Vid03] G. Vidal. Efficient classical simulation of slightly entangled quantum computations. *Phys. Rev. Lett.*, 91:147902, Oct 2003.

[Vid07] G. Vidal. Classical simulation of infinite-size quantum lattice systems in one spatial dimension. *Phys. Rev. Lett.*, 98:070201, Feb 2007.

[WF04] S. R. White and A. E. Feiguin. Real-time evolution using the density matrix renormalization group. *Phys. Rev. Lett.*, 93:076401, Aug 2004.

[Whi92] S. R. White. Density matrix formulation for quantum renormalization groups. *Phys. Rev. Lett.*, 69:2863–2866, Nov 1992.

[Whi93] S. R. White. Density-matrix algorithms for quantum renormalization groups. *Phys. Rev. B*, 48:10345–10356, Oct 1993.

[Wil13] M. M. Wilde. *Quantum Information Theory*. Cambridge University Press, 2013. See Property 9.1.6, p. 252.

[WZMR18] C. D. White, M. Zaletel, R. S. K. Mong, and G. Refael. Quantum dynamics of thermalizing systems. *Phys. Rev. B*, 97:035127, Jan 2018.

[ZBLN97] C. Zhu, R. H. Byrd, P. Lu, and J. Nocedal. Algorithm 778: L-bfgs-b: Fortran subroutines for large-scale bound-constrained optimization. *ACM Trans. Math. Softw.*, 23(4):550–560, dec 1997.

[ZV04] M. Zwolak and G. Vidal. Mixed-state dynamics in one-dimensional quantum lattice systems: A time-dependent superoperator renormalization algorithm. *Phys. Rev. Lett.*, 93:207205, Nov 2004.

A Notation II

While the main text covers the foundational concepts, this appendix aims to deepen your understanding of diagrammatic notation in tensor networks.

We begin by revisiting tensor contraction in more detail, along with some common operations that frequently appear in tensor calculus, algebra, and quantum mechanics.

Tensor contraction is the cornerstone of almost all tensor network operations. It involves multiplying the coefficients of tensors and summing over the contracted indices. For example, consider two order-3 tensors, $A_{i,j,k}$ and $B_{l,m,n}$. If we contract index j with m and k with n, the result is a new order-2 tensor $F_{i,l}$, with coefficients given by:

$$F_{i,l} = \sum_{\alpha,\beta} A_{i,\alpha,\beta} B_{l,\alpha,\beta}. \tag{20}$$

As discussed in the main text, tensor contraction is visually represented by connecting the legs of the tensors. With an understanding of tensor contraction, one can easily perform standard operations such as the scalar product, matrix-vector multiplication, matrix-matrix multiplication, and matrix trace. Figure 21 illustrates the diagrammatic representations of these operations.

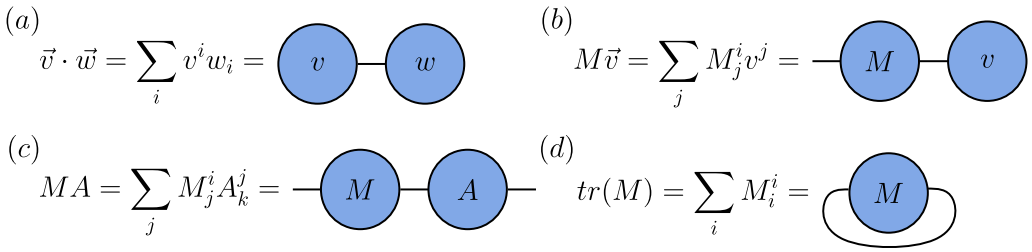

Figure 21: Tensor network diagrammatic representation of (a) scalar product between two vectors, (b) matrix-vector multiplication, (c) matrix-matrix multiplication, and (d) trace of a matrix.

Another fundamental operation in quantum mechanics is the tensor product. The tensor product of two tensors is performed by taking the element-wise product of their coefficients. Specifically, if we have a tensor A of order r and a tensor B of order m, the tensor product $A \otimes B$ results in a tensor of order $r + m$, with coefficients given by

$$(A \otimes B)_{i_1,\ldots,i_r,j_1,\ldots,j_m} = A_{i_1,\ldots,i_r} \cdot B_{j_1,\ldots,j_m}, \tag{21}$$

where the multiplication is the standard element-wise multiplication of the coefficients. In diagrammatic notation, the tensor product is represented by placing the tensors side by side, as shown in Fig.22(a).

Finally, we delve into how diagrammatic notation can make abstract concepts, which may initially seem complex in tensor notation, more intuitive. A particularly relevant example is the concept of isomorphisms in tensor networks.

An isomorphism is a mapping that preserves the structure and relationships between elements across different mathematical spaces. In the context of tensors, isomorphisms naturally arise. For example, the tensor spaces $\mathbb{C}^{d_1 \times \cdots \times d_n}$ and $\mathbb{C}^{\tilde{d}_1 \times \cdots \times \tilde{d}_n}$ are isomorphic whenever their overall dimensions match, i.e., $\prod_i d_i = \prod_i \tilde{d}_i$. This implies that a tensor can be reshaped or reorganized into different forms to suit the needs of a particular analysis, while preserving the original information it encodes.

Consider a tensor T of order 6. If we want to reshape this tensor into a matrix by grouping the first three indices together and the remaining three into another group, the

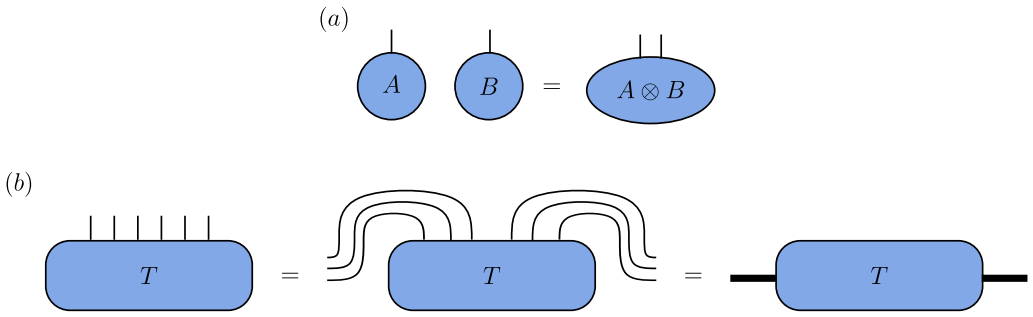

Figure 22: Diagrammatic representation of tensor network operations: (a) Tensor product of two tensors A and B, and (b) Reshaping of a tensor T of order 6 into a tensor of order 2 by grouping the first three legs into one larger leg and the remaining three into a second larger leg, indicating the increased dimensionality of the new legs.

transformation can be defined as follows

$$T_{i_1,i_2,i_3,i_4,i_5,i_6} \Leftrightarrow T_{I_1,I_2}, \tag{22}$$

where the new indices I_1 and I_2 are defined by

$$\begin{aligned} I_1 &= i_1 + (i_2 - 1)d_1 + (i_3 - 1)d_1 d_2, \\ I_2 &= i_4 + (i_5 - 1)d_4 + (i_6 - 1)d_4 d_5, \end{aligned} \tag{23}$$

with d_x representing the dimension associated with each index. The number of elements in I_1 and I_2 is $d_1 d_2 d_3$ and $d_4 d_5 d_6$, respectively, as expected. This reshaping operation is commonly implemented in computational libraries like NumPy. Diagrammatically, this operation is represented by grouping legs to reduce the tensor's order or by splitting legs to increase it. Figure 22(b) provides a diagrammatic representation of the grouping operation described by Eq.(22).

B Singular Value Decomposition and Schmidt Decomposition

The Singular Value Decomposition (SVD) is one of the most fundamental tools in linear algebra, widely used across various fields, including computer science and quantum information science, with significant applications in tensor networks. Given its importance and use in this work, we will briefly review some key aspects.

The SVD of an arbitrary rectangular matrix M of dimensions $n \times m$ is a factorization of the form:

$$M = USV^\dagger, \tag{24}$$

where U is a left-isometry (an isometric matrix) of dimensions $n \times \min(n, m)$ that satisfies $U^\dagger U = \mathbb{1}$, S is a non-negative diagonal matrix of dimensions $\min(n, m) \times \min(n, m)$ with non-negative entries, and $V^\dagger$ is a right-isometry of dimensions $\min(n, m) \times m$ that satisfies $VV^\dagger = \mathbb{1}$. The non-zero diagonal entries of S are known as the singular values of M, and their number is called the Schmidt rank.

In the context of tensor networks, one of the most important properties of SVD is that it provides the optimal approximation of a matrix M of rank r by a lower rank matrix $\tilde{M}$ with rank $\tilde{r} < r$ in the Frobenius norm $||M||_F^2 = \sum_{i,j} |M_{ij}|^2 = \mathrm{Tr}(M^\dagger M)$ [EY36]. This optimal approximation is achieved by retaining only the $\tilde{r}$ largest singular values in the

SVD of M. The approximation error ϵ is given by the sum of the squares of the discarded singular values

$$||M - \tilde{M}||_F = \sqrt{\sum_{i=\tilde{r}+1}^{r} s_i(M)}, \tag{25}$$

where $s_i(M)$ denotes the singular values of the matrix M.

Another key feature of SVD arises when it is applied to a bipartite quantum state $|\Psi_{AB}\rangle = \sum_{ij} C_{ij}|i\rangle_A|j\rangle_B$. The result is the Schmidt decomposition, which expresses the state in the form

$$|\Psi_{AB}\rangle = \sum_{k}^{\min(d_A,d_B)} s_k|a_k\rangle_A|b_k\rangle_B, \tag{26}$$

where d_A and d_B are the dimensions of the subsystems A and B, respectively. Here, s_k are the Schmidt coefficients (the singular values of the matrix C formed by the bipartite state coefficients C_{ij}), and $\{|a_k\rangle_A\}$ and $\{|b_k\rangle_B\}$ are the corresponding orthonormal Schmidt basis vectors.

The Schmidt decomposition has several important properties. For instance, the Schmidt coefficients can be used to compute the entanglement entropy of the reduced density matrices $\rho_{A(B)} = \mathrm{Tr}_{B(A)}(|\Psi_{AB}\rangle\langle\Psi_{AB}|)$, which quantifies the entanglement between the subsystems. Of particular relevance to us is the fact that, due to the orthonormality of the Schmidt basis, the vector 2-norm of $|\Psi_{AB}\rangle$ is identical to the Frobenius norm of the coefficient matrix C:

$$|||\Psi_{AB}\rangle||^2 = \sum_{i,j} ||C_{ij}||^2 = ||C||_F^2. \tag{27}$$

As a consequence, the best approximation $|\tilde{\Psi}_{AB}\rangle$ of the state $|\Psi_{AB}\rangle$ in the 2-norm is equivalent to the best low-rank approximation $\tilde{C}$ of the matrix C in the Frobenius norm. Therefore, the optimal approximation $|\tilde{\Psi}_{AB}\rangle$ is obtained by retaining only the $\tilde{r}$ largest Schmidt coefficients

$$|\tilde{\Psi}_{AB}\rangle = \sum_{k}^{\tilde{r}} s_k|a_k\rangle_A|b_k\rangle_B, \tag{28}$$

where the Schmidt basis vectors $\{|a_k\rangle_A\}$ and $\{|b_k\rangle_B\}$ are the same as in the original state, but now include only the $\tilde{r}$ largest Schmidt coefficients.

C Canonical forms

In this appendix, we aim to explain the most important representations of a Matrix Product State (MPS) and the procedures for obtaining them.

As discussed in the main text, a quantum state is not represented by a unique MPS. There exists a gauge freedom that allows us to insert an identity in the form XX^{-1} between any two tensors $A^{[i]}$ and $A^{[i+1]}$ in the MPS. By reabsorbing X and X^{-1} into $A^{[i]}$ and $A^{[i+1]}$, respectively, we obtain a new MPS that still represents the same quantum state. This flexibility, while preserving the physical state, can be leveraged to transform the MPS into specific forms known as canonical forms.

Canonical forms are particularly useful because they exhibit advantageous properties. For instance, in these forms, the leading left or right (or sometimes both) vectors are identities, which generally results in more stable and efficient numerical computations. Moreover, these forms are closely related to the Schmidt decomposition, enabling optimal truncation of the state when necessary.

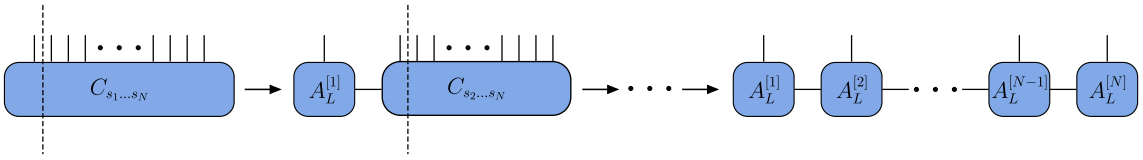

Figure 23: Construction of a MPS from a generic quantum state $|\psi\rangle$. The process involves an iterative series of SVDs across different bipartitions of the state, starting from the complex coefficient vector $C_{s_1 \ldots s_N}$. The resulting tensors $A_L^{[i]}$ are in the left canonical form.

Although the concepts behind MPS and infinite MPS (iMPS) are similar, canonical forms play a more critical role in the iMPS context. In MPS, these forms often emerge naturally during the construction process, whereas in iMPS, achieving canonical forms requires additional effort. For this reason, we will review the canonical forms separately for MPS and iMPS, highlighting their significance and the methods to obtain them in each case.

C.1 MPS

Here we will see how the canonical forms arise naturally when one constructs and MPS, and how the canonical forms are deeply related to the normalization of our state.

For this reason, first we will review how to construct an MPS from a general quantum state,

$$\psi = \sum_{s_1,\ldots,s_N} C_{s_1 \ldots s_N} |s_1, \ldots, s_N\rangle, \tag{29}$$

we begin by reshaping the state vector, originally of dimension d^N, into a matrix with dimensions $d \times d^{N-1}$, where

$$C_{s_1, s_2 \ldots s_N} = C_{s_1 \ldots s_N}. \tag{30}$$

Next, we apply SVD to C, yielding

$$C_{s_1, s_2 \ldots s_N} = \sum_{a_1} U^{[1]}_{s_1, a_1} \Lambda^{[1]}_{a_1, a_1} V^{[1]\dagger}_{a_1, s_2 \ldots s_N}, \tag{31}$$

where $U^{[1]}$ is the left-isometry, $\Lambda^{[1]}$ is the diagonal matrix of singular values, and $V^{[1]\dagger}$ is the right-isometry.

The first MPS tensor, $A^{[1]}$, is identified with $U^{[1]}$, and the subsequent matrix is defined as

$$C_{a_1, s_2 \ldots s_N} = \Lambda^{[1]}_{a_1, a_1} V^{[1]\dagger}_{a_1, s_2 \ldots s_N}. \tag{32}$$

This process is repeated iteratively. At each step, we reshape $C_{a_i, s_i \ldots s_N}$ into $C_{a_i s_i, s_{i+1} \ldots s_N}$ and perform SVD to obtain the next MPS tensors $A^{[i]}$, continuing until the original coefficients are decomposed as

$$\begin{aligned} C_{s_1 \ldots s_N} &= \sum_{a_1,\ldots,a_{N-1}} A^{[1]}_{s_1, a_1} A^{[2]}_{s_2, a_1, a_2} \cdots A^{[N-1]}_{s_{N-1}, a_{N-2}, a_{N-1}} A^{[N]}_{s_N, a_{N-1}} \\ &= A^{[1]}_{s_1} A^{[2]}_{s_2} \cdots A^{[N-1]}_{s_{N-1}} A^{[N]}_{s_N}, \end{aligned} \tag{33}$$

where the final expression omits repeated indices to emphasize the matrix multiplication among the tensors. Figure 23 provides a diagrammatic representation of this decomposition process.

When the MPS is constructed following this procedure, the state is said to be in the left-canonical form, which satisfies the following conditions

$$\sum_{s_i} A^{[i]\dagger}_{s_i} A^{[i]}_{s_i} = \mathbb{1}, \tag{34}$$

$$\sum_{s_i} A^{[i]}_{s_i} \Omega^{[i]} A^{[i]\dagger}_{s_i} = \Omega^{[i-1]}, \tag{35}$$

where $\Omega^{[i]}$ is the squared singular value matrix $\Lambda^{[i]^2}$ from the SVD of the wave function coefficient $C_{a_{i-1}s_i,s_{i+1}\ldots s_N}$.

The tensors $A^{[i]}$ obtained through this process naturally fulfill these conditions. The simplest to demonstrate is Eq.(34), which directly follows from the left-normalization property inherited from the SVD. Specifically, each left-isometry $U^{[i]}$, corresponding to the MPS tensors $A^{[i]}$, satisfies

$$\begin{aligned}\sum_{s_i,a_{i-1}} A^{[i]\dagger}_{a_i,a_{i-1},s_i} A^{[i]}_{s_i,a_{i-1},a'_i} &= \sum_{s_i,a_{i-1}} U^{\dagger}_{a_i,a_{i-1}s_i} U_{s_i a_{i-1},a'_i} \\ &= \delta_{a_i,a'_i}.\end{aligned} \tag{36}$$

However, Eq.(35) is less straightforward to demonstrate. To understand this, consider the intermediate coefficient $C^{[i]}$ and perform an SVD

$$C^{[i]} = U^{[i]} \Lambda^{[i]} V^{[i]\dagger}. \tag{37}$$

Multiplying $C^{[i]}$ by its conjugate transpose yields

$$C^{[i]} C^{[i]\dagger} = U^{[i]} \Lambda^{[i]} V^{[i]\dagger} V^{[i]} \Lambda^{[i]} U^{[i]\dagger} = U^{[i]} \Omega^{[i]} U^{[i]\dagger}, \tag{38}$$

which corresponds to the right side of Eq.(35).

Additionally, $C^{[i]}$ can also be expressed as

$$C^{[i]} = \Lambda^{[i-1]} V^{[i-1]\dagger}, \tag{39}$$

as derived from Eq. (32).

Combining Eq.(38) and Eq.(39) leads to the final relationship

$$\Omega^{[i-1]} = \Lambda^{[i-1]} V^{[i-1]\dagger} V^{[i]} \Lambda^{[i-1]} = U^{[i]} \Omega^{[i]} U^{[i]\dagger} = \sum_{s_i} A^{[i]}_{s_i} \Omega^{[i]} A^{[i]\dagger}_{s_i}. \tag{40}$$

An MPS can also be constructed by performing the same procedure starting from the right. In this case, each new tensor $A^{[i]}$ is associated with the isometry $V^{[i]\dagger}$ from each SVD. This procedure yields a right-normalized MPS in the so-called right-canonical form, which now fulfills the properties

$$\sum_{s_i} A^{[i]}_{s_i} A^{[i]\dagger}_{s_i} = \mathbb{1}, \quad \sum_{s_i} A^{[i]\dagger}_{s_i} \Omega^{[i-1]} A^{[i]}_{s_i} = \Omega^{[i]}. \tag{41}$$

A mixed-canonical form is achieved by performing SVDs from the left up to site i, and from the right for the remaining part of the state. The wavefunction coefficients are then expressed as

$$C_{s_1,\ldots,s_N} = A^{[1]}_{s_1} \cdots A^{[i]}_{s_i} \Lambda^{[i]} B^{[i+1]}_{s_{i+1}} \cdots B^{[N]}_{s_N}, \tag{42}$$

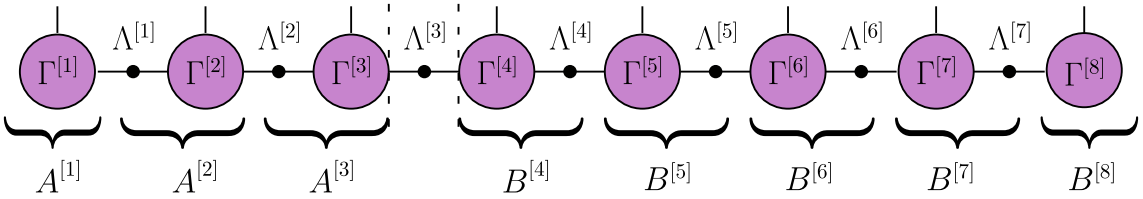

Figure 24: Matrix Product State (MPS) in Vidal form. The tensors $\Gamma^{[i]}$ are order-3 tensors that include the physical indices, while $\Lambda^{[i]}$ are non-negative diagonal matrices containing the singular values corresponding to the bipartition of the state at bond i. The figure also illustrates how, by selecting a specific bond, the state transitions into the mixed-canonical form.

where $A^{[k]}$ are left-normalized, $B^{[k]}$ are right-normalized, and $\Lambda^{[i]}$ is a diagonal matrix containing the singular values at bond $(i, i+1)$. The mixed-canonical form is particularly useful due to its direct connection to the Schmidt decomposition. If we describe the state across the bond $(i, i+1)$ and redefine the basis $\{|s_i\rangle\}$ into two new sets $\{|a_i\rangle_A\}$ and $\{|b_i\rangle_B\}$, we have

$$\begin{aligned} |a_i\rangle_A &= \sum_{s_1,\ldots,s_i} (A^{[1]}_{s_1} \cdots A^{[i]}_{s_i})_{a_i} |s_1, ..., s_i\rangle, \\ |b_i\rangle_B &= \sum_{s_{i+1},\ldots,s_N} (B^{[i+1]}_{s_{i+1}} \cdots B^{[N]}_{s_N})_{a_i} |s_{i+1}, ..., s_N\rangle, \end{aligned} \tag{43}$$

where, due to the left-normalization of the A tensors and the right-normalization of the B tensors, each new set of vectors forms an orthonormal basis.

If we write $\Lambda^i_{a_i,a_i} = \lambda_i$, then the state can be expressed as

$$|\Psi\rangle = \sum_i \lambda_i |a_i\rangle_A |b_i\rangle_B, \tag{44}$$

which corresponds exactly to the Schmidt decomposition of the subsystems $A \equiv \{s_1, ..., s_i\}$ and $B \equiv \{s_{i+1}, ..., s_N\}$.

This form allows for efficient low-rank approximations by retaining only the largest singular values, as detailed in Appendix B, and provides numerical stability while enabling efficient computation.

For this reason, it is highly desirable to work with an MPS that can easily transition to the mixed-canonical form at each bond. This representation, proposed by Vidal in Ref.[Vid03], is referred to in this work as the Vidal form. It is expressed as

$$|\Psi\rangle = \sum_{s_1,\ldots,s_N} \Gamma^{[1]}_{s_1} \Lambda^{[1]} \Gamma^{[2]}_{s_2} \Lambda^{[2]} \cdots \Gamma^{[N-1]}_{s_{N-1}} \Lambda^{[N-1]} \Gamma^{[N]}_{s_N} |s_1, ..., s_N\rangle, \tag{45}$$

where $\Gamma^{[i]}$ are order-3 tensors (except for the first and last, which are order-2), and $\Lambda^{[i]}$ are diagonal matrices containing the singular values at each bond.

The Vidal form can be easily constructed in the same way as the left-canonical or right-canonical form MPS. Starting with the wavefunction coefficient $C_{s_1,\ldots,s_N}$, we perform SVD from the left

$$\begin{aligned} C_{s_1,s_2\ldots s_N} &= \sum_{a_1} A^{[1]s_1}_{a_1} \Lambda_{a_1,a_1} V^{[1]\dagger}_{a_1,s_2\ldots s_N} = \sum_{a_1} \Gamma^{[1]s_1}_{a_1} C_{a_1 s_2, s_3\ldots s_N} = \\ &= \sum_{a_1}\sum_{a_2} \Gamma^{[1]s_1}_{a_1} A^{[2]s_1}_{a_1,a_2} \Lambda_{a_2,a_2} V^{[2]\dagger}_{a_2,s_3\ldots s_N} = \sum_{a_1}\sum_{a_2} \Gamma^{[1]s_1}_{a_1} \Lambda^{[1]}_{a_1,a_1} \Gamma^{[2]s_1}_{a_1,a_2} C_{a_2 s_3, s_4\ldots s_N}, \end{aligned} \tag{46}$$

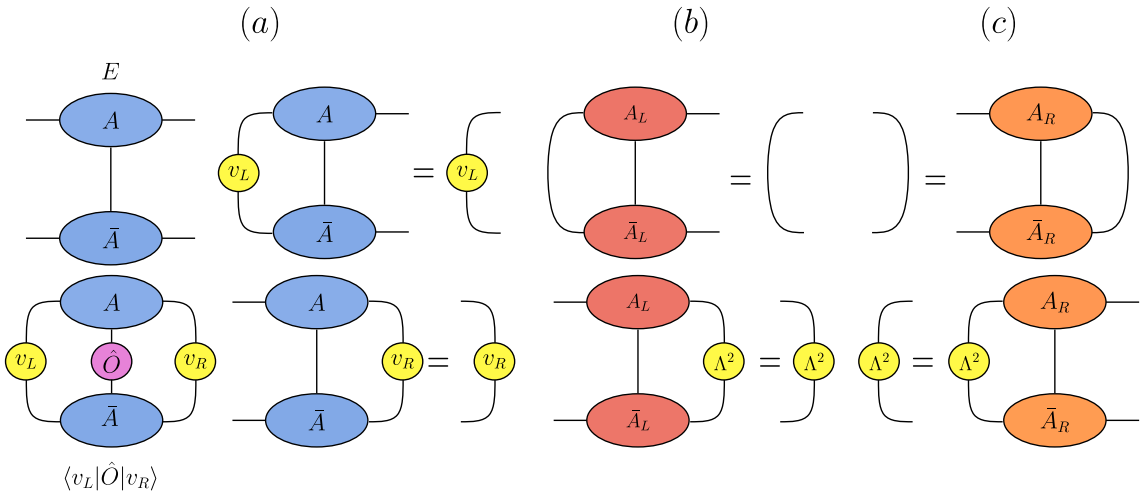

Figure 25: (a) Transfer operator E of an iMPS constructed from the tensor A, showing the leading left eigenvector V_L and the leading right eigenvector V_R of the transfer operator. This also illustrates how expectation values are computed in an iMPS. (b) In the left-canonical form, the leading left eigenvector corresponds to the identity matrix, while the leading right eigenvector corresponds to the square of the singular values Λ^2. (c) In the right-canonical form, the leading left eigenvector corresponds to the square of the singular values Λ^2, and the leading right eigenvector corresponds to the identity matrix.

where, at each step, we multiply the newly computed left-normalized tensor A^i by $\Lambda^{[i-1]}(\Lambda^{[i-1]})^{-1}$ on the left and absorb the inverse of the singular values into A^i. The first and last tensors can be handled similarly by introducing a dummy index and defining $\Lambda^{[0(N)]} = 1$ to maintain consistency.

This process can be done symmetrically starting from the right, thereby obtaining the $\Gamma^{[i]}$ tensors from the right-normalized tensors $B^{[i]}$. This procedure establishes a direct relationship between the Vidal form, the left-canonical form, and the right-canonical form

$$A^{[i]}_{s_i,a_{i-1},a_i} = \Lambda^{i-1}_{a_{i-1},a_{i-1}}\Gamma^{[i]}_{s_i,a_{i-1},a_i}, \quad B^{[i]}_{s_i,a_{i-1},a_i} = \Gamma^{[i]}_{s_i,a_{i-1},a_i}\Lambda^{i}_{a_i,a_i}. \tag{47}$$

As a result of these relationships, when we select a bond defined by the matrix $\Lambda^{[i]}$, all the tensors to the left of this bond transform into left-normalized tensors A, and all the tensors to the right become right-normalized tensors B, thus achieving the mixed-canonical form at the chosen bond, see Fig. 24. In this context, we have used A and B to represent left- and right-normalized tensors, respectively. However, in other sections, we typically use the notations A_L or A_R to denote these tensors explicitly. The choice of notation depends on the context; in this case, we opted for A and B to avoid excessive subscripts.

C.2 iMPS

In the case of an iMPS, obtaining canonical forms is more complex compared to finite MPS due to the infinite nature of the state, which is composed of an unbounded number of tensors. However, established procedures exist for achieving canonical forms in iMPS, and these procedures are closely tied to a central concept in iMPS theory: the transfer matrix. Therefore, before delving into the methods for obtaining canonical forms in iMPS, we will first review the concept of the transfer matrix [VHV19] and explore its critical role in defining canonical forms.

The transfer matrix, also known as the transfer operator, is defined as

$$E = \sum_{s=1}^{d} A^s \otimes \bar{A}^s, \tag{48}$$

where $\bar{A}$ denotes the complex conjugate of the tensor A. The transfer operator is a matrix with dimensions $D^2 \times D^2$, and it is represented diagrammatically in Fig.25 (a).

While the transfer matrix itself possesses significant properties, our primary interest lies in its leading right and left eigenvectors, v_r and v_l. These leading eigenvectors are typically Hermitian and non-degenerate [PGVWC07], capturing key properties of the state. For instance, they are crucial for computing expectation values in the infinite system, as expressed by

$$\langle \hat{O}_i \rangle = \lim_{N\to\infty} \frac{\mathrm{tr}(E^{N-1} E_{O_i})}{\mathrm{tr}(E^N)}, \tag{49}$$

where $\hat{O}_i$ is a one-body operator applied to site i, and E_{O_i} is defined as

$$E_{O_i} = \sum_{s_i, s_i'=1}^{d} A^{s_i'} \otimes \bar{A}^{s_i} \hat{O}_{s_i, s_i'}. \tag{50}$$

We can then perform a spectral decomposition of the transfer operator into left and right eigenvectors $\langle L_p|, |R_p\rangle$

$$E = \sum_p \mu_p |R_p\rangle\langle L_p|, \tag{51}$$

where $|\mu_p| \leq 1$ because the state is normalized. The largest eigenvalue $\mu_{\max} = 1$ corresponds to the leading eigenvectors. In the limit $N \to \infty$, only the terms associated with the leading eigenvectors survive in Eq.(49), simplifying the expression to

$$\langle \hat{O}_i \rangle = \frac{\langle v_l | E_{O_i} | v_r \rangle}{\mathrm{tr}(\langle v_l | v_r \rangle)}. \tag{52}$$

This equation highlights the crucial role that leading vectors play in calculating expectation values in infinite systems, as illustrated in Fig. 25(a). To facilitate these computations, it is beneficial to work in a gauge where these vectors are straightforward to calculate. This necessity for computational simplicity directly relates to the concept of canonical forms.

If we now recall the properties of the canonical forms in the finite case and apply them to iMPS, we find that the conditions that each canonical form fulfilled in the MPS context now correspond to conditions that impose the form of the leading vector in iMPS.

The left-canonical form in iMPS fulfills the conditions

$$\begin{aligned} \sum_{s,\alpha} A^s_{L;\alpha,\beta} \bar{A}^s_{L;\alpha,\gamma} = \delta_{\beta,\gamma} \Leftrightarrow \sum_s A_L^{s\dagger} A_L^s = \mathbb{1}, \\ \sum_{s,\beta} A^s_{L;\alpha,\beta} \Omega_{\beta,\beta} \bar{A}^s_{L;\gamma,\beta} = \Omega^{\alpha\gamma} \delta_{\alpha\gamma} \Leftrightarrow \sum_s A_L^s \Omega A_L^{s\dagger} = \Omega, \end{aligned} \tag{53}$$

where A_L is a left-normalized tensor, $\Omega = \Lambda^2$, and Λ are the singular values.

This equation indicates that the iMPS has a left leading vector $v_l = \mathbb{1}$ and a right leading vector $v_r = \Omega$. These conditions are represented diagrammatically in Fig.25 (b).

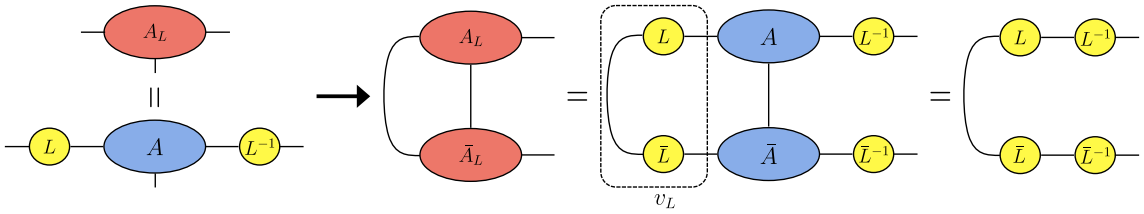

Figure 26: Gauge transformation to convert an iMPS composed of A tensors into the left-canonical form A_L. The matrix L is derived from the decomposition of the leading left eigenvector v_L in the iMPS, expressed as $v_L = L^\dagger L$. The figure also provides a graphical demonstration that the resulting tensor A_L is left-normalized.

Alternatively, the right-canonical form can be interpreted as an iMPS where the left leading vector is $v_l = \Omega$ and the right leading vector is $v_r = \mathbb{1}$. These conditions are also represented diagrammatically in Fig.25 (c).

Finally, the mixed-canonical form for an infinite MPS is defined by combining the left and right canonical forms. Although neither of the leading vectors can be directly associated with the identity, by focusing on the tensor A_C, chosen as the orthogonality center of the state, the contributions from the left and right subsystems effectively reduce to identities. This simplification facilitates the efficient computation of expectation values. Additionally, the orthogonality center A_C can be transformed into a diagonal matrix Λ, enabling the expression of the Schmidt decomposition across the chosen bond, similar to the finite case.

Understanding the relationship between canonical forms and their leading vectors provides a method to transition to the desired canonical form. For instance, to transform a general tensor A into the left-canonical form A_L, we can perform a gauge transformation that converts the left leading vector into the identity matrix. This process involves computing the left leading vector v_l of the system, which can be achieved numerically using techniques like the power method [For15] or the Lanczos algorithm [Pai80]. Since the leading vectors are Hermitian, v_l can be decomposed as $v_l = L^\dagger L$. By inserting the identity $L^{-1}L$ at each bond of the iMPS and reabsorbing the L and L^{-1} matrices into the tensor A, we obtain

$$A_L = LAL^{-1}, \tag{54}$$

where the resulting tensor A_L is now left-normalized, see Fig.26. This process partially fixes the gauge, ensuring that the left leading vector is the identity. However, residual gauge freedom remains, allowing transformations of the form $A'_L = UA_LU^\dagger$, which preserve the left leading vector. This residual freedom can be utilized to diagonalize the right leading vector into Ω, which will later be identified with the singular values.

A similar strategy can be applied to obtain the right-canonical form, where the goal is to find a gauge transformation $A_R = R^{-1}AR$ that normalizes the right leading vector, with the residual gauge freedom used to diagonalize the left leading vector.

With both the left and right canonical forms defined, we can now achieve the mixed-canonical form. As we have commented, we select a specific tensor A^i to serve as the orthogonality center. The tensors to the left of this center are transformed into the left-canonical form, while those to the right are transformed into the right-canonical form. The resulting center tensor is then given by $A_C = LAR$. It is important to note that the left and right normalized tensors can be recovered from the center tensor using the following

relationships

$$A_C = LAR = LAL^{-1}LR = A_L C,$$
$$A_C = LAR = LRR^{-1}AR = CA_R, \tag{55}$$

where the center tensor A_C is expressed in terms of the left (right) normalized tensor $A_{L(R)}$ and a matrix $C = LR$.

This matrix C acts as a crucial link between the left-normalized and right-normalized systems. The relation $CA_R = A_L C$, when viewed through the lens of the transfer matrix formalism, reveals that $CC^\dagger$ and $C^\dagger C$ correspond to the leading right and left vectors in the left and right canonical forms, respectively.

By diagonalizing the matrix C using Singular Value Decomposition (SVD), we obtain the isometries U and $V^\dagger$ along with the singular values Λ. We can then use the residual gauge freedom to transform the tensors on the left of C by $U^\dagger A_L U$ and similarly, transform the tensors on the right by $V A_R V^\dagger$. This process clarifies that, even within the iMPS framework, the mixed-canonical form is intimately connected to the Schmidt decomposition. This makes the mixed-canonical form the optimal choice for the efficient truncation of the iMPS. Furthermore, after applying these residual gauge transformations, the leading vectors $CC^\dagger$ and $C^\dagger C$ in the left and right canonical forms are transformed into $\Omega = \Lambda^2$.

As explained in the context of finite systems, it is advantageous to have this canonical form at every bond. In the infinite case, achieving this is more straightforward due to translational invariance, where the entanglement spectrum remains consistent across all bonds. Consequently, we only need to multiply by the identity $\Lambda\Lambda^{-1}$ across all other bonds to obtain the Vidal form

$$|\Psi\rangle = \sum_{\{s\}} \prod_{r\in\mathbb{Z}} \Gamma^{s_r}\Lambda^{s_r}|\{s\}\rangle. \tag{56}$$

D DMRG optimization

In this appendix, we will briefly explain one of the most standard techniques in TN tools: the Density Matrix Renormalization Group (DMRG). Originally developed in the context of many-body physics to obtain low-energy properties of 1-D quantum systems, the core of the DMRG algorithm can be traced back to a variational optimization of a cost function that depends quadratically on the tensors.

To clarify its origins, we first review the case where the cost function to optimize is the expectation value $\langle\Psi|H|\Psi\rangle$, where H is a given Hamiltonian and $|\Psi\rangle$ is a state in an MPS form that we want to optimize in order to minimize the cost function. For simplicity, we assume that H can be written as a matrix product operator (MPO), which is the operator analog of an MPS, with a physical leg that now has two indices.

The goal is to minimize the cost function variationally while imposing the normalization condition $\langle\Psi|\Psi\rangle = 1$. This can be achieved using a Lagrange multiplier to enforce the normalization constraint, yielding a new cost function f of the form

$$f(A, \bar{A}) = \langle\Psi|H|\Psi\rangle - \lambda\langle\Psi|\Psi\rangle, \tag{57}$$

where λ is the Lagrange multiplier, and A is the tensor forming the MPS, with $\bar{A}$ as its complex conjugate.

To optimize this function, we iteratively update each tensor A^i within the MPS until convergence to a fixed point. The tensor A^i is obtained by extremizing Eq.(57), specifically

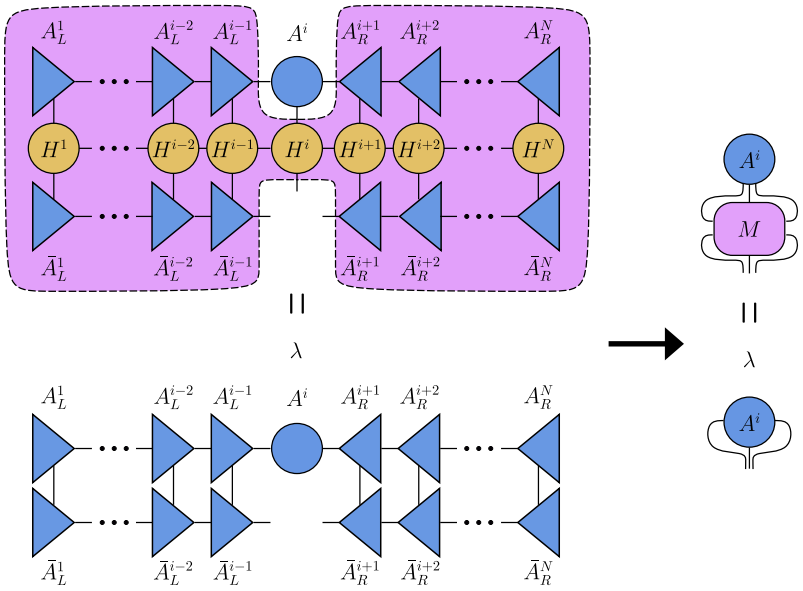

Figure 27: The algebraic equation to find the optimal A^i written in mixed-canonical form. Here, A_L^i represents the i-th tensor in left-canonical form, and A_R^i represents the i-th tensor in right-canonical form. The left and right canonical forms are normalized, reducing the problem to an eigenvector equation.

by solving

$$\frac{\partial f}{\partial \bar{A}^i} = \frac{\partial f}{\partial \bar{A}^i} \left(\langle \Psi | H | \Psi \rangle - \lambda \langle \Psi | \Psi \rangle \right) = 0, \tag{58}$$

where A^i and $\bar{A}^i$ are treated as independent variables. This approach is commonly used in the optimization of complex functions, relying on the principles of complex calculus, particularly the Wirtinger derivative[KQKR23].

Due to the linearity of tensor contractions, the derivative of the cost function with respect to $\bar{A}^i$ results in the same function with a "hole" where $\bar{A}^i$ would be. In the mixed-canonical form, where the orthogonality center is at A^i, the resulting equation directly corresponds to an eigenvalue problem

$$\sum_{a'} M_{a,a'} A^i_{a'} = \lambda A^i_a, \tag{59}$$

where the tensor A^i has been reshaped into a vector and the remaining tensor network into the matrix M. This process is illustrated in Fig.27.

Thus, the solution to the minimization function can be found by identifying the eigenvector with the smallest eigenvalue, a task well-suited to powerful numerical methods such as the Lanczos algorithm [Pai80]. Once the optimal A^i is found, it replaces the original tensor, and the process moves on to the next tensor, repeating until convergence.

The DMRG method is not restricted to only ground state problems, in general, DMRG can optimize cost functions quadratic in the coefficients of tensors (where "quadratic" counts A and $\bar{A}$ as the same tensor). For example, in Eq. (14) of the main text, during the factorization process, one could use DMRG to optimize the slow (or fast) tensor while keeping all others fixed. In this particular case, DMRG is straightforward and returns the same solution.

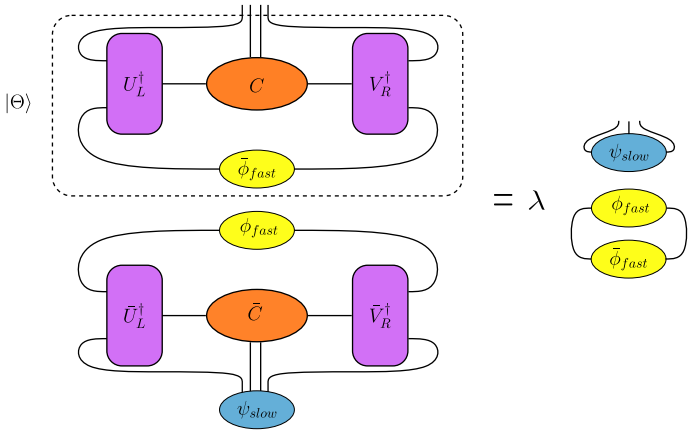

Figure 28: The algebraic equation to find the optimal tensor ψ^{slow} during factorization using the DMRG method. On the right-hand side, the contraction between the tensors ϕ^{fast} corresponds to the norm of the tensor, which is equal to 1.

Let's examine this scenario in detail. The cost function to maximize is

$$\begin{aligned} &\max_{|\psi^{slow}_{LSR}\rangle} |\langle\psi^{slow}_{LSR}| \otimes \langle\phi^{fast}_{L_f R_f}| U_L^\dagger \otimes \mathbb{1}_S \otimes V_R^\dagger |\Psi\rangle|^2 = \\ &= \max_{|\psi^{slow}_{LSR}\rangle} \operatorname{tr}\left(\langle\psi^{slow}_{LSR}| \otimes \langle\phi^{fast}_{L_f R_f}|\varphi\rangle\langle\varphi|\psi^{slow}_{LSR}\rangle \otimes |\phi^{fast}_{L_f R_f}\rangle\right), \end{aligned} \tag{60}$$

where $|\varphi\rangle \equiv U_L^\dagger \otimes \mathbb{1}_S \otimes V_R^\dagger |\Psi\rangle$.

In this equation, the state to optimize is $|opt\rangle \equiv |\psi^{slow}_{LSR}\rangle \otimes |\phi^{fast}_{L_f R_f}\rangle$, which represents a physical state, thus requiring the normalization constraint $\langle opt|opt\rangle = 1$ using a Lagrange multiplier. When we extremize the cost function to find the optimal tensor ψ^{slow}, we obtain a new eigenvalue problem, where the eigenvector corresponds to the optimal state $|\psi^{slow}_{\text{opt}}\rangle$, as depicted in Fig. 28. Note that in this case, the matrix to find the eigenvectors has the form $M = |\Theta\rangle\langle\Theta|$, which is equivalent to the density matrix of a pure state. Therefore, we know that the maximum eigenvalue (and the only one different from 0) is given by the eigenvector $|\Theta\rangle$, which corresponds to the optimal tensor ψ^{slow} obtained by solving Eq. (14).

The optimization of the unitary tensors U_L and V_R cannot be directly performed using the DMRG process, as the unitary constraint cannot be easily imposed with a Lagrange multiplier. Instead, the unitarity condition is more naturally treated as a singular value decomposition (SVD) problem rather than an eigenvalue problem. This is the typical approach for optimization under unitarity or isometry constraints, as seen in Ref. [EV09], which in our specific case aligns with the analytical solution provided in the main text.

In summary, DMRG is an incredibly powerful tool for solving optimization problems. It is not only highly efficient from a numerical perspective, allowing the mapping of the optimization problem to an eigenvalue problem for which highly optimized solvers exist, but also from an analytical perspective. DMRG provides a robust method for variational optimization using tensors and, in some cases, as demonstrated here, can offer insights that lead to analytical solutions.

E Properties of the Truncation

During the simple truncation process, we identified the system's slow and fast degrees of freedom. Subsequently, the density operator of the entire system $|\Psi\rangle\langle\Psi|$ was replaced with a mixed-state density operator. This involved substituting the fast mode density matrix $\rho_{L_f R_f}$ with a product state of its reduced density matrices, $\rho_{L_f} \otimes \rho_{R_f}$, as depicted in Fig.8 of the main text.

As a result, although the global state differs from the original, the new mixed state retains the same reduced density matrix for the subsystem LS. This can be verified by tracing out the subsystem R in both states, which involves connecting the legs representing the subsystem R. Specifically, in this context, connecting these legs is equivalent to

$$V_R V_R^\dagger = \mathbb{1}_R = \mathbb{1}_{R_s} \otimes \mathbb{1}_{R_f}, \tag{61}$$

where V_R is the disentangling unitary from Eq.(12). The tensor product of identities is represented by connecting the right legs of the slow tensors and the right legs of the fast tensors. Figure 29 illustrates how this procedure yields identical reduced density matrices for both the pure and mixed states. Analogous calculations for the subsystem SR demonstrate that the reduced density matrix is also preserved.

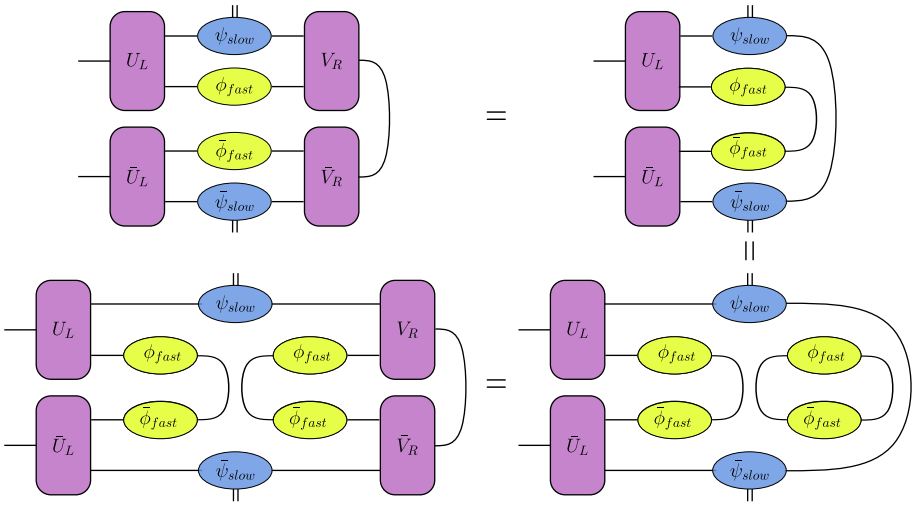

Figure 29: Comparison of the reduced density matrices ρ_{LS}. The upper diagram shows the reduced density matrix obtained from the pure state, while the lower diagram shows the reduced density matrix obtained from the mixed state. Since $|\psi^{fast}_{L_f R_f}\rangle$ is a normalized physical state, the full contraction in the lower diagram between the tensors ϕ_{fast} and $\bar{\phi}_{fast}$ is equal to 1. This operation can be understood as computing $tr(\rho_{fast}) = 1$.

Additionally, it can be shown that the fixed points of the transfer matrix, specifically the leading left and right vectors, remain unchanged. To demonstrate this, we leverage the properties of the left and right canonical forms, see Appendix C, along with their relationship to the Vidal form as described in Eq.(47).

Recall that before truncation, our initial state was described by a unit cell formed by $\{C, \Lambda^{-1}\}$ tensors, where C describes the block of S in our infinite system, and Λ represents the singular values. The relationship between the Vidal tensor Γ and the tensor C is simply:

$$C = \Lambda\Gamma\Lambda. \tag{62}$$

Next, consider the left-canonical form description A_L:

$$A_L = \Lambda\Gamma = C\Lambda^{-1}. \tag{63}$$

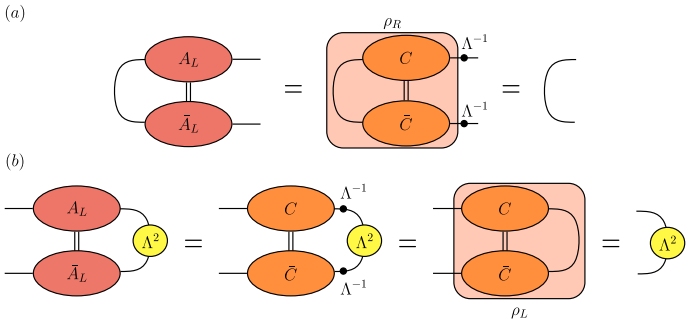

Figure 30: Leading vector equations in the left-canonical form. (a) Left leading vector equation, where A_L is substituted with the tensors corresponding to the unit cell used in the algorithm (C and Λ), showing the appearance of the system's reduced density matrix ρ_L. (b) Right leading vector equation, similarly showing the appearance of the reduced density matrix ρ_R.

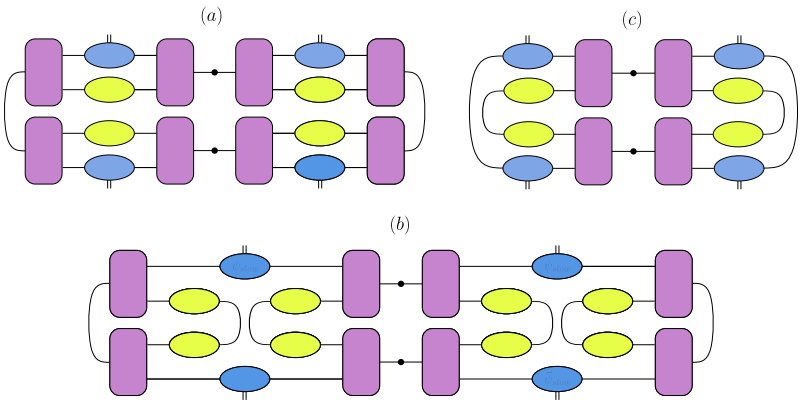

Figure 31: Conservation of the reduced density matrix for two consecutive blocks. (a) The reduced density matrix obtained from the pure state. (b) The reduced density matrix for the mixed state. (c) The result for both after applying the contraction of the corresponding isometries U_L and V_R. In case (b), we use the fact that $\mathrm{tr}(\rho_{fast}) = 1$.

By definition, the left leading vector of the transfer matrix formed by A_L is the identity $\mathbb{1}_L$. This must also hold for the transfer matrix expressed in terms of $C\Lambda^{-1}$. As shown in Fig.30 (a), the reduced density matrix ρ_R appears in the calculation. Since the reduced density matrix is unchanged during truncation, the same equation holds after truncation.

A similar approach applies to the right leading vector. In the left-canonical form, the leading right vector corresponds to $\Omega = \Lambda^2$. Again, this must hold if we express A_L in terms of C. From Fig.30(b), we observe that the reduced left density matrix ρ_L appears in the calculation, indicating that truncation does not affect the right leading vector either.

If the leading vectors are conserved in the left-canonical form, then the gauge transformations required to transition into other canonical forms will act identically before and after truncation, ensuring that the leading vectors are preserved.

Since the leading vectors remain unchanged, we can be confident that the expectation values of operators with support up to $2l$ sites are also conserved. To demonstrate this, note that, because we were already in the canonical form before truncation and remain so afterward, our leading vectors are identities. By using the same approach employed to show that the reduced density matrices are preserved, we can verify that the expectation values for up to $2l$ sites are also conserved, as shown in Fig.31.

F Gradient descent

The gradient descent was performed using automatic differentiation libraries, specifically the Python library JAX. The idea is that if we can access the gradient of our function, we can move in the direction of the steepest descent to approach the minimum value of the cost function. This movement follows the simplest approach

$$T_{i+1} = T_i - \alpha \nabla_{T_i} f(T), \tag{64}$$

where T is the variable of our optimization, α is the learning rate, $f(T)$ is the cost function that depends on T, and ∇_{T_i} is the gradient of the cost function with respect to its optimization variable.

Although this method has demonstrated usefulness, particularly in the context of deep learning, it presents some challenges. The most significant issue is finding the appropriate learning rate for the optimization. If the rate is too high, we risk moving away from our minimum and diverging instead of converging. Conversely, if it is too low, the optimization process will progress too slowly and may not advance towards the minimum. Another potential problem is getting stuck in a local minimum rather than the global one.

To address these issues, various optimizers help adapt the learning rate based on the gradient value and even use the second derivative to avoid local minima. Among these methods, ADAM and L-BFGS-B [KB17, ZBLN97] are ones of the most well-known and widely used. However, these methods are typically implemented for real tensors, and due to time constraints, we have not been able to adapt them for complex numbers. Instead, we experimented with different learning rates and observed which fit best over a few iterations, specifically 150 iterations. We repeated this process until reaching a fixed total number of iterations. Finally, we assessed the final value of the cost function. If it did not fall below a predetermined threshold, we proceeded to the next time evolution step. If it did fall below the threshold, we truncated and performed the time evolution with the new truncated state.

G Time evolution mixed state

In this appendix, we will describe in detail the methods we have used to implement the iTEBD in the factorization algorithm.

Both methods assume that we have applied a first-order Suzuki-Trotter expansion as stated in the main text, where the initial Hamiltonian H is divided into two reduced Hamiltonians, H_o and H_e, corresponding to the odd and even sectors, respectively. The time evolution operator is then decomposed into small 2-body operators U_1 and U_2, each applied to the even and odd bonds, respectively.

G.1 iMPDO$_{\text{v1}}$

In this method, we treat our iMPDO as a purification MPS. The time evolution operator becomes $U_{\sigma_n,\sigma_{n+1}} \otimes \mathbb{1}_{\sigma'_n} \otimes \mathbb{1}_{\sigma'_{n+1}}$, where σ represents the physical index and σ' refers to the purification sites. This translates to applying the U_1 operator to the physical indices while leaving the purification indices unchanged, as illustrated in Fig.32.

Before evolving our wave function, let's expand our initial state in the local basis of our unit cell centered at sites n and $n+1$ using the tensors $\{\hat{A}, \lambda_{AB}, \hat{B}\}$, where $\lambda_{AB} \equiv \Lambda_{AB}^{-1}$.

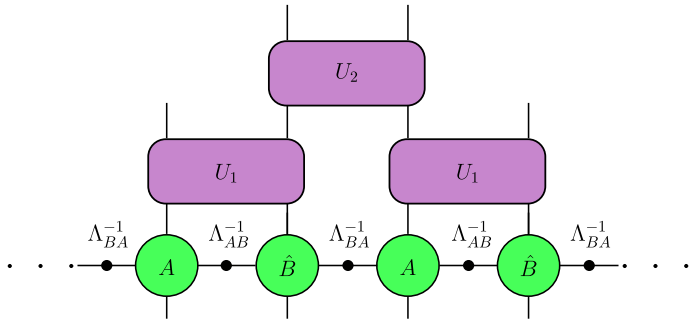

Figure 32: Diagrammatic scheme illustrating the tensor network to evolve an infinite Matrix Product Density Operator (iMPDO) one time step δ_t. The U_1 and U_2 operators are the local gates obtained from the first-order Suzuki-Trotter expansion, dividing our Hamiltonian into the odd and even sectors. The highlighted structure is repeated infinitely along the chains.

Our wave function $|\Psi\rangle$ is then written as

$$|\Psi\rangle = \sum_{\alpha,\gamma}^{\chi} \sum_{\sigma}^{d} \sum_{\sigma'}^{d_{\text{pur}}} \Theta_{[1]\alpha,\gamma}^{\sigma_n,\sigma'_n,\sigma_{n+1},\sigma'_{n+1}} \left|\Phi_{\alpha}^{L_n}\right\rangle |\sigma_n,\sigma'_n\rangle |\sigma_{n+1},\sigma'_{n+1}\rangle \left|\Phi_{\gamma}^{R_{n+1}}\right\rangle, \tag{65}$$

where $\left|\Phi_{\alpha}^{L_n}\right\rangle$ and $\left|\Phi_{\gamma}^{R_{n+1}}\right\rangle$ correspond to the Schmidt bases for the semi-infinite sublattices to the left and right of sites n and $n+1$, respectively. Due to Λ_{AB} and λ_{AB} being diagonal, we have chosen to indicate only the relevant indices for clarity. Additionally, in the summation over σ and σ', we have dropped the subscript as both run over the same dimension.

The coefficient $\Theta_{[1]\alpha,\gamma}^{\sigma_n,\sigma'_n,\sigma_{n+1},\sigma'_{n+1}}$ is given by

$$\Theta_{[1]\alpha,\gamma}^{\sigma_n,\sigma'_n,\sigma_{n+1},\sigma'_{n+1}} = \sum_{\beta}^{\chi} \hat{A}_{\alpha,\beta}^{\sigma_n,\sigma'_n} \lambda_{AB}^{\beta} \hat{B}_{\beta,\gamma}^{\sigma_{n+1},\sigma'_{n+1}}. \tag{66}$$

The next step is to update our evolved wave function by applying the 2-body gate U_1. Mathematically, this update corresponds to

$$\tilde{\Theta}_{[1]\alpha,\gamma}^{\tilde{\sigma}_n,\sigma'_n,\tilde{\sigma}_{n+1},\sigma'_{n+1}} = \sum_{\sigma_n,\sigma_{n+1}}^{d} U_{\sigma_n,\sigma_{n+1}}^{\tilde{\sigma}_n,\tilde{\sigma}_{n+1}} \Theta_{[1]\alpha,\gamma}^{\sigma_n,\sigma'_n,\sigma_{n+1},\sigma'_{n+1}}. \tag{67}$$

Next, we need to recover the new tensors $\tilde{A}$, $\tilde{B}$, and $\tilde{\lambda}_{AB}$ while preserving the canonical form. To do this, we reshape our tensor by combining the bond dimension, the purification leg, and the physical leg to form a matrix $\tilde{\Theta}_{[1]\tilde{\sigma}_{n+1}\sigma'_{n+1}\gamma}^{\tilde{\sigma}_n\sigma'_n\alpha}$. At this point, we perform an SVD and truncate the singular values to obtain the desired structure

$$\tilde{\Theta}_{[1]\tilde{\sigma}_{n+1}\sigma'_{n+1}\gamma}^{\tilde{\sigma}_n\sigma'_n\alpha} = \sum_{\beta}^{\chi_{\max}} \Gamma_{[A]\beta}^{\tilde{\sigma}_n\sigma'_n\alpha} \tilde{\Lambda}_{AB}^{\beta} \Gamma_{[B]\tilde{\sigma}_{n+1}\sigma'_{n+1}\gamma}^{\beta}, \tag{68}$$

where Γ_A and Γ_B are isometries, and $\tilde{\Lambda}_{AB}$ are the Schmidt values of the evolved state.

Finally, we multiply the right of the singular values by $\mathbb{1} = \tilde{\Lambda}_{AB}^{-1}\tilde{\Lambda}_{AB}$, reabsorb the singular values into Γ_A and Γ_B, and reshape the indices to obtain

$$\tilde{\Theta}_{[1]\alpha,\gamma}^{\tilde{\sigma}_n,\sigma'_n,\tilde{\sigma}_{n+1},\sigma'_{n+1}} = \sum_{\beta}^{\chi_{\max}} \tilde{A}_{\alpha,\beta}^{\tilde{\sigma}_n,\sigma'_n} \tilde{\lambda}_{AB}^{\beta} \tilde{B}_{\beta,\gamma}^{\tilde{\sigma}_{n+1},\sigma'_{n+1}}, \tag{69}$$

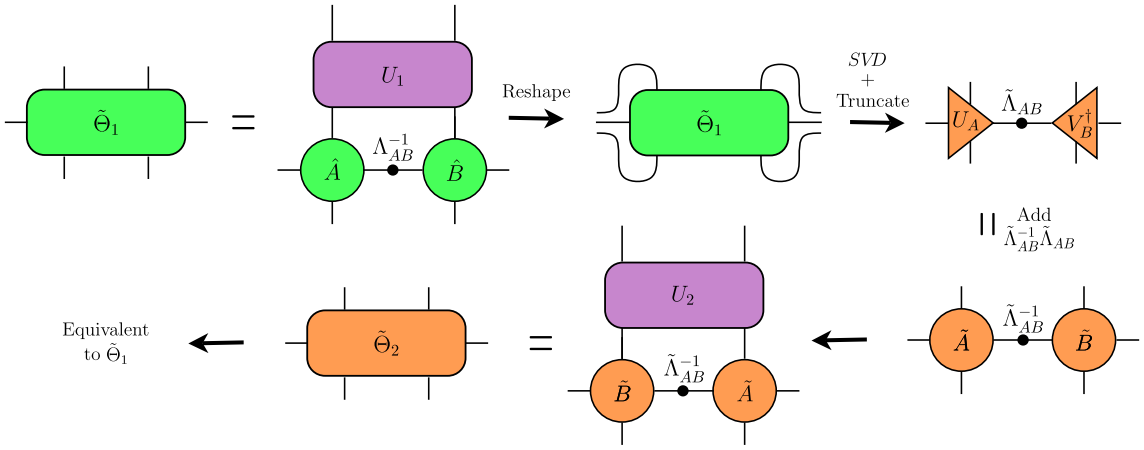

Figure 33: Diagrammatic scheme illustrating the TEBD procedure in our infinite matrix product density operator (iMPDO).

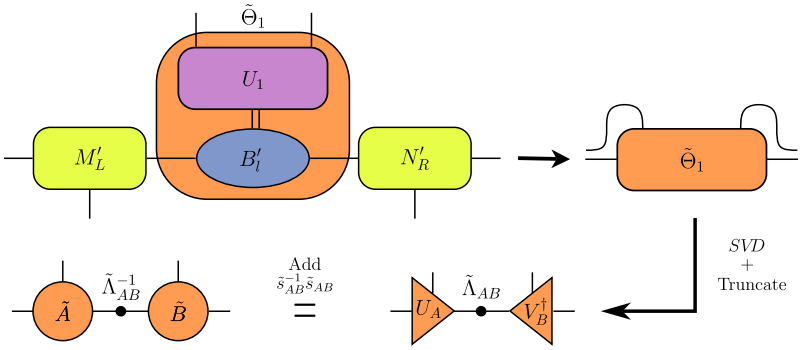

Figure 34: Diagrammatic scheme illustrating the procedure to perform the first iTEBD step.

where $\tilde{A}$, $\tilde{B}$, and $\tilde{\lambda}_{AB} \equiv \tilde{\Lambda}_{AB}^{-1}$ form the desired structure of our updated wave function.

To update the wave function and apply the second TEBD step with the operator U_2, we follow an equivalent procedure, interchanging the roles of $\hat{A}$, $\hat{B}$, Λ_{AB}^{-1}, and U_1 with the tensors $\tilde{B}$, $\tilde{A}$, Λ_{BA}^{-1}, and U_2, respectively. After this step, we obtain the new iMPDO representing the updated state $|\Psi(\delta_t)\rangle$, corresponding to the evolved state at time δ_t. The steps described above are summarized diagrammatically in Fig.33.

G.2 iMPDO$_{v2}$

In this method, we take advantage of the reduced bond dimension achieved through the factorization process. To do so, we first convert the tensor B_l into the new orthogonality center B'_l, as discussed in the main text.

Next, the first block of operators, U_1, is applied directly to the tensor B'_l. For consistency in notation, we denote this tensor as $\Theta_1 \equiv B'_l$. After applying the 2-body gate U_1, the resulting tensor is expressed as

$$\tilde{\Theta}^{\tilde{\sigma}_n,\tilde{\sigma}_{n+1}}_{[1]\alpha,\gamma} = \sum_{\sigma_n,\sigma_{n+1}}^{d} U^{\tilde{\sigma}_n,\tilde{\sigma}_{n+1}}_{\sigma_n,\sigma_{n+1}} \Theta^{\sigma_n,\sigma_{n+1}}_{[1]\alpha,\gamma}, \tag{70}$$

We then reshape the updated tensor by combining the bond dimension and the physical leg into a matrix $\tilde{\Theta}^{\tilde{\sigma}_n\alpha}_{[1]\tilde{\sigma}_{n+1}\gamma}$. In this matrix form, we perform an SVD and truncate, yielding

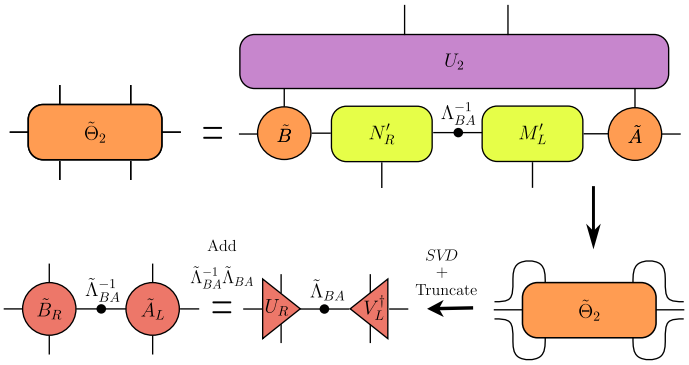

Figure 35: Diagrammatic scheme illustrating the procedure to perform the second iTEBD step.

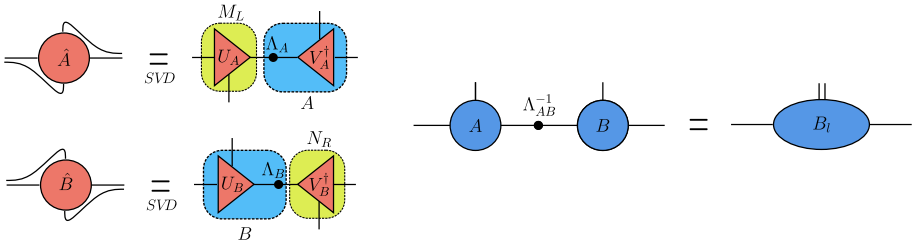

Figure 36: Diagrammatic scheme illustrating the procedure to recover the initial unit cell with the tensors M_L,N_R, and B_l from our iMPDO.

the new tensor

$$\tilde{\Theta}^{\tilde{\sigma}_n\alpha}_{[1]\tilde{\sigma}_{n+1}\gamma} = \sum_{\beta}^{\chi_{\max}} \Gamma^{\tilde{\sigma}_n\alpha}_{[A]\beta}\tilde{\Lambda}^{\beta}_{AB}\Gamma^{\beta}_{[B]\tilde{\sigma}_{n+1}\gamma}, \tag{71}$$

Here, Γ_A and Γ_B are isometries, and $\tilde{\Lambda}_{AB}$ are the singular values.

To ensure that the new isometries become the updated orthogonality centers, we multiply the updated singular values by $\tilde{\Lambda}^{-1}_{AB}\tilde{\Lambda}_{AB}$ and reabsorb the singular values into Γ_A and Γ_B. After reshaping the indices, the updated tensor is given by

$$\tilde{\Theta}^{\tilde{\sigma}_n\tilde{\sigma}_{n+1}}_{[1]\alpha\gamma} = \sum_{\beta}^{\chi_{\max}} \tilde{A}^{\tilde{\sigma}_n}_{\alpha,\beta}\tilde{\lambda}^{\beta}_{AB}\tilde{B}^{\tilde{\sigma}_{n+1}}_{\beta,\gamma}, \tag{72}$$

where $\tilde{A}$ and $\tilde{B}$ are tensors in the standard MPS form, each with one physical index, and $\tilde{\lambda}_{AB}$ corresponds to the inverse of the singular values $\tilde{\Lambda}_{AB}$.

It is crucial to note that throughout this process, the purification index is absent. Consequently, during the SVD of the tensor $\tilde{\Theta}_1$, the new bond dimension obtained is guaranteed to be upper-bounded by a smaller value compared to the previous method. This entire process is depicted diagrammatically in Fig.34.

To perform the second iTEBD step, we apply the 2-body gate U_2 to the tensors $\tilde{B}$ and $\tilde{A}$ of two consecutive unit cells. A challenge arises here: these tensors are separated by N'_R, Λ^{-1}_{BA}, and M'_L. To apply the 2-body gate and then truncate, we must contract all these tensors, resulting in an updated tensor $\tilde{\Theta}_2$, which is equivalent to the one obtained in the previous method. Therefore, the update process remains the same as previously described. For completeness, this process is illustrated in Fig. 35.

After completing the second iTEBD step, we obtain the updated tensors $\tilde{B}_R$ and $\tilde{A}_L$. However, these tensors now form an iMPDO rather than the initial MPS structure composed of the tensors M_L, B_l, and N_R. This issue also arises in the iMPDO method when

applying the slow and fast decomposition after a few iTEBD iterations. The solution remains consistent: we perform an SVD on the tensors $\tilde{B}_R$ and $\tilde{A}_L$ to separate the purification legs from the physical ones, then transfer the singular values to the tensors containing the physical indices. Finally, we group the new physical tensors into one, forming the new tensor B_l. The left isometries become the new M_L and N_R tensors. The entire process is illustrated in Fig. 36.

Bibliography

[EV09] G. Evenbly and G. Vidal. Algorithms for entanglement renormalization. *Physical Review B*, 79(14), April 2009.

[EY36] C. Eckart and G. Young. The approximation of one matrix by another of lower rank. *Psychometrika*, 1(3):211–218, 1936.

[For15] W. Ford. Chapter 22 - Large Sparse Eigenvalue Problems. In William Ford, editor, *Numerical Linear Algebra with Applications*, pages 533-549. Academic Press, Boston, 2015.

[KB17] D. P. Kingma and J. Ba. Adam: A Method for Stochastic Optimization. 2017.

[KQKR23] K. Koor, Y. Qiu, L. C. Kwek, and P. Rebentrost. A short tutorial on Wirtinger Calculus with applications in quantum information. 2023.

[Pai80] C. C. Paige. Accuracy and effectiveness of the Lanczos algorithm for the symmetric eigenproblem. *Linear Algebra and its Applications*, 34:235-258, 1980.

[PGVWC07] D. Perez-Garcia, F. Verstraete, M. M. Wolf, and J. I. Cirac, Matrix product state representations, *Quant. Inf. Comput.*, 7(5-6):401–430, 2007.

[VHV19] L. Vanderstraeten, J. Haegeman, and F. Verstraete, Tangent-space methods for uniform matrix product states, *SciPost Phys. Lect. Notes*, 7, 2019.

[Vid03] G. Vidal, Efficient Classical Simulation of Slightly Entangled Quantum Computations, *Phys. Rev. Lett.*, 91(14):147902, Oct 2003.

[ZBLN97] C. Zhu, R. H. Byrd, P. Lu, and J. Nocedal. Algorithm 778: L-BFGS-B: Fortran subroutines for large-scale bound-constrained optimization. *ACM Trans. Math. Softw.*, 23(4):550–560, Dec. 1997.