PhD Thesis

Simulation of two-dimensional strongly correlated systems via tree-like isometric tensor networks: from physical models to quantum computers

by

Bartosz Rzepkowski

Supervisors
Prof. Arkadiusz Wójs, PhD Eng.
Gunnar Möller, PhD

A thesis presented for the degree of
Doctor of Philosophy

in the
Faculty of Fundamental Problems of Technology
Institute of Theoretical Physics

Wrocław University
of Science and Technology

Wrocław 2023

The work co-financed by the European Union under the European Social Fund

"InterDok – Programy Interdyscyplinarnych Studiów Doktoranckich na Politechnice Wrocławskiej" Grant No. POWR.03.02.00-00-I003/16

European Funds
Knowledge Education Development

Wrocław University of Science and Technology

European Union
European Social Fund

To my beloved Ania,
who brightens everything she touches

"That was the thing about the world: it wasn't that things were harder than you thought they were going to be, it was that they were hard in ways that you didn't expect."

Lev Grossman

ACKNOWLEDGEMENTS

First of all, I would like to thank my advisors, Professor Arkadiusz Wójs and Gunnar Möller, PhD. Thank you for the countless hours spent together looking for solutions to fascinating problems and for the vast amount of knowledge passed on to me. Without you, I would never have been able to grasp even a fraction of the topics we have come across over the years. Moreover, thank you for all your help in the administrative duties related to this endeavour.

I would also like to express my gratitude to Katarzyna Roszak, PhD, for introducing me to the world of physics, showing its beauty and answering my endless questions. It was a great experience to work with you on such interesting projects.

Thanks are also due to two people from the Technical University of Munich. Professor Frank Pollmann, your innovative ideas allowed me to overcome multiple problems encountered while working on this thesis, and your help in understanding the nuances of Isometric Tensor Networks was invaluable. Johannes Hauschild, PhD, thank you for the in-depth explanations on the technical details regarding the implementation of numerous algorithms.

I would like to extend my gratitude to all my colleagues from the room 503. I will never forget the wonderful atmosphere you created in our office, and many fascinating discussions we had there.

I would like to thank all my friends and family, for your undying support throughout this journey. Especially, I would like to thank my Dad, for teaching me to always "keep fighting", and my Mom, for support not only in words, but also in deeds, especially at the end of writing this dissertation.

Finally, this work would never have been possible without the help of my wonderful girlfriend. Ania, thank you for believing in me, even when I did not. If not for you, I would never have come this far.

To all of you, from the bottom of my heart, thank you!

Bartek

ABSTRACT

The analysis of two-dimensional (2D) strongly correlated systems poses a significant challenge due to the exponentially increasing amount of computational resources required by an exact simulation. However, tensor network methods have arisen as potent approximation techniques that enable the resolution of this issue in certain cases. In this thesis, we conduct an extensive investigation of the application of these methods for the analysis of strongly correlated systems, emphasizing the study of the monolayer of CrI_3 and the simulation of random unitary (quantum) circuits.

We begin by applying the Matrix Product State (MPS) based techniques to analyze the magnetic ordering in the spin-3/2 XXZ Hamiltonian on a honeycomb lattice, being an effective model of the monolayer of CrI_3. We demonstrate that magnetic phases emerging in the system can be predicted with high precision by the classical approximation of the model, in a wide range of the parameter space. We find the correlation energy to be the greatest in magnitude for the in-plane ferromagnetic and antiferromagnetic phases, while being equal to zero in the case of the off-plane ferromagnetic one.

Next, we give a pedagogical introduction to two-dimensional Isometric Tensor Networks (isoTNS) and discuss the Moses Move and 2D version of Time Evolving Block Decimation ($TEBD^2$) algorithms that operate on them. We then present modifications to the basic isoTNS framework, which allow for the entire orthogonality surface to be moved into the bulk of the lattice, resulting in a reduced bond-size required when performing calculations. Additionally, we rearrange the entanglement flow in the network to create a tree-like structure, and provide a technique allowing for application of two-site operators on nodes lying on different branches. We compare the two methods on the task of finding the ground state of the transverse field Ising model on a square lattice, by means of imaginary time evolution.

Finally, we show how the modified $TEBD^2$ algorithm can be used to emulate the execution of random quantum circuits, with simultaneous estimation of both the two-qubit and the multi-qubit fidelities, which we compare with the ones obtained by an existing MPS-based technique. For the two demonstrated methods we note very high average fidelity, with respect to the number of parameters used, which is inversely proportional to the amount of entanglement being present in the system. We analyze the bottleneck of the altered $TEBD^2$ algorithm and suggest alternative approaches allowing for its circumvention.

STRESZCZENIE

Analiza dwuwymiarowych (2D) układów silnie skorelowanych stanowi niezwykle trudne wyzwanie ze względu na rosnącą wykładniczo ilość zasobów obliczeniowych wymaganych do ich dokładnej symulacji. W wybranych przypadkach problem ten może zostać rozwiązany dzięki zastosowaniu metod aproksymacyjnych, których sztandarowym przykładem są sieci tensorowe. W niniejszej pracy wykorzystujemy te metody w analizie układów silnie skorelowanych, kładąc szczególny nacisk na badanie własności monowarstwy CrI_3 oraz symulację obliczeń przeprowadzanych na komputerach kwantowych.

Stosujemy algorytmy wykorzystujące tzw. *Matrix Product States* (MPS) do analizy uporządkowania magnetycznego w hamiltonianie spin-3/2 XXZ na siatce plastra miodu, będącym efektywnym modelem monowarstwy CrI_3. Pokazujemy, że fazy magnetyczne pojawiające się w układzie można przewidzieć z dużą precyzją za pomocą klasycznego przybliżenia modelu w szerokim zakresie przestrzeni parametrów. Zauważamy, że energia korelacji jest największa dla faz ferromagnetycznej i antyferromagnetycznej w płaszczyźnie materiału, podczas gdy jest ona równa zeru w przypadku fazy ferromagnetycznej w osi prostopadłej do płaszczyzny.

Następnie przedstawiamy pojęcie *izometrycznych sieci tensorowych* (ang. *Isometric Tensor Networks*) (isoTNS) i omawiamy dwa algorytmy operujące na tej klasie struktur - *ruch Mojżesza* (ang. *Moses Move*) oraz dwuwymiarową wersję metody *Time Evolving Block Decimation* ($TEBD^2$). W dalszej części pracy wprowadzamy dwie modyfikacje podstawowego formalizmu isoTNS oraz wspomnianych metod. Pierwsza z nich umożliwia przesunięcie całej tzw. przestrzeni ortogonalności do wnętrza układu, efektywnie pozwalając na zmniejszenie wymiarów tensorów wykorzystywanych w trakcie obliczeń. Ponadto, pokazujemy metodę rearanżacji przepływu entropii splątania przez system, efektywnie przekształcając go w strukturę drzewiastą. Prezentujemy także technikę aplikacji operatorów działających na węzłach znajdujących się na różnych gałęziach takiego układu. Porównujemy podstawowe i zmodyfikowane metody wykorzystujące isoTNS na zadaniu znalezienia stanu podstawowego modelu Isinga w zewnętrznym polu poprzecznym (ang. *transverse field Ising model*) poprzez ewolucję w czasie urojonym.

Na koniec pokazujemy, w jaki sposób zmodyfikowany algorytm $TEBD^2$ może być wykorzystany do emulacji wykonywania losowych obwodów kwantowych, z jednoczesnym oszacowaniem wierności dwu- oraz wielokubitowej. Otrzymane wyniki porównujemy z rezultatami uzyskanymi za pomocą istniejącej metody bazującej na MPS. Dla dwóch zademonstrowanych technik odnotowujemy bardzo wysoką średnią precyzję dwukubitową, w stosunku do liczby użytych parametrów, która jest odwrotnie proporcjonalna do ilości splątania występującego w układzie. Przeprowadzamy dogłębną analizę wąskiego gardła zmodyfikowanego algorytmu $TEBD^2$ oraz proponujemy alternatywne podejście potencjalnie pozwalające na jego obejście.

CONTENTS

List of Figures

List of Tables

FIRST CHAPTER

INTRODUCTION

Strongly correlated materials have long been a major focus of research in the field of condensed matter physics. Systems belonging to this class cannot be described accurately in terms of its non-interacting constituents [1], resulting in different competing phases with extremely intriguing properties. One of the most notable instances of such material is a high-temperature superconductor [2], whose discovery has sparked an unabated interest in the community that continues to this day. Other examples worth mentioning are magnetic systems, in which due to the interaction between spins various magnetic orderings may arise, among which the simplest examples are the ferromagnetic (all spins aligned) and antiferromagnetic (spins antialigned) ones [3]. The analysis of physical properties of such materials is important not only from a purely scientific perspective, but can also lead to many technological applications.

However, exact simulations of such systems with the use of classical methods poses a great challenge due to the exponentially increasing size of the Hilbert space with respect to the number of particles. In 1982 Richard Feynman proposed to harness quantum physics for the purpose of computation [4] and build a completely new kind of machine, whose number of elements would be proportional to the size of the simulated system. Thus was born the idea of a *quantum computer*. Since then, significant progress has been made in the development of quantum computing hardware and algorithms for simulating quantum systems.

One method involves the usage of analog quantum simulators, which are specially designed quantum systems that can emulate the behavior of other quantum systems [5–8]. An alternative approach is to perform a digital quantum simulation, basing on the circuit model of quantum computation [9–16].

Nevertheless, building a fault-tolerant quantum computer is an extremely complex engineering challenge, as these machines are highly susceptible to errors that can cause the delicate quantum state to lose its properties, rendering the final result meaningless. These inaccuracies might arise from improperly applied gates on the *qubits* (basic units of quantum information), their initialization, measurement, or a decoherence process, in which the environment by interacting with qubits changes their state [17]. To protect quantum computers against this noise researchers have developed a variety of quantum error-correction codes (QEC) [18–20]. Yet, these methods require many more qubits and gates than the numbers needed to conduct the original computation [21, 22]. Because of this overhead, currently

available machines are not protected by the QEC, and the scale of computations that can be reliably performed using this technology is severely constrained by the noise. Because of that, these machines are referred to as *Noisy Intermediate-Scale Quantum* (NISQ) computers [23].

Until a fault-tolerant quantum computer is built, one of the most powerful approximation tools currently available for simulating quantum many-body systems are *tensor networks*. Two of the most prominent examples of these mathematical structures are *Matrix Product States* (MPS) [24–28] and *Projected Entangled Pair States* (PEPS) [29, 30]. The former can describe wavefunctions with a single-dimensional (1D) chain of connected tensors, while the latter is a two-dimensional (2D) structure, with tensors distributed on the nodes of a grid. To fully exploit the potential of an MPS, it is usually generated in the so-called *canonical form*. All of the tensors in such an MPS (except one) fulfil the *isometry condition*, which allows them to be ignored while updating the tensor network or calculating the expectation value of some observable, significantly accelerating the computations. Thus, MPS is an example of a 1D *Isometric Tensor Network*. Although MPSs are 1D structures, they can be used to simulate 2D systems. However, their scaling in this context is inferior to the 1D case, due to the long range correlations introduced in the tensor network. General PEPS (not preserving the canonical form) thanks to larger number of ancilla degrees of freedom per each tensor are not affected by this issue as severely, although their exact contraction comes at an exponential cost, due to which approximation techniques are used for that purpose. Nevertheless, recently different ways of *canonizing* PEPS were presented [31–34], leading to the 2D version of Isometric Tensor Networks (isoTNS).

In this thesis, we employ both 1D and 2D versions of Isometric Tensor Networks in large-scale numerical studies of different physical models, as well as noisy simulations of quantum circuits, characterized by a finite fidelity similar to the one manifested by the NISQ computers.

In Chapter 2 we review the basic concepts related to the formalism of tensor networks. We describe the properties of an MPS and show how it can be obtained, starting from a state vector, by means of *Singular Value Decomposition* (SVD) [28]. We explain how to *truncate* the degrees of freedom (referred to as the *bond-size*) of a given tensor, by keeping just the most relevant singular values obtained during the SVD. We illustrate different algorithms operating on MPSs, such as variational compression [28], *Time Evolving Block Decimation* (TEBD) [26, 28] and *Density Matrix Renormalization Group* (DMRG) [28, 35–38]. We conclude this chapter by showing how MPSs can be utilised to simulate 2D systems.

In Chapter 3 we study two chosen physical models using 1D methods. We localize the critical point of the transverse field Ising (TFI) model on a finite square lattice. Although the properties of an infinite version of this system were investigated in the past [39–42], we use results of our calculations as a benchmark, against which we compare techniques involving 2D tensor networks introduced in the further part of this work. Then, we examine the spin-3/2 XXZ Hamiltonian on a honeycomb lattice, which can serve as a model of the monolayer of CrI_3 [43]. We compare the predictions of magnetic phases manifested by this system, given by the classical approximation of the model, with the ones obtained via DMRG calculations.

In Chapter 4 we describe random quantum circuits used by Google Inc. in their famous quantum supremacy experiment [44]. We explain how MPSs can be used to simulate this class of systems and show that the fidelity of such calculations can be precisely approximated even in regimes for which exact classical simulation is intractable [45].

In Chapter 5 we review one of the techniques used to canonize PEPS and illustrate how to implement the 2D version of the Time Evolving Block Decimation ($TEBD^2$) algorithm using this formalism [32,34]. Then, we develop two modifications of the initial framework. First one

allows for a significant reduction of the bond-size required during calculations. The second one puts a constraint on given degrees of freedom of tensors stored in the grid, transforming it into a tree-like structure. We compare the two versions of the TEBD^2 algorithm on the task of obtaining the ground state of the TFI model through imaginary time evolution. We test the adjusted TEBD^2 algorithm on the task of simulation of random quantum circuits and demonstrate that the proposed alterations allow for an accurate approximation of the fidelity of such computations. We compare the results of our method with the ones given by simulation performed with the use of MPS-based techniques.

Finally, we conclude the thesis in Chapter 6 and provide suggestions for the future work.

SECOND CHAPTER

ISOMETRIC TENSOR NETWORKS IN 1D

We will begin this chapter with a comprehensive description of the tensor and tensor network concepts, followed by an introduction of a diagrammatic representation that makes it easy to visualize different kinds of tensors, and various operations that can be conducted on them. With this tool, we will define the most basic single-dimensional type of tensor network called *Matrix Product State* (MPS). By showing in detail the different ways in which an MPS can be obtained, we will introduce the notion of *normalization*, which, when used in the right way, allows for an immense increase in the speed of certain types of calculations. Then, we will demonstrate two ways of representing an operator that can be applied to such a tensor network. Finally, we will illustrate two algorithms that use these distinct representations of operators. The first method, called *Time Evolving Block Decimation* (TEBD), can be used to conduct both real and imaginary time evolution of an arbitrary quantum state stored in the form of an MPS, while the second one, named *Density Matrix Renormalization Group* (DMRG) allows to obtain the ground state of 1D physical systems.

We will end this chapter by showing how the properties of two-dimensional systems can be analyzed by means of one-dimensional MPS representation. This will be the building block for the structures and methods tailored specifically for the simulation of higher-dimensional systems, which will be discussed in detail in Chapter 5.

2.1 Basics

Tensor is a mathematical object that captures the concept of a *multilinear map* [46], i.e. a function of numerous variables $f : A_1 \times A_2 \times ... \times A_n \rightarrow B$, where $A_1, A_2, ..., A_n$ and B are vector spaces, with the condition of f being linear with respect to each A_i argument. A tensor can be represented by a multidimensional array, where the number of dimensions of such an object is described as *order*, or *rank*. Intuitively, the order of a tensor can be understood as a number of directions in which the array expands or the number of indices needed to explicitly define the location of each element in such an array. For example, the components of a two-dimensional array (which is also an order-2 tensor) could be denoted as $T_{i,j}$, where i and j are the aforementioned indices. However, indices may not only be displayed in the form of subscript, and the superscript notation is also the correct one. The location of the index tells us how it responds to the change of basis, which can be twofold. We will describe this difference shortly, but for that purpose we first need to introduce a general rule for the

contraction of two tensors. The contraction over a given index $j = 1, ..., n$ is defined as a sum of corresponding components multiplied together, and can only be performed if the index in question appears in both tensors once as a superscript and once as a subscript, i.e.

$$c_i^k = \sum_{j=1}^{n} a_i^j b_j^k = a_i^j b_j^k. \tag{2.1}$$

On the right hand side in the above equation we introduced the Einstein summation convention, which omits the summation sign. We will alternately use it with the standard notation, in which a subscript denotes iterating over a particular set. Both of these notations will be distinguishable based on the context in which they are employed (or the appropriate interpretation will be made clear in some circumstances). In special cases, we will also surround the index with square brackets, e.g., in $\hat{O}_{[i]}$, to emphasize the usual meaning of the index.

Knowing how we can contract two tensors, we can precisely describe the meaning of the subscript and superscript conventions for the index location. We will explain that on an example of an order-1 tensor, a vector. It can come in two forms, as a column vector (*contravariant*) and as a row vector (*covariant*). Let us define a $C_{i'}^i$ matrix that changes the basis in some arbitrary way, which can be written down as

$$\hat{e}_{i'} = e_i C_{i'}^i, \tag{2.2}$$

where e_i are the old basis vectors, while $\hat{e}_{i'}$ are the new ones. The components of a contravariant vector v in order to compensate for the basis change need to be transformed by a matrix that is an inverse of the $C_{i'}^i$ matrix,

$$\hat{v}^{i'} = (C^{-1})_i^{i'} v^i. \tag{2.3}$$

On the other hand, the components of a covariant vector w are transformed by the same matrix that carried out the change of the basis vectors,

$$\hat{w}_{i'} = w_i C_{i'}^i. \tag{2.4}$$

This reasoning can be extended to the case of higher-order tensors, for which there is a separate covariant or contravariant transformation operation for each of the tensor indices. Therefore, the $T^{i_1,i_2,...,i_n}_{j_i,j_2,...,j_m}$ tensor is transformed as follows

$$\hat{T}^{i_1',i_2',...,i_n'}_{j_i',j_2',...,j_m'} = (C^{-1})_{i_1}^{i_1'} \cdots (C^{-1})_{i_n}^{i_n'} T^{i_1,i_2,...,i_n}_{j_i,j_2,...,j_m} C_{j_1'}^{j_1} \cdots C_{j_m'}^{j_m}. \tag{2.5}$$

The $T^{i_1,i_2,...,i_n}_{j_i,j_2,...,j_m}$ tensor is sometimes described as an (n,m)-order one, however in this work we will refer to the order of the tensor as the total number of its dimensions. Thus, we will call $T^{i_1,i_2,...,i_n}_{j_i,j_2,...,j_m}$ the $(n+m)$-order tensor.

Although we have focused up to this point solely on the properties of a single tensor, the cautious reader may have noticed that we used multi-tensor operations for that purpose. Indeed, in Eqs. (2.1) to (2.5) we have already dealt with the most important concept of this work, i.e. *tensor network*, which is defined as a discrete set of tensors connected to each other by contractions. On the right-hand side of Eqs. (2.1) to (2.4) we can see contractions of just two tensors, which can be relatively easy to grasp, however the mathematical description of multiple tensors given for example in Eq. (2.5) becomes cumbersome and hard to track. In the following section we will see, how to diminish this hardness with an intuitive graphical tool.

2.1.1 Diagrammatic representation of tensors

Knowing the basic properties of tensors we can now define a formalism called *tensor diagram notation* (or *Penrose graphical notation*) [47]. Within this convention, we will represent each tensor with some geometric shape (usually we will use rectangles, ovals and diamonds) with a couple of line segments, called *legs* (or alternately, *wires* or *bonds*), attached to it. The number of legs is always equal to the order of a tensor, so each leg always resembles one of the tensor indices. With this approach, we can represent an order-0 tensor (scalar) as a shape without any legs, an order-1 tensor (vector) as a shape with one leg, and so on. We can see that this tool allows for easy visualization of higher-dimensional tensors, whose exact representation or mathematical description might be difficult to grasp. Examples of several tensors in diagrammatic notation are depicted in Fig. 2.1.

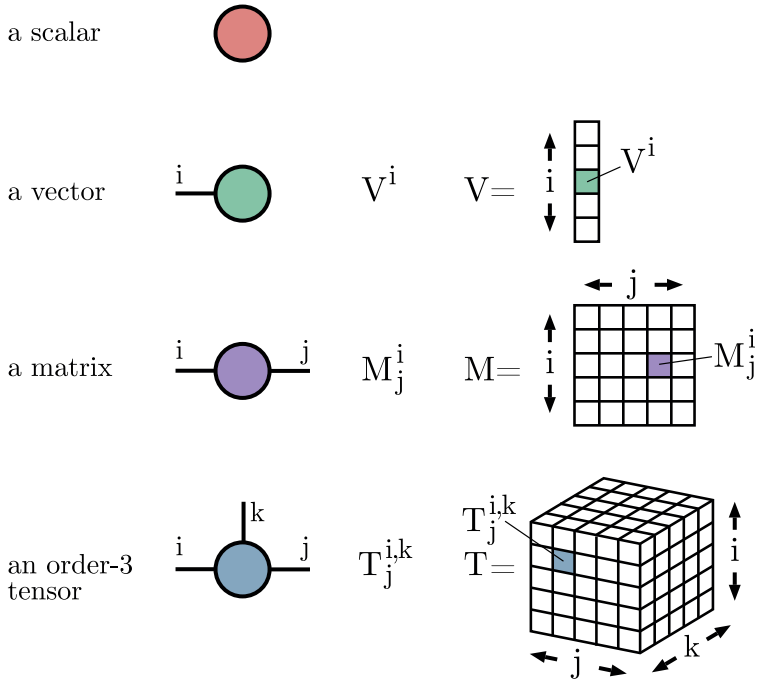


Figure 2.1: Examples of several low-dimensional tensors.

As we mentioned in Section 2.1, tensors are represented by multidimensional arrays. Such arrays can be freely reshaped by adding or removing dimensions, as long as the total number of elements stored in the initial and final structure is the same. For example, an order-3 tensor $T^i_{j_1,j_2}$ of size $3\times 2\times 2$ can be reshaped into a $T^i_{(j_1,j_2)}$ matrix of size 3×4. In this single example we introduced a number of notations that we will stick to in the rest of the work. Firstly, we will refer to the *size* of a given index as the upper bound on the set of numbers, over which it can iterate. Secondly, we will represent the *grouping* (or *combining*) of two indices by imply surrounding them with brackets. Going back to the example, we could end the whole reshaping procedure with the additional step of changing the label of the newly obtained index from (j_1, j_2) to j, and arrive simply at T^i_j.

Of course, the reverse operation, which aims to split one index into two is completely valid, and can be written down as a transition from T^i_j to $T^i_{j_1,j_2}$.

Both of the presented techniques are particularly easy to illustrate using the diagram notation, by replacing a single leg with two (splitting) or vice versa, by replacing two legs with one (grouping). These two operations are shown in Fig. 2.2.

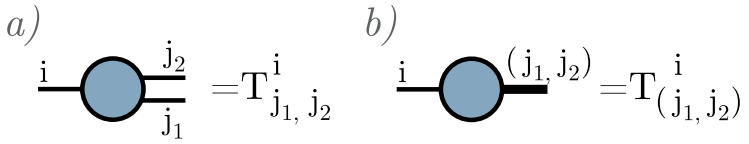


Figure 2.2: An example of combining two legs of a tensor. (a) $T^{i}_{j_1,j_2}$ before grouping. (b) $T^{i}_{(j_1,j_2)}$ after combination of j_1 and j_2 legs.

Apart from the possibility of splitting and combining the legs, they can also be bent or crossed. A leg that is bent in the opposite direction to its original one, can be understood as *Kronecker delta*. In turn, crossing two legs of a given tensor corresponds to swapping the indices assigned to those legs. Each of these operations can be viewed as a separate tensor, and their meaning is shown in Fig. 2.3.

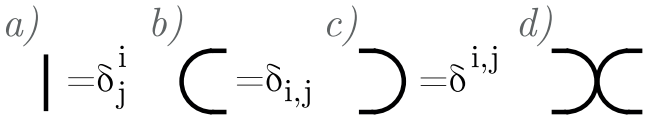


Figure 2.3: Legs transformations as standalone tensors. (a) Contraction or relabeling. (b,c) Kronecker deltas. (d) Swap operation [46].

By bending and crossing legs of a given tensor, we can arbitrarily rearrange its indices. For example, a M^{i}_{j} matrix can be transformed into $M^{j}_{i}, M_{i,j}, M_{j,i}, M^{i,j}$ and $M^{j,i}$. The sequence of operations that arrive at each of these representations is illustrated in Fig. 2.4.

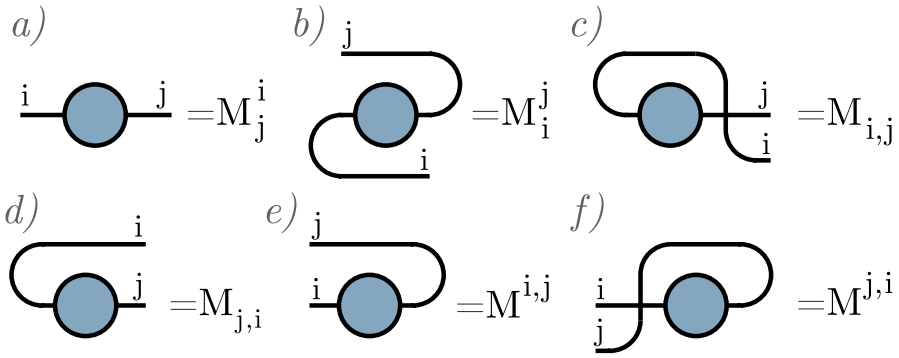


Figure 2.4: All possible rearrangements of indices of a matrix M^{i}_{j} [46].

Continuing the above reasoning, the *adjoint* of a tensor can be obtained by reversing the direction of its legs. If we combine this procedure with *complex conjugation* (which simply inverts the sign of the imaginary part of each complex number, being the constituent of the considered tensor) marked with a $*$ symbol standing next to the tensor label, we arrive at the operation of particular interest in the following sections of this chapter - the *Hermitian adjoint*. It will be commonly denoted with the † symbol. The relationship between all these three transformations is depicted in Fig. 2.5.

Knowing the rules governing bending and crossing of legs, the only operation left to be explained in Penrose notation is the joining of two bonds. Such connection of two legs corresponds to the contraction over the indices assigned to them. When summing over given indices, the legs in question disappear from the diagram, reducing the dimensionality of the resulting tensor. With this approach, we can easily represent such operations as matrix-vector

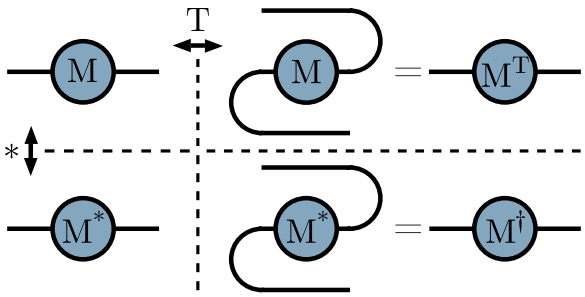


Figure 2.5: Relationships between transposition, conjugation and Hermitian adjoint of given tensor [46].

multiplication or even contraction of two tensors over multiple indices at once, which are shown in Fig. 2.6.

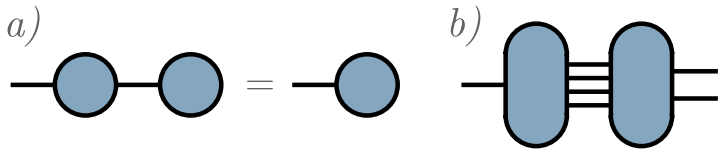


Figure 2.6: Examples of tensor contraction. (a) Multiplication of a vector by a matrix. (b) Contraction of two tensors over multiple bonds.

Moreover, we are not limited to joining legs that come solely from two different tensors. Nothing stands in the way of connecting two legs from a single tensor, as long as their sizes match. In this way, we can perform the operation of partial trace (shown in Fig. 2.7).

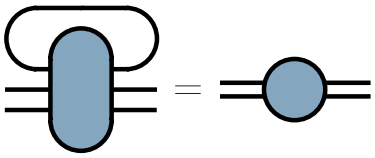


Figure 2.7: Partial trace conducted by contracting legs belonging to a single tensor.

Finally, we need to determine the cost of basic tensor contractions. For the two tensors A and B, each having legs of size $\{\chi_i^A\}$ and $\{\chi_j^B\}$, respectively, we define the cost of their contraction as

$$cost(A.B) = \prod_i \chi_i^A \prod_j \chi_j^B \Big/ \prod_{\chi \in A \cap B} \chi. \tag{2.6}$$

An useful intuition helping to easily calculate the cost given by the above formula, is to check how many legs are "touched" by the tensors to be contracted. Using this method we can determine that the contraction shown in Fig. 2.6.a would cost $\mathcal{O}(\chi^2)$, in Fig. 2.6.b $\mathcal{O}(\chi^7)$, and in Fig. 2.7 $\mathcal{O}(\chi^5)$, assuming that all legs of the tensors in question have the same size, equal to χ.

2.2 Singular Value Decomposition

Now that we have a solid foundation in tensor diagram notation, we can move on to more advanced algorithms and structures that use this formalism. We will start with a method called *Singular Value Decomposition* (SVD), which will be an essential part of all the techniques described in the sections to come.

SVD takes an $m \times n$ matrix M and decomposes it into three output matrices $M = USV^\dagger$. S is a $k \times k$ diagonal matrix, where $k = \min\{m, n\}$, storing real and non-negative *singular values* s_i, which are usually ordered in descending order. The U and $V^\dagger$ matrices have the shapes $m \times k$ and $k \times n$, respectively. They are called *semi-unitary matrices*, which means that they fulfil the *isometry condition* $U^\dagger U = V^\dagger V = \mathbf{1}$. Let us point out that in order to obtain the identity matrix, U should be multiplied by its Hermitian adjoint from the left, while $V^\dagger$ from the right. This is due to the fact that U has orthonormal columns and $V^\dagger$ has orthonormal rows.

This factorization procedure can be simply expressed using the Penrose notation as shown in Fig. 2.8.

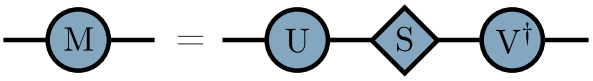


Figure 2.8: Diagrammatic representation of the SVD.

Above we used the diamond shape to represent S, to emphasize the fact that it is a diagonal matrix.

Since this method will be one of the crucial components of the algorithms described in this paper, it is extremely important to know its computational complexity. For an $m \times n$ matrix, the cost of SVD is equal to $\mathcal{O}(mn \cdot k)$, with $k = \min\{m, n\}$. We deliberately separated the number of singular values to be calculated from the size of the entire matrix on the left-hand side of the equation, with the intention of showing the correct intuition and making it simple to assess the complexity of SVD in the future.

In the context of quantum physics SVD is also used to derive the *Schmidt decomposition*. To understand it, let us denote by $|\psi\rangle$ the general pure state of a composite system consisting of subsystems A and B. Such a quantum state is commonly represented by a column vector

$$|\psi\rangle = \sum_{\alpha,\beta} \Psi^{(\alpha,\beta)} |\alpha\rangle_A |\beta\rangle_B, \tag{2.7}$$

where $\{|\alpha\rangle_A\}$ and $\{|\beta\rangle_B\}$ form orthonormal bases in their corresponding Hilbert spaces. Now, if we focus solely on the coefficients of $|\psi\rangle$, we can treat them as entries in a matrix (being the outcome of a vector reshape). As a result, each of these coefficients can be denoted as $\hat{\Psi}^{\alpha}_{\beta}$. If we further decompose this matrix using SVD and then restore the column vector form, we can rewrite the initial quantum state as

$$|\psi\rangle = \sum_{\gamma=1}^{k} s_\gamma |\gamma\rangle_A |\gamma\rangle_B, \tag{2.8}$$

where s_γ are called *Schmidt values*. The $\{|\gamma\rangle_A\}$ and $\{|\gamma\rangle_B\}$ sets are orthonormal, and can be extended to form the bases of their assigned Hilbert spaces. Now, if we express the reduced density matrices of A and B as follows

$$\hat{\rho}_A = \sum_{\gamma=1}^{k} s_\gamma^2 |\gamma\rangle_{A\,A}\langle\gamma|, \qquad \hat{\rho}_B = \sum_{\gamma=1}^{k} s_\gamma^2 |\gamma\rangle_{B\,B}\langle\gamma|, \tag{2.9}$$

we can see that due to the normalization condition of the density matrices (their trace must be equal to 1), we have $\sum_{\gamma=1}^{k} s_\gamma^2 = 1$. From this we can immediately see that Schmidt coefficients are just square roots of the eigenvalues of a given diagonalisable matrix, while the corresponding eigenvectors of the $\hat{\rho}_A$ and $\hat{\rho}_B$ are the left and right singular vectors, respectively [28].

Due to this fact, we can use singular values to analyze the entanglement between A and B subsystems. For that purpose we can use the von Neumann entropy

$$S_{(A|B)}(|\psi\rangle) = -Tr\,\hat{\rho}_A \log \hat{\rho}_A = -Tr\,\hat{\rho}_B \log \hat{\rho}_B = -\sum_{\gamma=1}^{k} s_\gamma^2 \log s_\gamma^2, \tag{2.10}$$

or a more general Renyi entropy

$$S^{\alpha}_{(A|B)}(|\psi\rangle) = \frac{1}{1-\alpha} \log\Big(\sum_k s_k^{2\alpha}\Big). \tag{2.11}$$

In Eq. (2.11) the α appearing as the superscript should not be treated as one of the indices introduced in the previous section, but rather as a signature denoting the value of parameter used to calculate the Renyi entropy.

The number of non-zero Schmidt values is called *Schmidt rank*, and directly distinguishes the product states (in the above case $k = 1$) from the entangled ones ($k > 1$).

2.3 Matrix Product States

We have already demonstrated that a quantum state, which is often represented as a column vector, may be transformed into a matrix while constructing the Schmidt decomposition. Furthermore, this reshaping procedure made it easy for us to acquire information about the entanglement between its A and B subsystems. We can continue this reasoning by increasing the number of subsystems, as it takes place, e.g., in a chain of particles of length L. In the further part of this work we will refer to each subsystem of such a chain as *node* (or *site*). In general, each node might represent either a single particle or a grouped set of particles. We unify these two scenarios by focusing only on the size of a local Hilbert space of each node, which will be denoted as p_i, for $i = 1, ..., L$.

2.3.1 Generating MPSs

Using the above convention we will introduce an algorithm allowing for the generation of an MPS [28]. We can write down an arbitrary quantum state of a chain of length L as

$$|\psi\rangle = \sum_{p_1,p_2,...,p_L} \Psi^{(p_1,p_2,...,p_L)} |p_1, p_2, ..., p_L\rangle. \tag{2.12}$$

Let us once again focus exclusively on the coefficients of this vector, and apply reshaping similar to the one given in previous section. We begin with "detaching" the first subsystem from the remaining ones and as a result form a matrix, which is subsequently decomposed with the use of SVD

$$\Psi^{(p_1,p_2,...,p_L)} = \hat{\Psi}^{p_1}_{(p_2,...,p_L)} = U^{p_1}_{a_1} S^{a_1}_{a_1} (V^\dagger)^{a_1}_{(p_2,...,p_L)}. \tag{2.13}$$

If we perform the multiplication of the $S^{a_1}_{a_1}$ and $(V^\dagger)^{a_1}_{(p_2,...,p_L)}$ matrices (the purpose of which will be explained shortly) to obtain a $\Theta^{a_1}_{(p_2,...,p_L)}$ matrix, we can rewrite the coefficients of the quantum state under study as

$$U^{p_1}_{a_1} S^{a_1}_{a_1} (V^\dagger)^{a_1}_{(p_2,...,p_L)} = U^{p_1}_{a_1} \Theta^{a_1}_{(p_2,...,p_L)}. \tag{2.14}$$

Now, in order to "detach" p_2 from $p_3, ..., p_L$ we can conduct on $\Theta^{a_1}_{(p_2,...,p_L)}$ a similar operation to the one presented in Eq. (2.13), with the only difference being the combination of the a_1 and p_2 indices. The result of this procedure can be written as

$$U^{p_1}_{a_1} \Theta^{a_1}_{(p_2,...,p_L)} = U^{p_1}_{a_1} \hat{\Theta}^{(a_1,p_2)}_{(p_3,...,p_L)} = U^{p_1}_{a_1} U^{(a_1 p_2)}_{a_2} S^{a_2}_{a_2} (V^\dagger)^{a_2}_{(p_3,...,p_L)}. \tag{2.15}$$

By repeating the steps of multiplication of the S and $V^\dagger$ matrices, followed by proper reshaping, and finally SVD, we can represent the coefficients of the considered quantum state as follows

$$\Psi^{(p_1,p_2,...,p_L)} = U^{p_1}_{a_1} U^{(a_1,p_2)}_{a_2} U^{(a_2,p_3)}_{a_3} \dots U^{(a_{L-2},p_{L-1})}_{a_{L-1}} \Theta^{a_{L-1}}_{p_L}. \tag{2.16}$$

We can see that in order to calculate the coefficients of $|\psi\rangle$, represented as in Eq. (2.16), it is necessary to compute the product of matrices, from which this structure has taken its name - *Matrix Product State* [24–28].

Each matrix given in Eq. (2.16), except the ones lying on the edges, can be reshaped into an order-3 tensor, by simply splitting the (a_{i-1}, p_i) leg into two separate ones - a_{i-1} and p_i

$$U^{p_1}_{a_1} U^{(a_1,p_2)}_{a_2} U^{(a_2,p_3)}_{a_3} \dots U^{(a_{L-2},p_{L-1})}_{a_{L-1}} \Theta^{a_{L-1}}_{p_L} = A^{p_1}_{a_1} A^{a_1,p_2}_{a_2} A^{a_2,p_3}_{a_3} \dots A^{a_{L-2},p_{L-1}}_{a_{L-1}} \Theta^{a_{L-1}}_{p_L}. \tag{2.17}$$

From this picture it is easy to see that if we were to conduct the contraction over all $a_{i'}$ legs, we would obtain a tensor with remaining $p_1, p_2, ..., p_L$ legs. The p_i indices cannot be removed from an MPS, because they serve as the formal description of the state of a physical system under study. On the other hand, the $a_{i'}$ legs introduce some additional degrees of freedom to each tensor stored in an MPS, which disappear when we perform a summation over a given $a_{i'}$ leg. In order to illustrate the dual nature of these indices, we shall refer to p_i legs as the *physical* ones, while $a_{i'}$ will be denoted as *virtual* ones.

Diagram notation makes it simple to show all of the procedures that have been covered in this section up until this point. To obtain an MPS we begin with a vector Ψ having one p leg, describing the physical degrees of freedom of all the nodes included in the considered chain. We split the p leg into $p_1, p_2, ..., p_L$ ones, which consists of replacing one leg with L new ones. Now, we can group the $p_2, p_3, ..., p_L$ legs on one side of the tensor, leaving the remaining p_1 on the other side. This tensor, being in fact a matrix, is further factorized with the use of SVD, giving us U, S and $V^\dagger$ matrices (depicted by an oval, a diamond, and another oval), connected with legs labeled as a_1. Subsequently, contraction of S with $V^\dagger$ gives us new Θ tensor, having one virtual leg a_1 and $p_2, p_3, ..., p_L$ physical ones (grouped so far as a single leg). We finalize the procedure by "detaching" the p_2 leg from all the physical ones and combine it with a_1, to prepare Θ for the next SVD.

In summary, the algorithm consists of $L-1$ SVDs interspersed with proper tensor reshaping. All of the steps involved in this method are shown in Fig. 2.9.

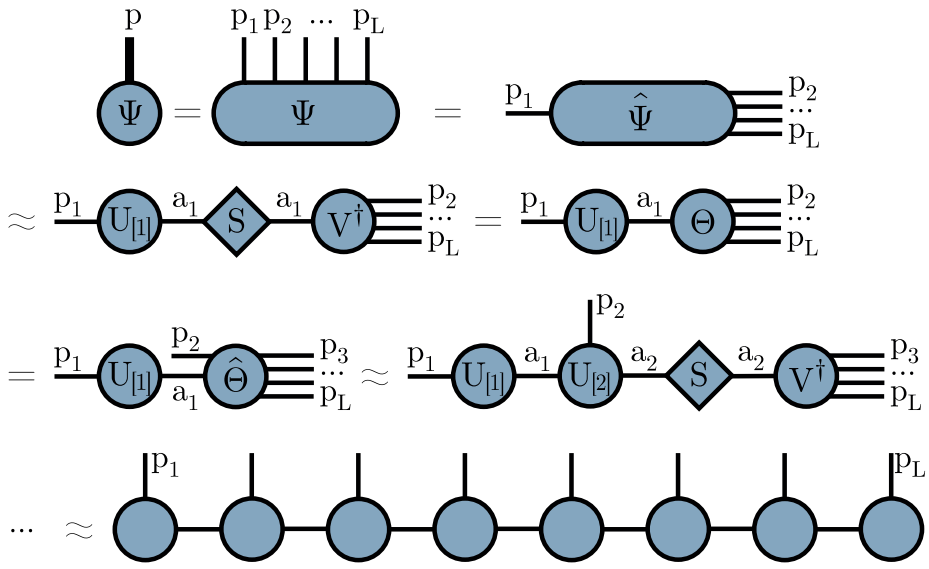


Figure 2.9: Transformation of a vector into an MPS by a series of SVDs.

Although the Einstein summation convention eases the process of writing formulae involving several tensors, a proper description of steps presented in Eqs. (2.12) to (2.17) is still a difficult task. We have seen that the diagram notation alleviates this hardness and makes it simple and understandable to depict complex algorithms utilizing tensor operations.

2.3.2 Canonical form of an MPS

Let us focus now on the properties of the MPS obtained in Eq. (2.16). Any transformation of the initial vector is only worthwhile, if we can benefit computationally from this new representation. By examining some of the physical properties of the chain under study we can in fact see a huge numerical gain. We will discuss this on the example of calculating the norm of the state. Using the bracket notation, we would calculate the norm of a vector as $\langle\psi|\psi\rangle$. In the MPS language, this can be expanded as

$$\begin{aligned}\langle\psi|\psi\rangle =&(\Theta^\dagger)^{a'_{L-1},p'_L}(A^\dagger)^{a'_{L-2}}_{a'_{L-1},p'_{L-1}}\dots(A^\dagger)^{a'_2}_{a'_3,p'_3}(A^\dagger)^{a'_1}_{a'_2,p'_2}(A^\dagger)_{a'_1,p'_1}\\ &A^{p_1}_{a_1}A^{a_1,p_2}_{a_2}A^{a_2,p_3}_{a_3}\dots A^{a_{L-2},p_{L-1}}_{a_{L-1}}\Theta^{a_{L-1}}_{p_L}.\end{aligned} \tag{2.18}$$

In Section 2.2 we have seen that U matrices had the $U^\dagger U = \mathbf{1}$ feature. If we apply this property to the innermost A tensors in Eq. (2.18), which emerge from reshaping of U matrices, we may obtain

$$\begin{aligned}&(\Theta^\dagger)^{a'_{L-1},p'_L}(A^\dagger)^{a'_{L-2}}_{a'_{L-1},p'_{L-1}}\dots(A^\dagger)^{a'_2}_{a'_3,p'_3}(A^\dagger)^{a'_1}_{a'_2,p'_2}(A^\dagger)_{a'_1,p'_1}A^{p_1}_{a_1}A^{a_1,p_2}_{a_2}A^{a_2,p_3}_{a_3}\dots A^{a_{L-2},p_{L-1}}_{a_{L-1}}\Theta^{a_{L-1}}_{p_L}\\ &=(\Theta^\dagger)^{a'_{L-1},p'_L}(A^\dagger)^{a'_{L-2}}_{a'_{L-1},p'_{L-1}}\dots(A^\dagger)^{a'_2}_{a'_3,p'_3}(A^\dagger)^{a'_1}_{a'_2,p'_2}(A^\dagger)_{a'_1,p'_1}\delta^{p'_1}_{p_1}A^{p_1}_{a_1}A^{a_1,p_2}_{a_2}A^{a_2,p_3}_{a_3}\dots A^{a_{L-2},p_{L-1}}_{a_{L-1}}\Theta^{a_{L-1}}_{p_L}\\ &=(\Theta^\dagger)^{a'_{L-1},p'_L}(A^\dagger)^{a'_{L-2}}_{a'_{L-1},p'_{L-1}}\dots(A^\dagger)^{a'_2}_{a'_3,p'_3}(A^\dagger)^{a'_1}_{a'_2,p'_2}(A^\dagger)_{a'_1,p_1}A^{p_1}_{a_1}A^{a_1,p_2}_{a_2}A^{a_2,p_3}_{a_3}\dots A^{a_{L-2},p_{L-1}}_{a_{L-1}}\Theta^{a_{L-1}}_{p_L}\\ &=(\Theta^\dagger)^{a'_{L-1},p_L}(A^\dagger)^{a'_{L-2}}_{a'_{L-1},p_{L-1}}\dots(A^\dagger)^{a'_2}_{a'_3,p_3}(A^\dagger)^{a'_1}_{a'_2,p_2}\delta_{a'_1,a_1}A^{a_1,p_2}_{a_2}A^{a_2,p_3}_{a_3}\dots A^{a_{L-2},p_{L-1}}_{a_{L-1}}\Theta^{a_{L-1}}_{p_L}\end{aligned}$$

$$= (\Theta^\dagger)^{a'_{L-1},p_L}(A^\dagger)^{a'_{L-2}}_{a'_{L-1},p_{L-1}} \dots (A^\dagger)^{a'_2}_{a'_3,p_3}(A^\dagger)^{a'_1}_{a'_2,p_2}\hat{A}^{p_2}_{a'_1,a_2}A^{a_2,p_3}_{a_3}\dots A^{a_{L-2},p_{L-1}}_{a_{L-1}}\Theta^{a_{L-1}}_{p_L}. \tag{2.19}$$

Above we utilized the $\delta^{p'_1}_{p_1}$ (presented in Fig. 2.3.a), which merely relabels the legs. By doing this, we highlighted the fact that we were contracting the physical legs of two tensors. To improve readability, further δ operators of this type were suppressed. Assuming the inherent relabeling, we will omit these operators in the remaining part of this work when contracting the physical legs.

Due to the fact that U matrix resulting from an SVD is a semi-unitary matrix, we were able to omit the contraction of $(A^\dagger)_{a'_1,p'_1}$ and $A^{p_1}_{a_1}$ tensors. Moreover, the same rule applies to the remaining A tensors in the MPS, which allows us to reduce the above equation in the following fashion

$$\begin{aligned}
&(\Theta^\dagger)^{a'_{L-1},p'_L}(A^\dagger)^{a'_{L-2}}_{a'_{L-1},p'_{L-1}} \dots (A^\dagger)^{a'_2}_{a'_3,p'_3}(A^\dagger)^{a'_1}_{a'_2,p'_2}\hat{A}^{p_2}_{a'_1,a_2}A^{a_2,p_3}_{a_3}\dots A^{a_{L-2},p_{L-1}}_{a_{L-1}}\Theta^{a_{L-1}}_{p_L}\\
&= (\Theta^\dagger)^{a'_{L-1},p'_L}(A^\dagger)^{a'_{L-2}}_{a'_{L-1},p'_{L-1}} \dots (A^\dagger)^{a'_2}_{a'_3,p'_3}\delta_{a'_2,a_2}A^{a_2,p_3}_{a_3}\dots A^{a_{L-2},p_{L-1}}_{a_{L-1}}\Theta^{a_{L-1}}_{p_L}\\
&\vdots\\
&= (\Theta^\dagger)^{a'_{L-1},p'_L}\delta_{a'_{L-1},a_{L-1}}\Theta^{a_{L-1}}_{p_L} = (\Theta^\dagger)^{a'_{L-1},p'_L}\hat{\Theta}_{a'_{L-1},p_L} = n.
\end{aligned} \tag{2.20}$$

Since Θ tensors often do not meet the isometry condition, we are forced to undertake their contraction, which yields the scalar n, being the norm that we were trying to calculate.

At this stage, a number of observations should be taken. First, we observed from Eqs. (2.18) to (2.20) that, because of how we constructed the MPS, we may ignore contractions of all the tensors to the left of the Θ tensor when computing some property of the MPS. The A tensors shall hereafter be referred to as *left-normalized* ones. Secondly, the contraction of Θ and $\Theta^\dagger$ tensors could not be omitted, because Θ was not obtained from a semi-unitary matrix. This fact also explains, why we multiplied the S and $V^\dagger$ matrices in Eq. (2.14), giving as a result Θ. Another set of semi-unitary matrices may be produced via SVD of Θ, which would make it possible to skip the computation of tensor products in the final form of the MPS. On the other hand, factorization a matrix that is already a semi-unitary one is futile, because it does not provide any numerical advantage in the future. To emphasize this special property of the Θ tensor, we will refer to it as the *orthogonality center*. Finally, we will call all MPSs consisting of left-normalized tensors followed by the ortohogonality center, as the *left-canonical* ones.

To highlight the normalization feature of given tensor we will add arrows to its legs, indicating the direction of its normalization. Additionally, we will follow the convention of assigning an incoming arrow to each physical leg. With this extended formalism we can illustrate a left-canonical MPS as shown in Fig. 2.10.

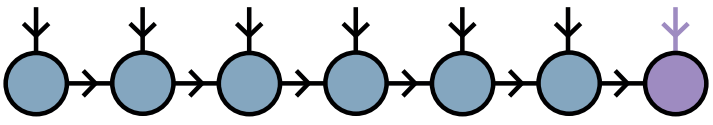


Figure 2.10: MPS in the left-canonical form.

The Hermitian adjoint of a given tensor will be depicted by a tensor mirrored along the horizontal line, with the legs directions reversed. With this improved framework the diagrammatic representation of the norm computation given in Eqs. (2.18) to (2.20) also becomes particularly clear, and is illustrated in Fig. 2.11.

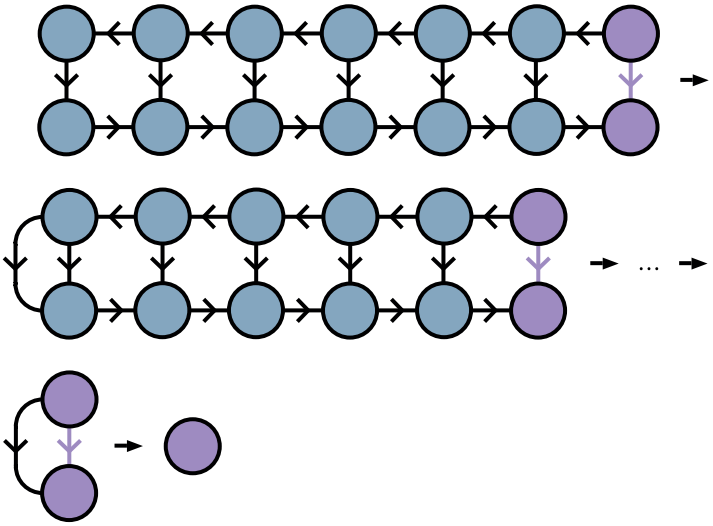


Figure 2.11: Calculation of the norm of an MPS in the left-canonical form.

So far, in the process of generating an MPS we were contracting the S and $V^\dagger$ matrices, resulting from consecutive SVDs. Nothing, however, prevents the contraction of S with the U matrix, rather than the $V^\dagger$ one. In this way, we could "detach" the physical legs starting from the right edge of the initial tensor and arrive at

$$\begin{aligned}
\Psi^{(p_1,p_2,\ldots,p_L)} &= \hat{\Psi}^{(p_1,p_2,\ldots,p_{L-1})}_{p_L} = U^{(p_1,p_2,\ldots,p_{L-1})}_{a_{L-1}} S^{a_{L-1}}_{a_{L-1}} (V^\dagger)^{a_{L-1}}_{p_L} \\
&= \Theta^{(p_1,p_2,\ldots,p_{L-1})}_{a_{L-1}} (V^\dagger)^{a_{L-1}}_{p_L} \\
&= \hat{\Theta}^{(p_1,p_2,\ldots,p_{L-2})}_{(p_{L-1},a_{L-1})} (V^\dagger)^{a_{L-1}}_{p_L} \\
&= U^{(p_1,p_2,\ldots,p_{L-2})}_{a_{L-2}} S^{a_{L-2}}_{a_{L-2}} (V^\dagger)^{a_{L-2}}_{(p_{L-1},a_{L-1})} (V^\dagger)^{a_{L-1}}_{p_L} \\
&\vdots \\
&= \Theta^{p_1}_{a_1} (V^\dagger)^{a_1}_{(p_2,a_2)} \ldots (V^\dagger)^{a_{L-2}}_{(p_{L-1},a_{L-1})} (V^\dagger)^{a_{L-1}}_{p_L}.
\end{aligned} \tag{2.21}$$

Again, with proper reshaping we can convert all $V^\dagger$ tensors as follows

$$\Theta^{p_1}_{a_1} (V^\dagger)^{a_1}_{(p_2,a_2)} \ldots (V^\dagger)^{a_{L-2}}_{(p_{L-1},a_{L-1})} (V^\dagger)^{a_{L-1}}_{p_L} = \Theta^{p_1}_{a_1} B^{a_1,p_2}_{a_2} \ldots B^{a_{L-2},p_{L-1}}_{a_{L-1}} B^{a_{L-1}}_{p_L}. \tag{2.22}$$

The newly acquired B tensors are *right-normalized* since they are the final outcome of the transformation of $V^\dagger$ matrices. This suggests that if we were to recalculate the MPS norm, we could do so without multiplying any of the tensors assigned to nodes from 2 to L, and instead just conduct the contraction of the left-most Θ with $\Theta^\dagger$. We will call all MPSs in the form presented in Eq. (2.22) as the *right-canonical* ones, and a diagram representation of such a state is illustrated in Fig. 2.12.

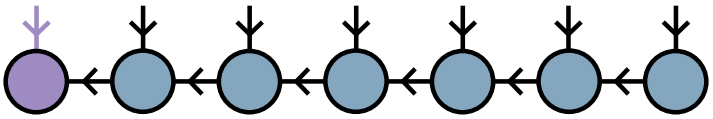


Figure 2.12: MPS in the right-canonical form.

There is one final note to be made in the context of MPS normalization. In the case of an MPS in the left-canonical form, we have seen on an example of norm calculation that we could ignore the contraction of all $L-1$ tensors located to the left from the orthogonality center, and conduct only the contraction of the rightmost tensor with its Hermitian adjoint. The same would be true, if we wanted to calculate some other feature of the right-most site, e.g., the expectation value of particle spin in the Z axis. In Section 2.4, we will go into more depth about how this can be accomplished, but for now, let us just say that in that case we could also ignore all of the left-normalized A tensors, and focus solely on the Θ tensor. The right-normalized MPS would likewise exhibit all of these features, although in this scenario we would omit the contraction of all B tensors located to the right from the orthogonality center.

However, no matter which of the two presented forms of MPS we would choose, if we wanted to examine some property on any of the sites located in the bulk, we would be forced to conduct some additional contractions, which we would like to avoid. To evade this issue we will present another technique, allowing for the *shifting of the orthogonality center*.

This can be also achieved by means of the SVD. Let us assume that we were given an MPS in the left-canonical form and wanted to shift the orthogonality center to some tensor located in the bulk. We begin with the multiplication of the two right-most tensors

$$A^{p_1}_{a_1} A^{a_1,p_2}_{a_2} \dots A^{a_{L-3},p_{L-2}}_{a_{L-2}} A^{a_{L-2},p_{L-1}}_{a_{L-1}} \Theta^{a_{L-1}}_{p_L} = A^{p_1}_{a_1} A^{a_1,p_2}_{a_2} \dots A^{a_{L-3},p_{L-2}}_{a_{L-2}} \Theta^{(a_{L-2},p_{L-1})}_{p_L}. \tag{2.23}$$

Now, if we use SVD to factorize the newly obtained Θ, we essentially arrive at the second-to-last step of the procedure generating an MPS in the left-canonical form. Although this time, we contract S with the U matrix instead of $V^\dagger$, giving us

$$\begin{aligned} A^{p_1}_{a_1} A^{a_1,p_2}_{a_2} \dots A^{a_{L-3},p_{L-2}}_{a_{L-2}} \Theta^{(a_{L-2},p_{L-1})}_{p_L} &= A^{p_1}_{a_1} A^{a_1,p_2}_{a_2} \dots A^{a_{L-3},p_{L-2}}_{a_{L-2}} U^{(a_{L-2},p_{L-1})}_{a_{L-1}} S^{a_{L-1}}_{a_{L-1}} (V^\dagger)^{a_{L-1}}_{p_L} \\ &= A^{p_1}_{a_1} A^{a_1,p_2}_{a_2} \dots A^{a_{L-3},p_{L-2}}_{a_{L-2}} \Theta^{a_{L-2},p_{L-1}}_{a_{L-1}} B^{a_{L-1}}_{p_L}. \end{aligned} \tag{2.24}$$

In the last transformation, in addition to multiplying U and S, we also reshaped the Θ tensor and relabeled the $V^\dagger$ one (knowing that it is a right-normalized tensor). We can see that the MPS presented in Eq. (2.24) incorporates both the left-normalized A tensors and the right-normalized B ones. For this reason, we shall refer to this type of MPS form as *mixed-canonical*.

The orthogonality center can be moved to any location along the chain using a sequence of consecutive SVDs, giving us the most general form of an MPS in the mixed-canonical form

$$A^{p_1}_{a_1} A^{a_1,p_2}_{a_2} \dots A^{a_{L-3},p_{L-2}}_{a_{L-2}} \Theta^{a_{L-2},p_{L-1}}_{a_{L-1}} B^{a_{L-1}}_{p_L} = A^{p_1}_{a_1} \dots A^{a_{i-2},p_{i-1}}_{a_{i-1}} \Theta^{a_{i-1},p_i}_{a_i} B^{a_i,p_{i+1}}_{a_{i+1}} \dots B^{a_{L-1}}_{p_L}. \tag{2.25}$$

Using the same logic as before, when calculating the norm of an MPS in the mixed-canonical form, it is sufficient to multiply only Θ by $\Theta^\dagger$, disregarding the tensors A and B. We would also proceed in a similar way in the case of analyzing any property of the ith node.

The diagrammatic representation of a mixed-canonical MPS along with the procedure of shifting of the orthogonality center and norm calculation are illustrated in Fig. 2.13.

2.3.3 Vidal representation

The canonical form, while powerful, has its limitations. For instance, if the orthogonality center is located at the right edge of the chain and we would like to measure some property of

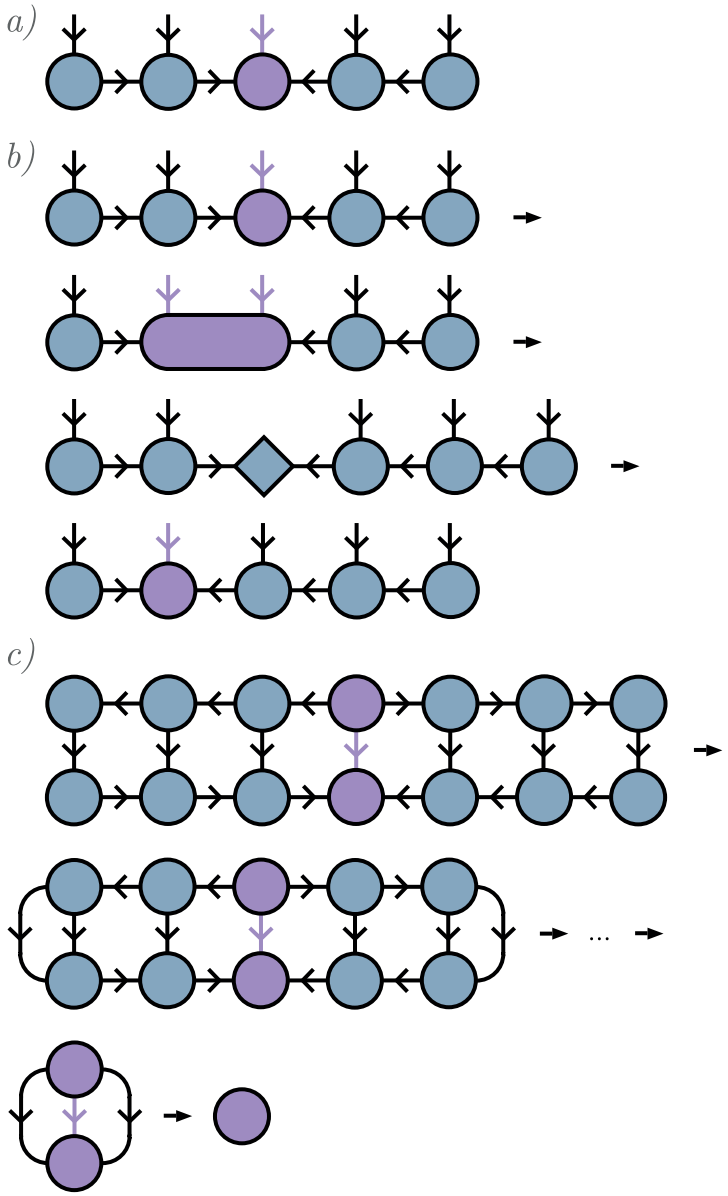


Figure 2.13: (a) MPS in the mixed-canonical form. (b) Shifting of the orthogonality center. (c) Norm calculation in the case of an MPS in the mixed-canonical form.

the left-most site, a sequence of SVDs would need to be run along the entire chain. However, we can obtain a slightly different representation of an MPS, called Vidal representation [25], which alleviates this problem.

Once again, let us assume that we were given a state vector which we would like to transform into a tensor network, in a fashion similar to the one illustrated while generated an MPS in left-canonical form. After the first SVD we would obtain the following state

$$\Psi^{(p_1,p_2,\ldots,p_L)} = \hat{\Psi}^{p_1}_{(p_2,\ldots,p_L)} = U^{p_1}_{a_1} S^{a_1}_{a_1} (V^\dagger)^{a_1}_{(p_2,\ldots,p_L)}. \tag{2.26}$$

This time, before incorporating the S matrix into $V^\dagger$ we are copying the singular values obtained during SVD. Let us call these Schmidt values as $S_{[1]}$ to emphasize the fact that they were generated during the first factorization in the whole procedure. The following contraction leads to

$$U^{p_1}_{a_1} S^{a_1}_{a_1} (V^\dagger)^{a_1}_{(p_2,\ldots,p_L)} = U^{p_1}_{a_1} \Theta^{a_1}_{(p_2,\ldots,p_L)}, \tag{2.27}$$

which is further factorized, resulting in

$$U^{p_1}_{a_1} \Theta^{a_1}_{(p_2,\ldots,p_L)} = U^{p_1}_{a_1} \hat{\Theta}^{(a_1,p_2)}_{(p_3,\ldots,p_L)} = U^{p_1}_{a_1} U^{(a_1,p_2)}_{a_2} S^{a_2}_{a_2} (V^\dagger)^{a_2}_{(p_3,\ldots,p_L)}. \tag{2.28}$$

Knowing the next steps of the procedure, we can see that the already obtained U tensors will not be involved in any other SVD, and as a result will not change at any point in the further part of the algorithm. We can utilize this fact and multiply the $U^{(a_1 p_2)}_{a_2}$ from the left by $1/S_{[1]}$, which can be understood as a simple division by the singular values obtained during the first factorization. This operation gives us the following state

$$U^{p_1}_{a_1} U^{(a_1,p_2)}_{a_2} S^{a_2}_{a_2} (V^\dagger)^{a_2}_{(p_3,\ldots,p_L)} = \Gamma^{p_1}_{a_1} S^{a_1}_{a_1} \Gamma^{(a_1,p_2)}_{a_2} S^{a_2}_{a_2} (V^\dagger)^{a_2}_{(p_3,\ldots,p_L)}, \tag{2.29}$$

where we denoted by Γ tensors, which may not necessarily be isometries (with the exception of the left-most tensor). By repeating the steps outlined above $L-1$ times, we can see the following pattern to emerge

$$\begin{aligned}
\Psi^{(p_1,p_2,\ldots,p_L)} &= U^{p_1}_{a_1} S^{a_1}_{a_1} (V^\dagger)^{a_1}_{(p_2,\ldots,p_L)} = \Gamma^{p_1}_{a_1} \Theta^{a_1}_{(p_2,\ldots,p_L)} \\
&= \Gamma^{p_1}_{a_1} U^{(a_1,p_2)}_{a_2} S^{a_2}_{a_2} (V^\dagger)^{a_2}_{(p_3,\ldots,p_L)} = \Gamma^{p_1}_{a_1} S^{a_1}_{a_1} \Gamma^{(a_1,p_2)}_{a_2} \Theta^{a_2}_{(p_3,\ldots,p_L)} \\
&= \Gamma^{p_1}_{a_1} S^{a_1}_{a_1} \Gamma^{(a_1,p_2)}_{a_2} U^{(a_2,p_3)}_{a_3} S^{a_3}_{a_3} (V^\dagger)^{a_3}_{(p_4,\ldots,p_L)} = \Gamma^{p_1}_{a_1} S^{a_1}_{a_1} \Gamma^{(a_1,p_2)}_{a_2} S^{a_2}_{a_2} \Gamma^{(a_2,p_3)}_{a_3} \Theta^{a_3}_{(p_4,\ldots,p_L)} \\
&= \ldots = \Gamma^{p_1}_{a_1} S^{a_1}_{a_1} \Gamma^{(a_1,p_2)}_{a_2} S^{a_2}_{a_2} \Gamma^{(a_2,p_3)}_{a_3} \ldots U^{(a_{L-2},p_{L-1})}_{a_{L-1}} S^{a_{L-1}}_{a_{L-1}} (V^\dagger)^{a_{L-1}}_{p_L} \\
&= \Gamma^{p_1}_{a_1} S^{a_1}_{a_1} \Gamma^{(a_1,p_2)}_{a_2} S^{a_2}_{a_2} \Gamma^{(a_2,p_3)}_{a_3} \ldots S^{a_{L-2}}_{a_{L-2}} \Gamma^{(a_{L-2},p_{L-1})}_{a_{L-1}} S^{a_{L-1}}_{a_{L-1}} \Gamma^{a_{L-1}}_{p_L},
\end{aligned} \tag{2.30}$$

where in the final step, we took advantage of the fact that the $V^\dagger$ tensor already had the appropriate form, thus no matrix division was necessary.

By reshaping the bulk Γ matrices into order-3 tensors and renaming all the S matrices to Λ, to match the widely used convention, we arrive at the final form of the MPS in the Vidal representation

$$\Psi^{(p_1,p_2,\ldots,p_L)} = \Gamma^{p_1}_{a_1} \Lambda^{a_1}_{a_1} \Gamma^{a_1,p_2}_{a_2} \Lambda^{a_2}_{a_2} \Gamma^{a_2,p_3}_{a_3} \ldots \Lambda^{a_{L-2}}_{a_{L-2}} \Gamma^{a_{L-2},p_{L-1}}_{a_{L-1}} \Lambda^{a_{L-1}}_{a_{L-1}} \Gamma^{a_{L-1}}_{p_L}, \tag{2.31}$$

which is also illustrated in Fig. 2.14 with the use of Penrose notation.

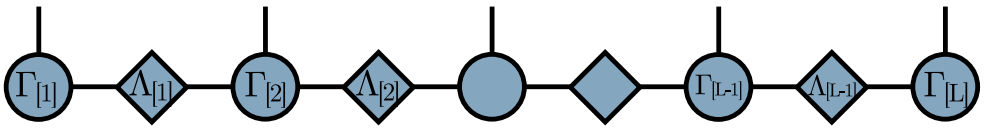


Figure 2.14: MPS in the Vidal representation.

Now, in order to obtain the orthogonality center at an arbitrary site of the chain, it is sufficient to multiply the corresponding tensor by left and right singular values

$$\Gamma^{p_1}_{a_1} \ldots \Gamma^{a_{i-2},p_{i-1}}_{a_{i-1}} (\Lambda^{a_{i-1}}_{a_{i-1}} \Gamma^{a_{i-1},p_i}_{a_i} \Lambda^{a_i}_{a_i}) \Gamma^{a_i,p_{i+1}}_{a_{i+1}} \ldots \Gamma^{a_{L-1}}_{p_L} \rightarrow \Gamma^{p_1}_{a_1} \ldots \Gamma^{a_{i-2},p_{i-1}}_{a_{i-1}} \Theta^{a_{i-1},p_i}_{a_i} \Gamma^{a_i,p_{i+1}}_{a_{i+1}} \ldots \Gamma^{a_{L-1}}_{p_L}. \tag{2.32}$$

Moreover, we can convert between Vidal representation and the canonical form in a very convenient way. By simply incorporating the Schmidt values from the left and right sides, we may obtain the A and B tensors, respectively. We can write this down explicitly, as

$$\Gamma^{p_1}_{a_1}(\Lambda^{a_1}_{a_1}\Gamma^{a_1,p_2}_{a_2})(\Lambda^{a_2}_{a_2}\Gamma^{a_2,p_3}_{a_3})\dots(\Lambda^{a_{L-1}}_{a_{L-1}}\Gamma^{a_{L-1}}_{p_L}) \rightarrow A^{p_1}_{a_1}A^{a_1,p_2}_{a_2}A^{a_2,p_3}_{a_3}\dots\Theta^{a_{L-1}}_{p_L} \tag{2.33}$$

and

$$(\Gamma^{p_1}_{a_1}\Lambda^{a_1}_{a_1})\dots(\Gamma^{a_{L-3},p_{L-2}}_{a_{L-2}}\Lambda^{a_{L-2}}_{a_{L-2}})(\Gamma^{a_{L-2},p_{L-1}}_{a_{L-1}}\Lambda^{a_{L-1}}_{a_{L-1}})\Gamma^{a_{L-1}}_{p_L} \rightarrow \Theta^{p_1}_{a_1}\dots B^{a_{L-3},p_{L-2}}_{a_{L-2}}B^{a_{L-2},p_{L-1}}_{a_{L-1}}B^{a_{L-1}}_{p_L}. \tag{2.34}$$

Of course it is also possible to convert Vidal representation into a mixed-canonical one by combining the final two methods described. The transformation from canonical form to Vidal representation can also be achieved, by simply reversing the steps taken above, however without the prior knowledge of the singular values at each bond it would involve a sequence of SVDs.

Division by the singular values is the final topic worth addressing in the context of the representation shown in this section. Firstly, this operation is correctly defined, because we are not storing the Schmidt values equal to 0, in order to decrease the bond dimensions of the MPS. Secondly, division by very small singular values might lead to some numerical errors, like the loss of normalization of the whole state. However, this issue can be simply fixed by disregarding the entries which are smaller than some given threshold, like 10^{-14}.

2.3.4 MPS bond-sizes

Let us finish the analysis of the properties of an arbitrary MPS, with a thorough investigation of the bond-sizes of its tensors. Let us suppose, for the sake of simplicity that the chain under study consists of even number of sites, each having a physical leg of size d. If we were to generate an MPS in the left-canonical form, we would begin the whole procedure with a vector of size d^L. Then, we would reshape it into a $d \times d^{L-1}$ matrix, and further factorize it with the use of SVD, arriving at matrices U_1, S_1 and $V_1^\dagger$, where we added the subscript to highlight how many SVDs were used to generate the matrices in question. By the properties of SVD, U_1, S_1 and $V_1^\dagger$ are of sizes $d \times d$, $d \times d$ and $d \times d^{L-1}$, respectively. We can immediately relabel the U_1 matrix as A_1, because it already has desired dimensions and no further reshaping is required. Multiplication of S_1 and $V_1^\dagger$, giving as a result Θ_1, does not change the bond-sizes, so Θ_1 has shape $d \times d^{L-1}$. We reshape Θ_1 to "detach" one physical leg from the remaining ones, and combine it with the already existing virtual leg, resulting in a $d^2 \times d^{L-2}$ matrix $\hat{\Theta}_1$. Decomposition of $\hat{\Theta}_1$ gives U_2, S_2 and $V_2^\dagger$ of sizes $d^2 \times d^2$, $d^2 \times d^2$ and $d^2 \times d^{L-2}$, respectively. We could reshape U_2 into an order-3 tensor to clearly distinguish the physical legs from the virtual ones, giving as a result a $d \times d \times d^2$ A_2 tensor. We can see now that repeated application of SVD followed by reshaping will result in a sequence of tensors with exponentially increasing sizes $d \times d$, $d \times d \times d^2$, $d^2 \times d \times d^3$ and so on. However, this will only happen until we reach the central bond of the chain. In this step, the matrix to be factorized would be of size $d^{L/2} \times d^{L/2}$ (with the first dimension being the result of multiplication of the virtual bond-size $d^{L/2-1}$ with the physical leg size d), so the bonds connecting three resulting matrices $U_{L/2}$, $S_{L/2}$ and $V_{L/2}^\dagger$ would be of size $d^{L/2}$. After the incorporation of singular values into the $V_{L/2}^\dagger$ we would be left with the largest bond in the whole MPS, of size $d^{L/2}$. In the following step, the matrix to be factorized would be of size $d^{L/2+1} \times d^{L/2-1}$, so the minimal value among the dimensions involved would be equal to $d^{L/2-1}$, which would result in a smaller bond-size than the one

obtained in the previous step. We would see this decrease in virtual bond-sizes throughout the whole second part of the MPS, with the final bond connecting sites $L-1$ and L, being of size d.

Using the Penrose notation makes all the steps described above much clearer. In Fig. 2.15 we show once again the procedure of generating an MPS in the left-canonical form, however this time we are extending the picture by adding the sizes of all legs. Thanks to this we can predict the bond-size resulting form a given SVD by simply choosing the smaller value among the bond-sizes of the tensor to be factorized.

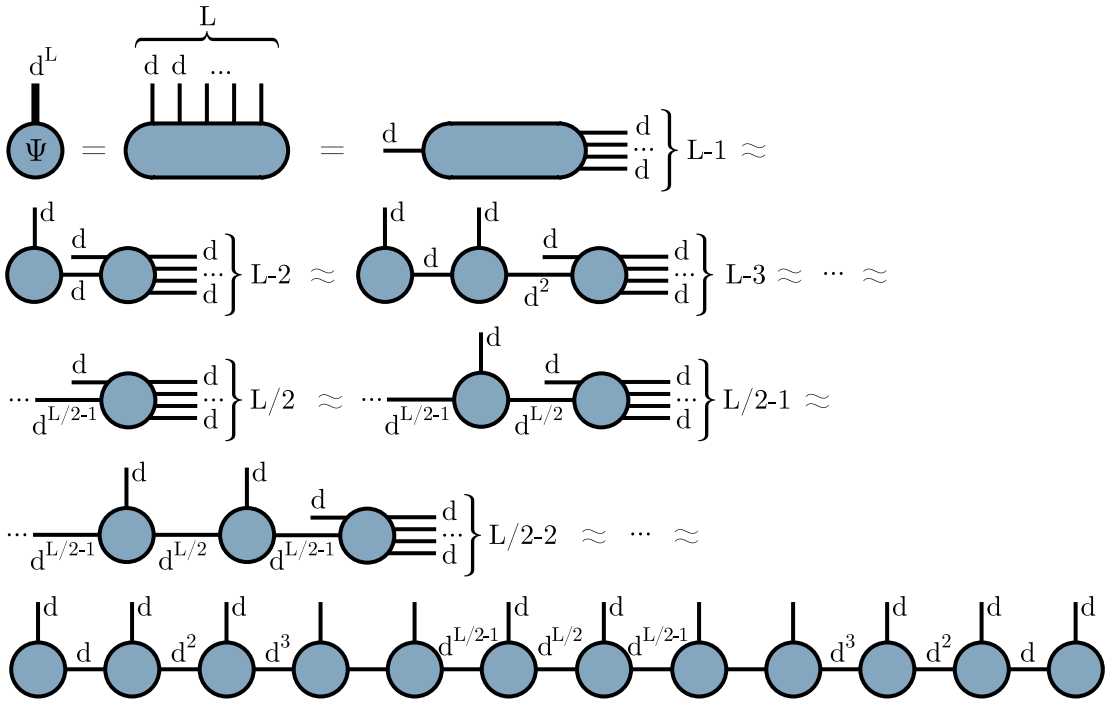


Figure 2.15: MPS generation starting from a state stored as a vector.

If we closely examine the steps involved in the above scheme, we can notice two things. Firstly, after the generation of an MPS is finished, we can arbitrarily contract two neighbouring tensors and split them once again with the use of SVD, without any increase in the bond-size connecting them. Despite the apparent senselessness of this technique, it has serious implications in the two-dimensional case, where a similar procedure would result in an increase in the bond dimension. For these kinds of tensor networks an extra care is needed to preserve the desired structure of legs sizes, which will be explained in more detail in Chapter 5.

Secondly, if we compare the sizes of the initial vector and the final tensor network, generated by this naive algorithm, we can see that the latter one uses much more parameters to encode the same state! Let us recall that the state vector was of size d^L. However, the two central tensors in an MPS have shapes $d^{L/2-1} \times d \times d^{L/2}$ and $d^{L/2} \times d \times d^{L/2-1}$, respectively, so just these two objects take twice as much memory as the initial vector. To address this issue, SVDs are usually supplemented with *truncations*, which consist of discarding all but the first χ singular values, where χ is a parameter set upfront launching any procedure operating on an MPS. Singular values obtained from an SVD are always ordered in a non-ascending order, thanks to which in the process of removing all entries with indices greater than χ, we are effectively getting rid of the values that carry the least amount of information. Alternatively, a given *cutoff* level can be chosen. All singular values that are smaller than this threshold are

removed, similarly as in the case in which we are keeping only the first χ entries. An example of such a truncation is depicted in Fig. 2.16.

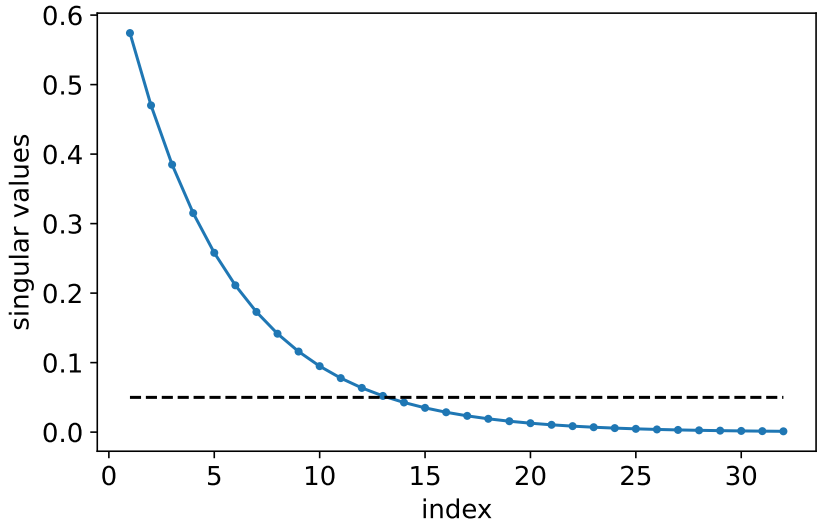


Figure 2.16: Example set of singular values with a chosen cutoff level.

When we preserve the first χ singular values during truncation, we are actually keeping the entries that carry the most entanglement. It has been shown that ground states of 1D systems, obeying the so called *area law* [48, 49], can be efficiently represented by MPSs with constant bond-size χ. The area law states that the boundary area, rather than the volume, determines how much entanglement entropy is present in a given region of the system. In 1D this corresponds to a situation, in which after undergoing some initial growth, the scaling of entanglement present in the ground state of gapped, local systems (with exponentially decaying correlation lengths), becomes constant. Moreover, it has been shown that the complexity of representing ground states of 1D critical systems with the use of MPSs also scales polynomially with the number of sites [50].

However, the above properties do not hold for 2D systems represented via MPSs. For example, for a rectangular lattice of size $Lx \times Ly$, and a gapped Hamiltonian, the entanglement entropy is proportional to $\min\{Lx, Ly\}$, which entails a large bond-size χ. Furthermore, with gapless models this scaling becomes noticeably worse.

Regardless of the type of system under study, the *error* associated with each truncation performed on an MPS can be calculated as the sum of discarded Schmidt values

$$\epsilon_{trunc} = \sum_{i>\chi} s_i^2. \tag{2.35}$$

Equivalently, we can define the *fidelity* of such an operation as

$$f_{trunc} = 1 - \epsilon_{trunc} = 1 - \sum_{i>\chi} s_i^2, \tag{2.36}$$

where we used the fact that the squares of all singular values must sum up to 1.

2.4 Diagram representation of operators

We showed in the last section how tensor networks may be used to represent any given quantum state. Now, we will look at various methods for expressing operators that can be applied to the aforementioned states, using the same formalism.

2.4.1 Single-site operators

We begin with the most basic kind of operator that acts only on a single site in the chain. In fact, this operator (in the form of a matrix) does not need any kind of special transformation, and already can be expressed in the tensor diagram formalism as a shape with two legs. We will follow the convention assigning the incoming arrow to the first leg, and an outgoing arrow to the second one. The application of such an operator to an MPS is also particularly simple, because it relies solely on its contraction with a corresponding tensor from the MPS. Both the single-site operator and its application to an MPS are shown in Fig. 2.17.

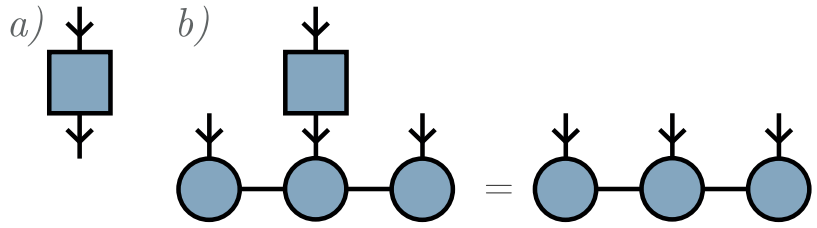


Figure 2.17: (a) Graphical representation of a single-site operator. (b) Application of an operator to an MPS. The arrows on horizontal bonds were omitted, as in the case of single-site operators it is not necessary to preserve the canonical form of the MPS.

We can notice that the application of such an operator does not increase any bond dimension in the tensor network, thus it does not generate any entanglement in the state. Although it can introduce some shuffling, which might further lead to more entanglement being generated after the application of a multi-site operator (which will be introduced in the following section).

At this point, let us elaborate on the norm calculation example from the previous section and examine how the expectation value of a single-site operator can be calculated, when the quantum state is expressed as an MPS. If the mentioned MPS is in a mixed-canonical form, e.g., the expectation value of an $\hat{O}_{[i]}$ operator ($\hat{O}$ operator acting on ith site) can be written down as

$$
\begin{aligned}
\langle\psi|\hat{O}_{[i]}|\psi\rangle =& (B^\dagger)^{a'_{L-1},p'_L}\dots(B^\dagger)^{a'_i}_{a'_{i+1},p'_{i+1}}(\Theta^\dagger)^{a'_{i-1}}_{a'_i,p'_i}(A^\dagger)^{a'_{i-2}}_{a'_{i-1},p'_{i-1}}\dots(A^\dagger)_{a'_1,p'_1}O^{p_i}_{p'_i}\\
& A^{p_1}_{a_1}\dots A^{a_{i-2},p_{i-1}}_{a_{i-1}}\Theta^{a_{i-1},p_i}_{a_i}B^{a_i,p_{i+1}}_{a_{i+1}}\dots B^{a_{L-1}}_{p_L}\\
=& (B^\dagger)^{a'_{L-1},p'_L}\dots(B^\dagger)^{a'_i}_{a'_{i+1},p'_{i+1}}(\Theta^\dagger)^{a'_{i-1}}_{a'_i,p'_i}O^{p_i}_{p'_i}(A^\dagger)^{a'_{i-2}}_{a'_{i-1},p_{i-1}}\dots(A^\dagger)_{a'_1,p_1}\\
& A^{p_1}_{a_1}\dots A^{a_{i-2},p_{i-1}}_{a_{i-1}}\Theta^{a_{i-1},p_i}_{a_i}B^{a_i,p_{i+1}}_{a_{i+1}}\dots B^{a_{L-1}}_{p_L}\\
=& (B^\dagger)^{a'_{L-1},p'_L}(B^\dagger)^{a'_{L-2}}_{a'_{L-1},p'_{L-1}}\dots(B^\dagger)^{a'_i}_{a'_{i+1},p'_{i+1}}\left((\Theta^\dagger)^{a'_{i-1}}_{a'_i,p'_i}O^{p_i}_{p'_i}\delta_{a'_{i-1},a_{i-1}}\Theta^{a_{i-1},p_i}_{a_i}\right)\\
& B^{a_i,p_{i+1}}_{a_{i+1}}\dots B^{a_{L-2},p_{L-1}}_{a_{L-1}}B^{a_{L-1}}_{p_L}
\end{aligned}
$$

$$
\begin{aligned}
&= (B^\dagger)^{a'_{L-1},p'_L}(B^\dagger)^{a'_{L-2}}_{a'_{L-1},p'_{L-1}} \dots (B^\dagger)^{a'_i}_{a'_{i+1},p'_{i+1}} \left((\Theta^\dagger)^{a'_{i-1}}_{a'_i,p'_i} O^{p_i}_{p'_i} \hat{\Theta}^{p_i}_{a'_{i-1},a_i}\right) \\
&B^{a_i,p_{i+1}}_{a_{i+1}} \dots B^{a_{L-2},p_{L-1}}_{a_{L-1}} B^{a_{L-1}}_{p_L} \\
&= (B^\dagger)^{a'_i}_{a'_{i+1},p'_{i+1}} \dots (B^\dagger)^{a'_{L-2}}_{a'_{L-1},p'_{L-1}} (B^\dagger)^{a'_{L-1},p'_L} B^{a_{L-1}}_{p_L} B^{a_{L-2},p_{L-1}}_{a_{L-1}} \dots B^{a_i,p_{i+1}}_{a_{i+1}} \\
&\left((\Theta^\dagger)^{a'_{i-1}}_{a'_i,p'_i} O^{p_i}_{p'_i} \hat{\Theta}^{p_i}_{a'_{i-1},a_i}\right) \\
&= \delta^{a'_i,a_i} (\Theta^\dagger)^{a'_{i-1}}_{a'_i,p'_i} O^{p_i}_{p'_i} \hat{\Theta}^{p_i}_{a'_{i-1},a_i} = (\hat{\Theta}^\dagger)^{a'_{i-1},a_i}_{p'_i} O^{p_i}_{p'_i} \hat{\Theta}^{p_i}_{a'_{i-1},a_i} = o.
\end{aligned} \tag{2.37}
$$

To better reflect the order in which the actual contractions would occur, we permuted several tensors in the second and penultimate line above. It should be emphasized that in general changing the order of, e.g., matrix multiplications, can only be reversed if those matrices commute. However, by using Einstein notation we are explicitly indicating, which contraction is conducted at any particular moment, which justifies the tensor reordering.

As expected, the whole expression resulted in a scalar o, which is the expectation value of an operator $\hat{O}_{[i]}$. As previously, all of the steps presented in Eq. (2.37) become particularly easy to follow, when we use Penrose notation, and are shown in Fig. 2.18.

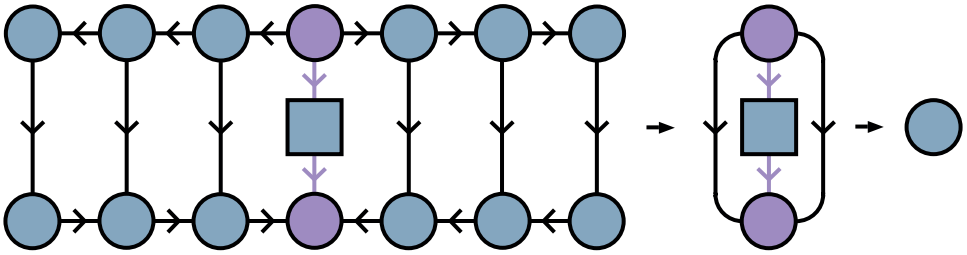


Figure 2.18: Calculation of single-site expectation value of an MPS in the mixed-canonical form.

Single-site operators might also be used to calculate the long-range correlations. In that case, we would simply apply two operators to a pair of the chain sites, which in general do not need to be adjacent. Nevertheless, in that scenario the contraction of all tensors between these two sites in question needs to be performed (both in the ket and bra states) with the additional condition that the orthogonality center must be one of these tensors. This procedure is illustrated in Fig. 2.19.

2.4.2 Multi-site operators

As in the case of single-site operators, we begin the construction of a multi-site operator from its matrix definition. As previously, this tensor can be depicted as a shape with two legs, the former incoming, and the latter outgoing one. Now, the only transformation that needs to be performed on this tensor is splitting of legs, to match the sizes of physical legs of sites. For example, an operator acting on two spin$-1/2$ sites can be expressed as a 4×4 matrix, which should be reshaped into an order-4 tensor of size $2 \times 2 \times 2 \times 2$. Of course we can also define operators acting on more than just two sites, which would differ only in the number of legs. Examples of some multi-site operators are shown in Fig. 2.20.

Application of such operators to an MPS, or calculation of their expectation values, is very similar to the case of single-site operators. The only difference lies in the number of tensors involved. We can begin the whole procedure by contracting the sites of the MPS, resulting in

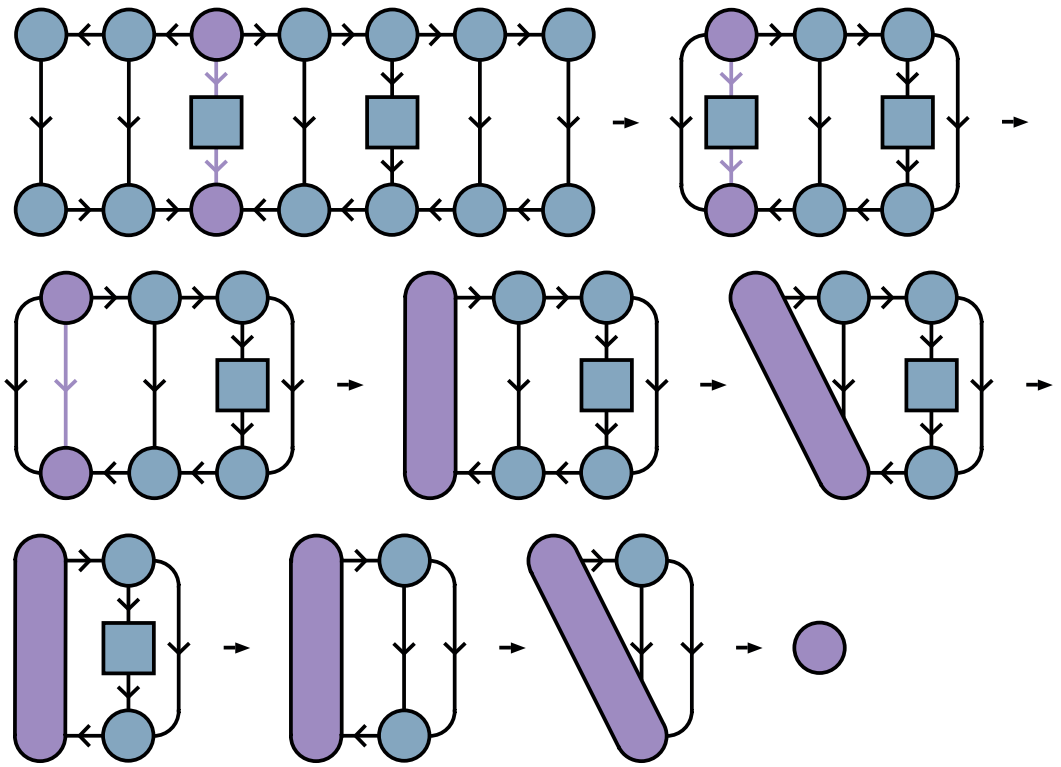


Figure 2.19: Calculation of long-range correlations of an MPS in the mixed-canonical form.

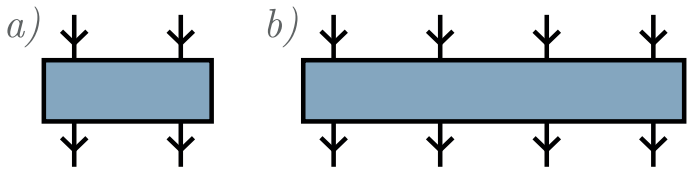


Figure 2.20: Examples of (a) a two-site operator, and (b) a four-site one.

a multi-site orthogonality center. To preserve the properties of the canonical form we need to make sure that one of the tensors being contracted is the initial single-site orthogonality center, while the remaining tensors must form its immediate surroundings. This operation is particularly easy in the case of just two sites, when it is sufficient to contract the orthogonality center with one of its neighbouring nodes, before proceeding to the contraction with the operator. After the application of such an operator is finished, we can restore the initial structure of the MPS, with one tensor per node, by means of SVD. An example of application of a 2-site operator and the calculation of its expectation value is shown in Fig. 2.21.

As can be seen from Fig. 2.21, the SVD conducted in order to restore the initial structure of the MPS leads to an increase in the bond-size, which corresponds to entanglement being generated in the state. To keep the memory footprint constant, we can truncate the bond in question, or run a compression algorithm (which will be introduced later on).

Let us exemplify the construction of a 2-site operator on an explicit model. For that purpose, we choose the *transverse field Ising* (TFI) model, whose Hamiltonian is defined as follows

$$\hat{H} = -J \sum_{i=1}^{L-1} \hat{\sigma}_i^x \hat{\sigma}_{i+1}^x - g \sum_{i=1}^{L} \hat{\sigma}_i^z. \tag{2.38}$$

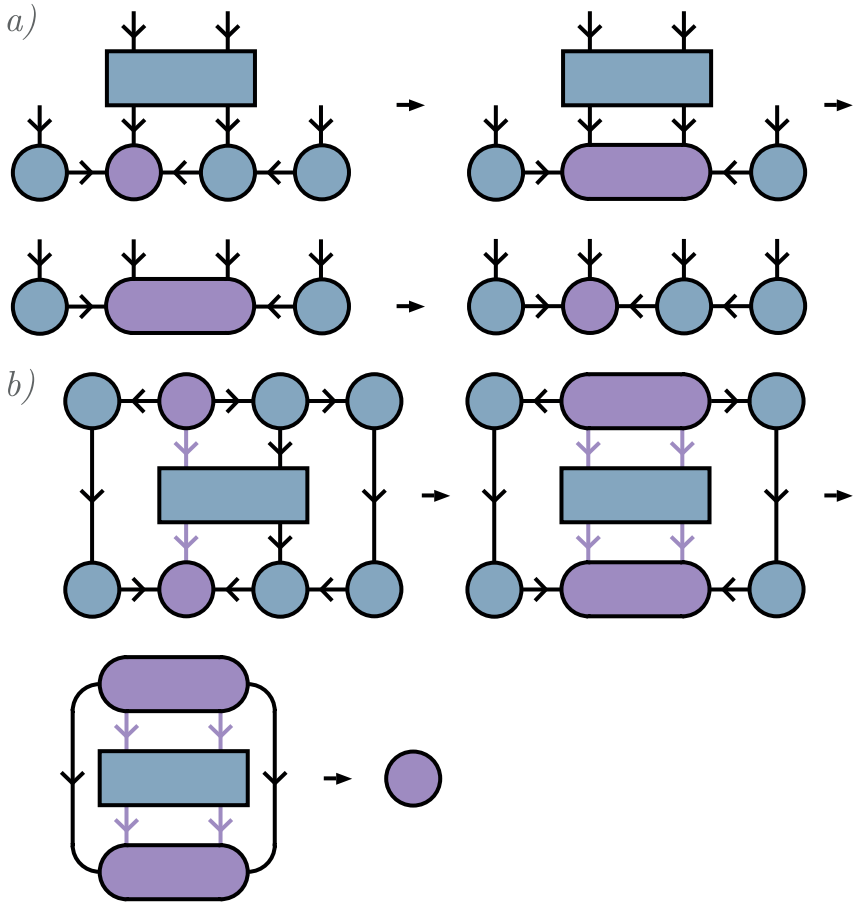


Figure 2.21: (a) Application of a two-site operator on an MPS. (b) Calculation of an expectation value with the use of a multi-site operator.

Assuming our chain consist of only 2 sites, above abbreviated formula can be expanded as

$$\hat{H} = -J(\hat{\sigma}^x \otimes \hat{\sigma}^x) - g(\hat{\sigma}^z \otimes \mathbb{1} + \mathbb{1} \otimes \hat{\sigma}^z). \tag{2.39}$$

Let us firstly focus on the interaction term. By doing explicit matrix multiplication we get

$$-J(\hat{\sigma}^x \otimes \hat{\sigma}^x) = -J \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \otimes \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} = \begin{pmatrix} 0 & 0 & 0 & -J \\ 0 & 0 & -J & 0 \\ 0 & -J & 0 & 0 \\ -J & 0 & 0 & 0 \end{pmatrix}. \tag{2.40}$$

The resulting matrix could be already reshaped into an order-4 tensor, giving us the desired two-site operator representing the interaction between the two sites. To obtain the whole Hamiltonian under study, this two-site operator could be supplemented with two single-site ones, describing single site terms. However, if we were to calculate the expectation value of energy of a given state for such an operator, represented by three separate tensors, we would have to conduct more contractions in order to arrive at the desired result. We can simplify this operation by incorporating the single-site terms into the bond-operator given in Eq. (2.40). To get the most general form possible, we can also assume that the external field acting on the two sites (given by the g variable) differs, and indicate its strength by g_1 and g_2, for the first and second sites, respectively. This gives us

$$-J(\hat{\sigma}^x \otimes \hat{\sigma}^x) - g_1(\hat{\sigma}^z \otimes \mathbb{1}) - g_2(\mathbb{1} \otimes \hat{\sigma}^z) = \begin{pmatrix} -g_1 - g_2 & 0 & 0 & -J \\ 0 & -g_1 + g_2 & -J & 0 \\ 0 & -J & g_1 - g_2 & 0 \\ -J & 0 & 0 & g_1 + g_2 \end{pmatrix}. \tag{2.41}$$

The final step to be made is to reshape the matrix given in Eq. (2.41) into an order-4 tensor (a matrix of matrices)

$$\begin{pmatrix} -g_1 - g_2 & 0 & 0 & -J \\ 0 & -g_1 + g_2 & -J & 0 \\ 0 & -J & g_1 - g_2 & 0 \\ -J & 0 & 0 & g_1 + g_2 \end{pmatrix} \rightarrow \begin{pmatrix} \begin{pmatrix} -g_1 - g_2 & 0 \\ 0 & -J \end{pmatrix} \begin{pmatrix} 0 & -g_1 + g_2 \\ -J & 0 \end{pmatrix} \\ \begin{pmatrix} 0 & -J \\ g_1 - g_2 & 0 \end{pmatrix} \begin{pmatrix} -J & 0 \\ 0 & g_1 + g_2 \end{pmatrix} \end{pmatrix}. \tag{2.42}$$

Although the above operator was designed with just two sites in mind, in combination with the canonical form of the MPS it can be used to calculate the expectation value of energy in the TFI model for a chain of arbitrary length. The procedure would consist of just two steps, which should be repeated in a *sweep* going over all bonds in the MPS. We could begin with an MPS in the right-canonical form, in which, for the record, the orthogonality center is located on the left-most site of the chain. We could calculate the bond-energy for these two sites utilizing the fact that contractions of all of the tensors from the third onward could be omitted. Then, we could shift the orthogonality center to the second site and calculate again the energy of the next two sites. By conducting this procedure for $L-1$ times we would obtain energies assigned to each virtual bond in the whole MPS. Then, the total energy would be simply the sum of all the component energies. The whole sweeping operation is show in Fig. 2.22.

The method described in the last paragraph needs just one last adjustment to work properly. Due to the way in which the whole algorithm works, we would measure twice the single-site terms contributing to the bond-energies. To remedy that it is sufficient to divide by 2 all g_i variables for the sites in the bulk of the chain, leaving g_1 and g_L unchanged.

2.4.3 Matrix Product Operators

The techniques shown above for the generation of operators as tensor networks although correct, have their limitations. For example, efficient representation of long-range interactions with this approach becomes problematic. Furthermore, it is desirable to express the operators as a series of connected tensors, with just one tensor per site of the chain, similar to how we created the MPS representations of quantum states.

By generalizing the formula for an arbitrary MPS defined Eq. (2.16), we can express the coefficients of an operator $\hat{O}$ as

$$O^{(p_1,p_2,\ldots,p_L)}_{(p'_1,p'_2,\ldots,p'_L)} = W^{p_1}_{p'_1,a_1} W^{p_2,a_1}_{p'_2,a_2} \ldots W^{p_{L-1},a_{L-2}}_{p'_{L-1},a_{L-1}} W^{p_L,a_{L-1}}_{p'_L}. \tag{2.43}$$

We can see that similarly to the case of an MPS, to obtain given coefficient of $\hat{O}$ it is necessary to conduct a product of matrices, hence the name *Matrix Product Operator* (MPO) [37, 51–53]. A graphical depiction of an MPO is shown in Fig. 2.23.

While SVD-based methods for the effective generation of MPOs have been proposed [54], we will demonstrate an alternative strategy based on the so-called *Finite State Machines* (FSM)

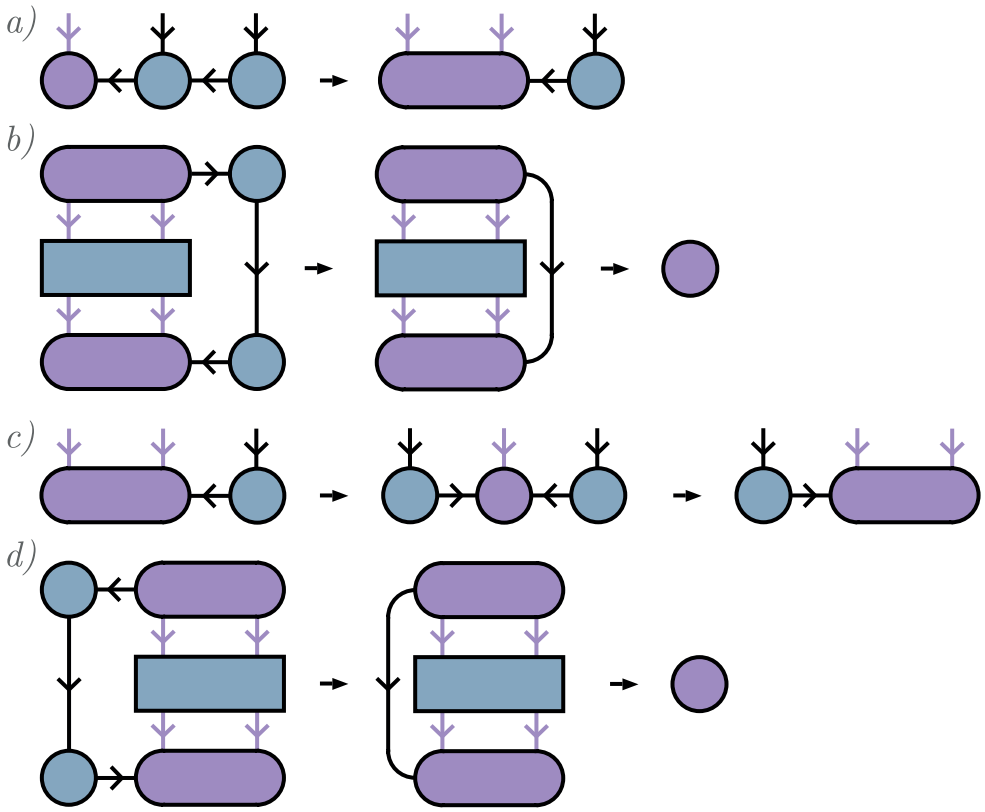


Figure 2.22: Calculation of the expectation value of energy with the use of just a single two-site operator. (a) Generation of multi-site orthogonality center. (b) Calculation of the first bond energy. (c) Shifting of the orthogonality center. (d) computation of the second bond energy.

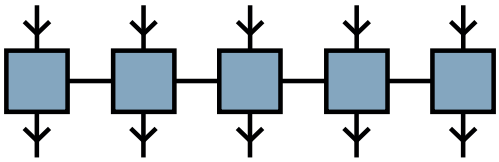


Figure 2.23: Example of an MPO.

[55]. We will present it on the example of the Heisenberg XXZ model, whose Hamiltonian is defined as

$$\hat{H} = \sum_{i=1}^{L-1} \left(\frac{J}{2} \hat{S}_i^+ \hat{S}_{i+1}^- + \frac{J}{2} \hat{S}_i^- \hat{S}_{i+1}^+ + J^z \hat{S}_i^z \hat{S}_{i+1}^z \right) - h \sum_{i=1}^{L} \hat{S}_i^z. \tag{2.44}$$

The above abbreviated formula can be of course expanded as

$$\begin{aligned} \hat{H} = & \frac{J}{2} \hat{S}_1^+ \otimes \hat{S}_2^- \otimes \mathbb{1} \otimes \ldots + \frac{J}{2} \hat{S}_1^- \otimes \hat{S}_2^+ \otimes \mathbb{1} \otimes \ldots + J^z \hat{S}_1^z \otimes \hat{S}_2^z \otimes \mathbb{1} \otimes \ldots \\ & - h \hat{S}_1^z \otimes \mathbb{1} \otimes \mathbb{1} \otimes \ldots - \mathbb{1} \otimes h \hat{S}_2^z \otimes \mathbb{1} \otimes \ldots + \\ & \ldots \end{aligned} \tag{2.45}$$

Now, in order to define the Hamiltonian of the whole system, we will introduce an FSM, which would act in a similar fashion to the Turing machine traversing a given tape. The FSM will have a set of internal states, just like the Turing machine, however in this comparison

we will replace the tape with a string of operators appearing in $\hat{H}$, each acting on a local Hilbert space. As a starting point for our FSM we choose the left-most edge of the string, from which we will proceed rightwards. The FSM begins in the R (ready) state and can transition to another state by inserting a specific operator in the string. Let us assume that there are four additional states: 1, 2, 3, which will be the "intermediate" states, and the final one F. Starting at the first position in the string, we can insert four different operators: $\mathbb{1}$ (leaving the state of the FSM unchanged), $\hat{S}^+$, $\hat{S}^-$ and $\hat{S}^z$, changing the state to 1, 2 and 3, respectively. Placing of any of the spin operators forces us to immediately insert the corresponding operator appearing in $\hat{H}$, to complete the interaction term. For example, putting of $\hat{S}^+$ imposes placement of $\frac{J}{2}\hat{S}^-$, which also changes the state of the FSM to F. Similarly, for the case of remaining intermediate states 2 and 3, the FSM can transition to F by inserting $\frac{J}{2}\hat{S}^+$ and $J^z\hat{S}^z$, accordingly. Finally, the FSM being in the F state has already placed all possible spin operators in the string, leaving the unit operator $\mathbb{1}$ as the only possible option to be inserted. However, as already mentioned, F is the final state, so similarly to the case of R, placement of $\mathbb{1}$ leaves FSM in F.

Summarizing, we can list all possible transitions in the FSM as follows: R - R by $\mathbb{1}$, R - 1 by $\hat{S}^+$, R - 2 by $\hat{S}^-$, R - 3 by $\hat{S}^z$, R - F by $-h\hat{S}^z$ (the single-site term), 1 - F by $\frac{J}{2}\hat{S}^-$, 2 - F by $\frac{J}{2}\hat{S}^+$, 3 - F by $J^z\hat{S}^z$ and finally F - F by $\mathbb{1}$. The corresponding FSM is depicted in Fig. 2.24.a.

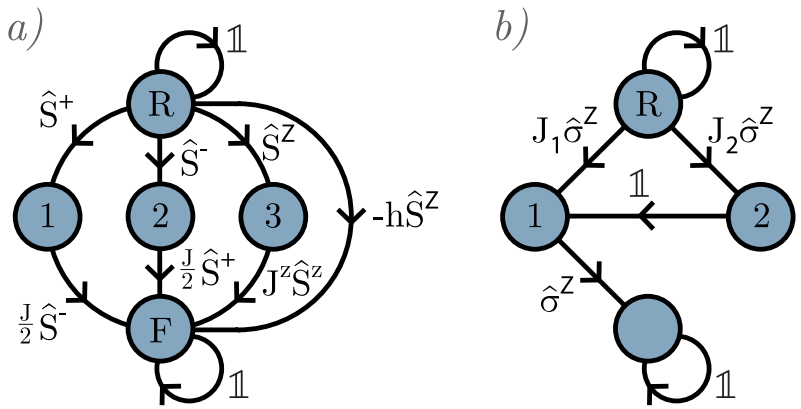


Figure 2.24: Finite state machines generating MPOs for (a) the nearest-neighbors transverse field and (b) the long-range J_1, J_2 Ising models.

However, the graphical illustration is not the only possible way to encode all possible changes of the FSM state. For that purpose we can use an operator-valued matrix, whose basis is spanned by the 5 states of the FSM. Each operator appearing in such a matrix corresponds to the transition between corresponding states, resulting in the following bulk-tensor of the MPO

$$W_{[i]} = \begin{array}{c} \\ R \\ 1 \\ 2 \\ 3 \\ F \end{array} \begin{array}{c} \begin{array}{ccccc} R & 1 & 2 & 3 & F \end{array} \\ \begin{pmatrix} \mathbb{1} & 0 & 0 & 0 & 0 \\ \hat{S}^+ & 0 & 0 & 0 & 0 \\ \hat{S}^- & 0 & 0 & 0 & 0 \\ \hat{S}^z & 0 & 0 & 0 & 0 \\ -h\hat{S}^z & (J/2)\hat{S}^- & (J/2)\hat{S}^+ & J^z\hat{S}^z & \mathbb{1} \end{pmatrix} \end{array}. \tag{2.46}$$

Each entry in above 5×5 matrix is also a 2×2 matrix, resulting in $W_{[i]}$ being a order-4 tensor, with two virtual bonds of size 5. Recalling the shapes of the tensors included in an

MPS we can notice that its first and last tensors have one virtual bond less than the bulk-tensors (assuming no periodic boundary conditions). We must take this fact into account while generating the edge tensors of the MPO, and adjust their sizes in a similar fashion. For the product $W_{[1]} \cdot W_{[2]} \cdots W_{[L]}$ to yield the Hamiltonian given in Eq. (2.44), we need to surround it with the boundary vectors $v_L = \begin{pmatrix} 0 & 0 & 0 & 0 & 1 \end{pmatrix}$ and $v_R = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \end{pmatrix}^T$, arriving at $v_L \cdot W_{[1]} \cdot W_{[2]} \cdots W_{[L]} \cdot v_R$. The v_L and v_R vectors can be incorporated in the left- and right-most tensors of the MPO, respectively, resulting in

$$W_{[1]} = \begin{pmatrix} -h\hat{S}^z & (J/2)\hat{S}^- & (J/2)\hat{S}^+ & J^z\hat{S}^z & \mathbb{1} \end{pmatrix}, \qquad W_{[L]} = \begin{pmatrix} \mathbb{1} \\ \hat{S}^+ \\ \hat{S}^- \\ \hat{S}^z \\ -h\hat{S}^z \end{pmatrix}. \tag{2.47}$$

As implied at the beginning of this section, MPOs can also incorporate long-range interactions. We will exemplify the construction of such an operator on the case of a J_1, J_2 Ising model

$$\hat{H} = J_1 \sum_{i=1}^{L-1} \hat{\sigma}_i^z \hat{\sigma}_{i+1}^z + J_2 \sum_{i=1}^{L-2} \hat{\sigma}_i^z \hat{\sigma}_{i+2}^z. \tag{2.48}$$

The FSM for this model is constructed analogous to the case of the Heisenberg model, with the difference that we introduce one possible transition between intermediate states via an unitary operator. Graphical illustration of such an FSM is given in Fig. 2.24.b. Again, by choosing the states of the FSM as basis for the matrix, we can encode all possible transitions in the following bulk-tensor

$$W'_{[i]} = \begin{array}{c} \\ R \\ 1 \\ 2 \\ F \end{array} \begin{array}{c} \begin{array}{cccc} R & 1 & 2 & F \end{array} \\ \begin{pmatrix} \mathbb{1} & 0 & 0 & 0 \\ J_1\hat{\sigma}^z & 0 & \mathbb{1} & 0 \\ J_2\hat{\sigma}^z & 0 & 0 & 0 \\ 0 & \hat{\sigma}^z & 0 & \mathbb{1} \end{pmatrix} \end{array}. \tag{2.49}$$

We can obtain the edge-tensors in a way similar to the one given in the example of a short-range Hamiltonian, arriving at

$$W'_{[1]} = \begin{pmatrix} 0 & \hat{\sigma}^z & 0 & \mathbb{1} \end{pmatrix}, \qquad W'_{[L]} = \begin{pmatrix} \mathbb{1} \\ J_1\hat{\sigma}^z \\ J_2\hat{\sigma}^z \\ 0 \end{pmatrix}. \tag{2.50}$$

Having defined the algorithm for expressing Hamiltonians in the form of MPOs, let us look at two possible, particularly interesting cases of their use. Firstly, the application of an MPO to an MPS can be performed as shown in Fig. 2.25.

We can see that after the contractions over physical legs are finished, the resulting tensor network has double horizontal bonds. These legs can be combined together, giving as a result an MPS, with virtual bonds of larger sizes than the initial state. This unwanted property will be remedied by methods introduced in the following section.

The second operation that can be carried out on an MPO, is the calculation of its expectation value. This is achieved by "sandwiching" an MPO between an MPS and its Hermitian adjoint, as illustrated in Fig. 2.26.

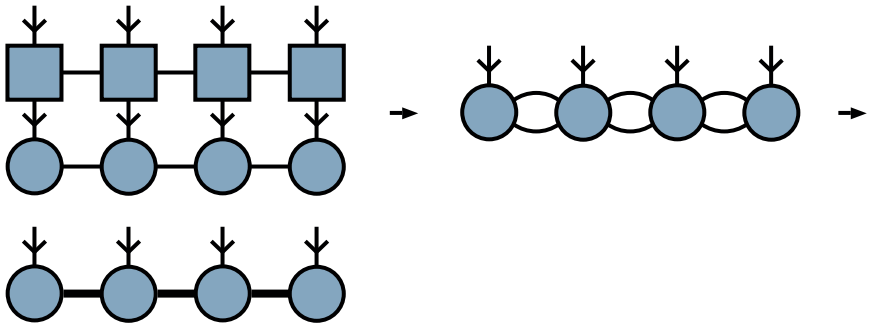


Figure 2.25: Application of an MPO to an MPS.

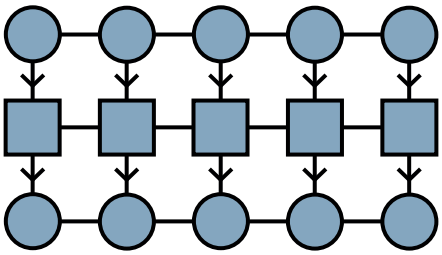


Figure 2.26: Calculation of expectation value of energy of an MPO.

It should be pointed out that contraction over all bonds depicted in Fig. 2.26 results in a scalar, being the desired expectation value, while no other operations such as shifting of the orthogonality center are needed. If the MPO expresses some particular Hamiltonian, this value would be the expectation value of energy of the model under study. We can see that this method is much simpler than the entire procedure required to obtain the expectation value of energy using bond operators, demonstrating the utility of MPOs.

2.5 Tensor network compression

While investigating the properties of a state produced by applying an MPO to an MPS we have seen that its horizontal bonds have larger sizes than the initial ones. Therefore, the size of the MPS and, consequently, its memory footprint would rapidly increase as a result of a number of MPO applications. To prevent this from happening we need a method of *compressing* given MPS. We will present two methods to achieve this goal: an *SVD-based* and a *variational* one.

2.5.1 Compression by SVD

Let us assume that we were given an MPS in the left-canonical form, with maximal bond-size χ', which we would like to compress. The most naive approach would be to conduct a procedure similar to the one allowing for cannonization of given MPS that is to conduct a sweep of SVDs with truncations throughout the whole chain, going from right to left [28]. However in this scenario, each SVD would factorize a tensor with just a single physical leg of size d. Let us once again write down the coefficients of state $|\psi\rangle$ stored as an MPS in the left-canonical form

$$\Psi^{(p_1,p_2,\ldots,p_L)} = A^{p_1}_{a_1} A^{a_1,p_2}_{a_2} A^{a_2,p_3}_{a_3} \ldots A^{a_{L-2},p_{L-1}}_{a_{L-1}} \Theta^{a_{L-1}}_{p_L}. \tag{2.51}$$

We can factorize the right-most Θ tensor by means of SVD as follows

$$\begin{aligned} A^{p_1}_{a_1} A^{a_1,p_2}_{a_2} A^{a_2,p_3}_{a_3} \dots A^{a_{L-2},p_{L-1}}_{a_{L-1}} \Theta^{a_{L-1}}_{p_L} &= A^{p_1}_{a_1} A^{a_1,p_2}_{a_2} A^{a_2,p_3}_{a_3} \dots A^{a_{L-2},p_{L-1}}_{a_{L-1}} U^{a_{L-1}}_{a_{L-1}} S^{a_{L-1}}_{a_{L-1}} B^{a_{L-1}}_{p_L} \\ &\approx A^{p_1}_{a_1} A^{a_1,p_2}_{a_2} A^{a_2,p_3}_{a_3} \dots A^{a_{L-2},p_{L-1}}_{a_{L-1}} \tilde{U}^{a_{L-1}}_{a_{L-1}} \tilde{S}^{a_{L-1}}_{a_{L-1}} \tilde{B}^{a_{L-1}}_{p_L}, \end{aligned} \tag{2.52}$$

where we immediately relabeled the $V^\dagger$ matrix as B tensor (because it already had a proper form). Second line in the above equation represents the truncation of singular values, which is indicated by the $\sim$ symbol over the tensors affected by this operation.

We can proceed to the next step of compression by contracting the $\tilde{U}$ and $\tilde{S}$ matrices with the A tensor assigned to the $(L-1)$th site, resulting in

$$A^{p_1}_{a_1} A^{a_1,p_2}_{a_2} A^{a_2,p_3}_{a_3} \dots A^{a_{L-2},p_{L-1}}_{a_{L-1}} \tilde{U}^{a_{L-1}}_{a_{L-1}} \tilde{S}^{a_{L-1}}_{a_{L-1}} \tilde{B}^{a_{L-1}}_{p_L} = A^{p_1}_{a_1} A^{a_1,p_2}_{a_2} A^{a_2,p_3}_{a_3} \dots \tilde{\Theta}^{a_{L-2},p_{L-1}}_{a_{L-1}} \tilde{B}^{a_{L-1}}_{p_L}. \tag{2.53}$$

By conducting these two steps of decomposition and tensor multiplication $L-1$ times (in total), we arrive at the right-normalized, compressed MPS

$$A^{p_1}_{a_1} A^{a_1,p_2}_{a_2} A^{a_2,p_3}_{a_3} \dots A^{a_{L-2},p_{L-1}}_{a_{L-1}} \Theta^{a_{L-1}}_{p_L} \approx \dots \approx \tilde{\Theta}^{p_1}_{a_1} \tilde{B}^{a_1,p_2}_{a_2} \tilde{B}^{a_2,p_3}_{a_3} \dots \tilde{B}^{a_{L-2},p_{L-1}}_{a_{L-1}} \tilde{B}^{a_{L-1}}_{p_L}. \tag{2.54}$$

The consecutive steps of this method are presented graphically in Fig. 2.27.

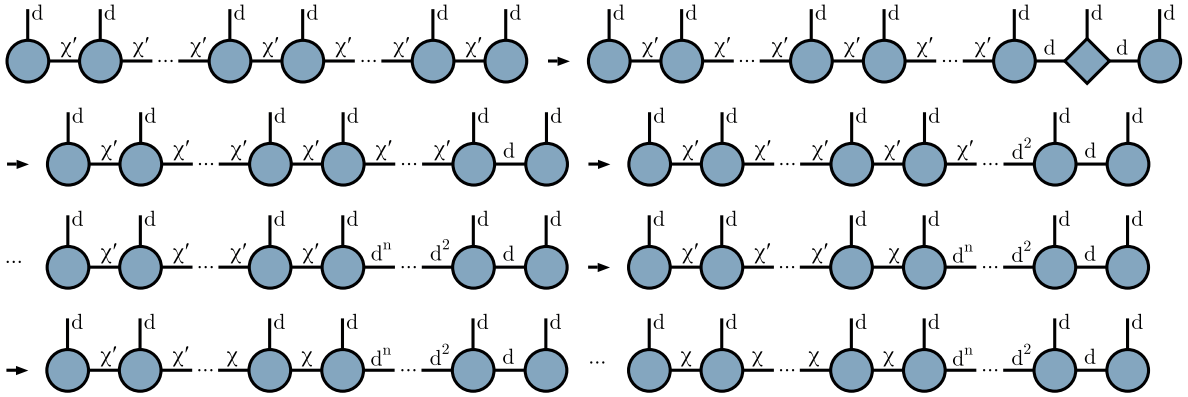


Figure 2.27: Diagrammatic representation of the SVD compression.

While getting rid of the singular values during SVDs, we can adopt two strategies. Firstly, we can stick to a predetermined value of χ as the bond-size for the resulting MPS, however this approach does not provide any bounds on the distance (measured as the 2-norm) between the state that we want to compress, and the final one. An alternative policy would be to pick an acceptable error ϵ, and during truncation omit the number of singular values, whose sum of squares is smaller than or equal to ϵ (as described in Eq. (2.35)). This approach would implicitly pick the bond-size χ, allowing for a more flexible construction of the final MPS. The risk associated with this method is that the singular values may be distributed uniformly, e.g., in a highly entangled state, in which case the number of omitted singular values would be minor, leading to a large bond-size χ. We can see then that there is a certain trade-off between the fidelity of compression and the number of variational parameters stored in the final MPS. However, this issue will only be relevant for highly correlated states, while in a wide range of cases we will be able to effectively compress a particular state while maintaining its properties.

We will now take a closer look at the computational cost of this kind of compression. It is known that this technique is very fast for $\chi' \sim \chi$, however becomes slow for $\chi' \gg \chi$ [28]. Let us therefore analyze the complexities of the most expensive SVD and tensor contraction.

Starting from the right edge of the chain, we group the right virtual leg of each subsequent Θ tensor with the physical leg, to create a matrix that can be decomposed by SVD. The resulting bonds connecting U, S and $V^\dagger$ matrices will be of size d (assuming no truncation is performed yet, and $\chi' \gg d$), due to the properties of SVD. After proper multiplication and reshaping, the next tensor to be decomposed will be of size $\chi' \times (d \cdot d)$. The factorization of this matrix will give in turn a bond of size d^2, which will become the right, virtual bond of the next Θ tensor. This situation will repeat up to the nth iteration, in which the right bond, being of size d^n, after the combination with physical leg will result in bond of size d^{n+1}, which will become larger than χ. This will be the moment of the first truncation, resulting in the U matrix of size $\chi' \times \chi$ and $V^\dagger$ matrix of size $\chi \times d^{n+1}$. After incorporation of singular values into U and its further multiplication with the tensor located on the left, we will obtain the largest Θ tensor to be factorized during the whole process, being of size $\chi' \times d \times \chi$. The right dimension of Θ will never become larger than χ, while under the initial assumptions we are guaranteed that the left bond will be of size χ' or smaller. Knowing the computational cost of SVD (see Section 2.2) we get the complexity of the largest factorization being equal to $\mathcal{O}(\chi' d\chi \cdot \chi') = \mathcal{O}((\chi')^2 d\chi)$ for $\chi' \leq d\chi$, and $\mathcal{O}(\chi' d\chi \cdot d\chi) = \mathcal{O}((\chi') d^2 \chi^2)$ otherwise. These limits however fail, when the initial MPS is not in the canonical form. In that case, the mere costs of SVDs bringing it to a properly normalized form are equal to $\mathcal{O}((\chi')^3 d)$ each [28].

Finally, the U tensor resulting from the decomposition (and further truncation) of the largest possible Θ will be of size $\chi' \times \chi$. Thus, its further multiplication with the next tensor with dimensions $\chi' \times d \times \chi'$ will cost $\mathcal{O}((\chi')^2 d\chi)$ (see Section 2.1.1).

We will end this section with a remark that an SVD-based compression is not an optimal one. Additionally, by its design, it can only operate on a single bond, which could make it difficult to escape local minima. However, this method can generate a good starting point for the next algorithm to be introduced - the variational compression.

2.5.2 Variational compression

Another approach to obtain a compressed state $|\tilde{\psi}\rangle$ from the initial one $|\psi\rangle$ is to minimize the distance between them, measured as 2-norm that is $\||\psi\rangle - |\tilde{\psi}\rangle\|_2^2 = (\langle\psi| - \langle\tilde{\psi}|)(|\psi\rangle - |\tilde{\psi}\rangle) = \langle\psi|\psi\rangle - \langle\psi|\tilde{\psi}\rangle - \langle\tilde{\psi}|\psi\rangle + \langle\tilde{\psi}|\tilde{\psi}\rangle$, with respect to $|\tilde{\psi}\rangle$. This can be achieved by choosing an initial guess for $|\tilde{\psi}\rangle$ (being random, or given by the SVD-compression), followed by further iterative method making use of the fact that all tensors appearing in the MPSs can be treated as variational parameters [28]. In each iteration we will be fixing all tensors besides a single one, which will be updated in a way minimizing the aforementioned distance.

Assuming no normalization of the MPSs (e.g., in the case of randomly chosen initial guess for $|\tilde{\psi}\rangle$), we can call tensors of $|\psi\rangle$ and $|\tilde{\psi}\rangle$ as T and $\tilde{T}$, respectively. Then, we can find an updated tensor $\tilde{T}^{p_i}$ (assigned to the ith site) by extremizing the distance with respect to $(\tilde{T}^\dagger)_{p_i'}$. The latter tensor appears only in the $\langle\tilde{\psi}|\tilde{\psi}\rangle - \langle\tilde{\psi}|\psi\rangle$ part for the expression for the norm, which gives us

$$\begin{aligned}\frac{\partial}{\partial(\tilde{T}^\dagger)_{p_i'}}(\langle\tilde{\psi}|\tilde{\psi}\rangle-\langle\tilde{\psi}|\psi\rangle)=&\Big((\tilde{T}^\dagger)^{a_{L-1}',p_L'}\dots(\tilde{T}^\dagger)^{a_i'}_{a_{i+1}',p_{i+1}'}(\tilde{T}^\dagger)^{a_{i-2}'}_{a_{i-1}',p_{i-1}'}\dots(\tilde{T}^\dagger)_{a_1',p_1'}\Big)\\&\times\Big(\tilde{T}^{p_1}_{a_1}\dots\tilde{T}^{a_{i-2},p_{i-1}}_{a_{i-1}}\tilde{T}^{a_{i-1},p_i}_{a_i}\tilde{T}^{a_i,p_{i+1}}_{a_{i+1}}\dots\tilde{T}^{a_{L-1}}_{p_L}\Big)\\&-\Big((\tilde{T}^\dagger)^{a_{L-1}',p_L'}\dots(\tilde{T}^\dagger)^{a_i'}_{a_{i+1}',p_{i+1}'}(\tilde{T}^\dagger)^{a_{i-2}'}_{a_{i-1}',p_{i-1}'}\dots(\tilde{T}^\dagger)_{a_1',p_1'}\Big)\\&\times\Big(T^{p_1}_{a_1}\dots T^{a_{i-2},p_{i-1}}_{a_{i-1}}T^{a_{i-1},p_i}_{a_i}T^{a_i,p_{i+1}}_{a_{i+1}}\dots T^{a_{L-1}}_{p_L}\Big)=0.\end{aligned}\tag{2.55}$$

As can be seen, the $(\tilde{T}^\dagger)_{p_i'}$ disappeared from the above equation, leaving some dangling legs, which (as will be shown later) will serve us as the bonds of the newly obtained tensor. By reorganizing Eq. (2.55) we obtain

$$\begin{aligned}&\Big((\tilde{T}^\dagger)^{a_{L-1}',p_L'}\dots(\tilde{T}^\dagger)^{a_i'}_{a_{i+1}',p_{i+1}'}(\tilde{T}^\dagger)^{a_{i-2}'}_{a_{i-1}',p_{i-1}'}\dots(\tilde{T}^\dagger)_{a_1',p_1'}\Big)\\&\times\Big(\tilde{T}^{p_1}_{a_1}\dots\tilde{T}^{a_{i-2},p_{i-1}}_{a_{i-1}}\tilde{T}^{a_{i-1},p_i}_{a_i}\tilde{T}^{a_i,p_{i+1}}_{a_{i+1}}\dots\tilde{T}^{a_{L-1}}_{p_L}\Big)\\&=\Big((\tilde{T}^\dagger)^{a_{L-1}',p_L'}\dots(\tilde{T}^\dagger)^{a_i'}_{a_{i+1}',p_{i+1}'}(\tilde{T}^\dagger)^{a_{i-2}'}_{a_{i-1}',p_{i-1}'}\dots(\tilde{T}^\dagger)_{a_1',p_1'}\Big)\\&\times\Big(T^{p_1}_{a_1}\dots T^{a_{i-2},p_{i-1}}_{a_{i-1}}T^{a_{i-1},p_i}_{a_i}T^{a_i,p_{i+1}}_{a_{i+1}}\dots T^{a_{L-1}}_{p_L}\Big).\end{aligned}\tag{2.56}$$

We can easily permute the order in which tensors appeared in Eq. (2.56) by using the Einstein notation, allowing us to organize the contractions as follows

$$\begin{aligned}&\Big((\tilde{T}^\dagger)^{a_{i-2}'}_{a_{i-1}',p_{i-1}'}\dots\Big((\tilde{T}^\dagger)_{a_1',p_1'}\tilde{T}^{p_1}_{a_1}\Big)\dots\tilde{T}^{a_{i-2},p_{i-1}}_{a_{i-1}}\Big)\\&\times\Big((\tilde{T}^\dagger)^{a_i'}_{a_{i+1}',p_{i+1}'}\dots\Big((\tilde{T}^\dagger)^{a_{L-1}',p_L'}\tilde{T}^{a_{L-1}}_{p_L}\Big)\dots\tilde{T}^{a_i,p_{i+1}}_{a_{i+1}}\Big)\tilde{T}^{a_{i-1},p_i}_{a_i}\\&=\Big((\tilde{T}^\dagger)^{a_{i-2}'}_{a_{i-1}',p_{i-1}'}\dots\Big((\tilde{T}^\dagger)_{a_1',p_1'}T^{p_1}_{a_1}\Big)\dots T^{a_{i-2},p_{i-1}}_{a_{i-1}}\Big)\\&\times\Big((\tilde{T}^\dagger)^{a_i'}_{a_{i+1}',p_{i+1}'}\dots\Big((\tilde{T}^\dagger)^{a_{L-1}',p_L'}T^{a_{L-1}}_{p_L}\Big)\dots T^{a_i,p_{i+1}}_{a_{i+1}}\Big)T^{a_{i-1},p_i}_{a_i}.\end{aligned}\tag{2.57}$$

Although the above equation appears to be very complicated at first glance, it becomes fairly simple to understand once the correct order of contractions is explained. Let us firstly focus on the $\Big((\tilde{T}^\dagger)^{a_{i-2}'}_{a_{i-1}',p_{i-1}'}\dots\Big((\tilde{T}^\dagger)_{a_1',p_1'}\tilde{T}^{p_1}_{a_1}\Big)\dots\tilde{T}^{a_{i-2},p_{i-1}}_{a_{i-1}}\Big)$ part. By conducting contractions starting from the inner-most brackets we can notice that we are summing over the bonds of the tensors assigned to the first site, followed by contraction of tensors on the second site (including the result of the first operation), and so forth, up to the $(i-1)$th site. This whole procedure can be thought of as computing the *left environment* of the ith site.

A similar reasoning can be conducted for the $\Big((\tilde{T}^\dagger)^{a_i'}_{a_{i+1}',p_{i+1}'}\dots\Big((\tilde{T}^\dagger)^{a_{L-1}',p_L'}\tilde{T}^{a_{L-1}}_{p_L}\Big)\dots\tilde{T}^{a_i,p_{i+1}}_{a_{i+1}}\Big)$ part appearing in Eq. (2.57). In that case, we are just starting contractions from the Lth site and proceed leftwards, up to the $(i+1)$th site, obtaining the *right environment* of the ith site.

As a result, we can relabel the left and right environments of the ith site as $\tilde{L}$ and $\tilde{R}$, respectively, and arrive at

$$\Big(\tilde{L}_{a_{i-1}',a_{i-1}}\cdot\tilde{R}^{a_i',a_i}\Big)\tilde{T}^{a_{i-1},p_i}_{a_i}=L_{a_{i-1}',a_{i-1}}T^{a_{i-1},p_i}_{a_i}R^{a_i',a_i}.\tag{2.58}$$

Above we separated the $\tilde{L}$ and $\tilde{R}$ labels from the L and R ones to emphasize the fact that in the former case we are contracting the MPSs representing the compressed state $|\tilde{\psi}\rangle$ and its Hermitian adjoint $\langle\tilde{\psi}|$, while in the latter we are operating on $|\psi\rangle$ and $\langle\tilde{\psi}|$.

We can treat the two disjoint parts $\tilde{L}$ and $\tilde{R}$ of the tensor network on the left hand-side of Eq. (2.58) as a single operator $\tilde{E}^{a_i',a_i}_{a_{i-1}',a_{i-1}}$. Moreover, contraction of all of the tensors on the right-hand side gives a single tensor $(T')^{a_i',p_i}_{a_{i-1}'}$. If we combine these two facts, we can rewrite Eq. (2.58) as

$$\tilde{E}^{a_i',a_i}_{a_{i-1}',a_{i-1}} \tilde{T}^{a_{i-1},p_i}_{a_i} = (T')^{a_i',p_i}_{a_{i-1}'}, \tag{2.59}$$

whose constituents can be further reshaped, resulting in

$$\tilde{E}^{(a_{i-1}',a_i')}_{(a_{i-1},a_i)} \tilde{T}^{(a_{i-1},a_i)}_{p_i} = (T')^{(a_{i-1}',a_i')}_{p_i}. \tag{2.60}$$

The last operation performed allows us to see the $\tilde{E}^{(a_{i-1}',a_i')}_{(a_{i-1},a_i)} \tilde{T}^{(a_{i-1},a_i)}_{p_i}$ contraction as a simple matrix-matrix multiplication. If we also take into account the fact that the p_i index iterates over the columns of the $\tilde{T}^{(a_{i-1},a_i)}_{p_i}$ and $(T')^{(a_{i-1}',a_i')}_{p_i}$ matrices, we may *slice* the data by taking corresponding column vectors from each of the two matrices, for each value of p_i. As a result, we can write down a system of linear equations for each p_i as follows

$$Ax = b, \tag{2.61}$$

which can be further solved via an iterative solver, like the conjugate gradient or trust regions methods.

However, the computational cost of steps involved in obtaining the $\tilde{E}^{a_i',a_i}_{a_{i-1}',a_{i-1}}$ operator and subsequent solving of the system of linear equations can be immensely reduced. At the beginning of this section we assumed no normalization was imposed on the $|\psi\rangle$ and $|\tilde{\psi}\rangle$ states. Although, if the MPSs in question were in the mixed canonical form, with the orthogonality center of the compressed state being at the site that was updated, we could notice that $\tilde{L}_{a_{i-1}',a_{i-1}} = \delta_{a_{i-1}',a_{i-1}}$ and $\tilde{R}^{a_i',a_i} = \delta^{a_i',a_i}$. As a result, we could rewrite Eq. (2.59) as

$$\tilde{T}^{a_{i-1},p_i}_{a_i} = (T')^{a_i',p_i}_{a_{i-1}'}. \tag{2.62}$$

In this way, we are able to omit the contractions resulting in $\tilde{L}_{a_{i-1}',a_{i-1}}$ and $\tilde{R}^{a_i',a_i}$ environments and the further use of an iterative solver. Thus, the updated tensor on given site can be obtained only by the contraction of $|\psi\rangle$ with $\langle\tilde{\psi}|$, from which a single tensor was removed. Fig. 2.28 shows diagrammatic representations of the more general compressing algorithm, as well as the canonical form-based one.

There are a number of steps that might be taken in order to improve the speed of compression. Firstly, after optimization of given tensor of $|\tilde{\psi}\rangle$ is finished, we need to shift the orthogonality center, so that the update on the following site can be made. This can be done, as usual, by means of the SVD. For the sake of simplicity, let us assume that we have optimized the ith site and need to proceed rightwards (a similar reasoning is of course valid for the leftward move). We can notice that the tensor being updated at given moment (or more specifically, its Hermitian adjoint) disappears from the right hand-side of Eq. (2.58). This means that while performing the factorization we can keep only the U matrix, while S and $V^\dagger$ may be omitted, because the tensor resulting from their contraction will be modified during

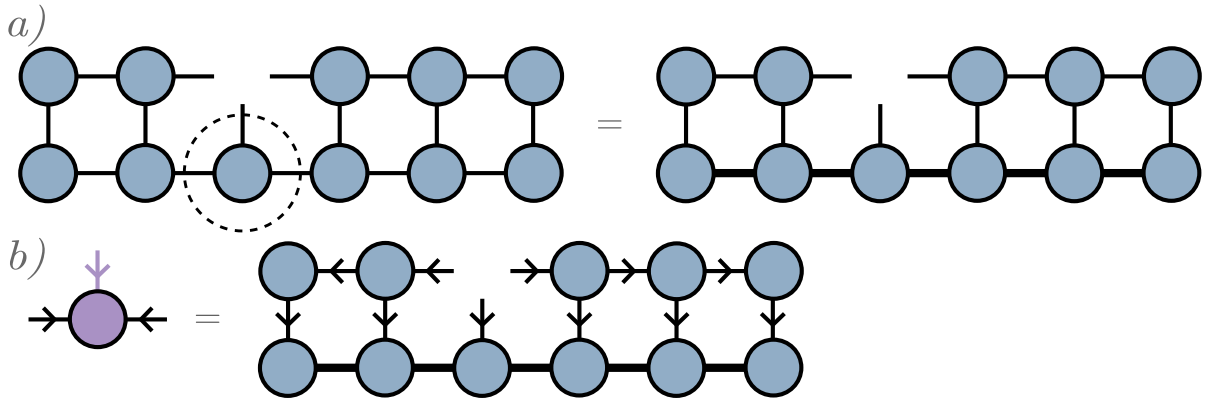


Figure 2.28: Variational compression algorithm (a) in the most general form, and (b) utilising the canonical form of the MPS.

the update anyway. In the case of the leftward move the $V^\dagger$ matrix can be preserved, while S and U tensors can be ignored. Moreover, since each of the $|\psi\rangle$ state tensors will be contracted with a corresponding tensor from the $\langle\tilde{\psi}|$, it is not necessary to keep $|\psi\rangle$ in canonical form. The consequence of this is the observation that no further SVDs need to be conducted on $|\psi\rangle$, which remains constant throughout each iteration of the compression algorithm.

With the variational compression it is also possible to update two sites simultaneously. This modification reduces the chances of getting caught in a local minimum and can greatly reduce the distance between the initial state, and the compressed one. Graphical illustration of this version of the algorithm is shown in Fig. 2.29.

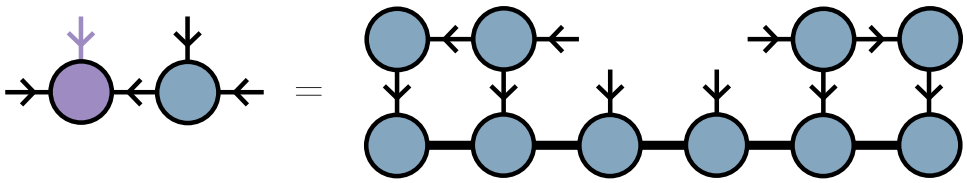


Figure 2.29: Two-site version of the variational compression algorithm.

Finally, in order to reduce the number of contractions, the left and right environments of given sites are usually stored in an array. Then, when a given tensor of $|\tilde{\psi}\rangle$ is updated, its Hermitian adjoint is contracted with its left (right) environment, along with the corresponding tensor from $|\psi\rangle$, resulting in the new environment of the site located to the right (left). This technique is illustrated in Fig. 2.30.

2.6 Time Evolving Block Decimation

So far, we have mostly concentrated on the examination of quantum states that remained constant over time. The only exceptions to this rule were situations, in which these states were modified by application of an operator, which was represented as a bond operator or an MPO. In this section we will build upon these methods, and demonstrate the *Time Evolving Block Decimation* (TEBD) algorithm [25, 26], which enables the time evolution of slightly entangled physical states. Next, we will show how a small modification to this method can be used to determine the ground state of a chosen physical model.

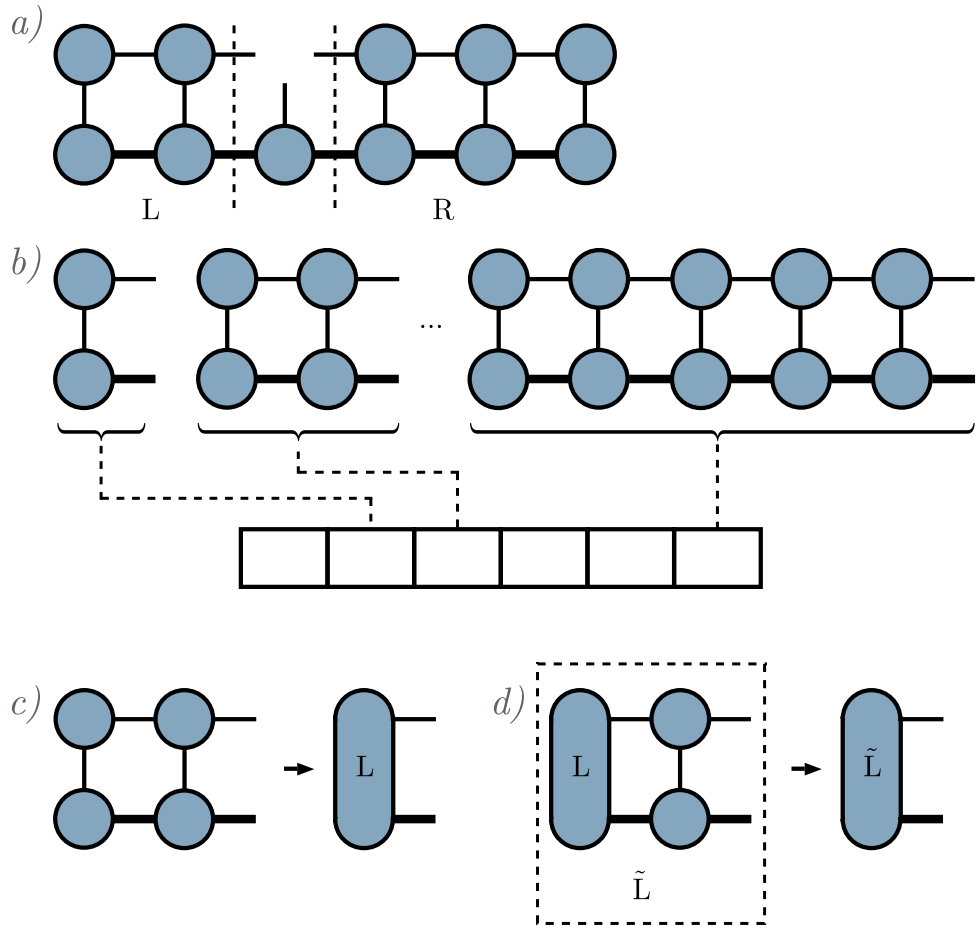


Figure 2.30: (a) Division of the environment of a given site into the left and right parts. (b) Storing of the left part of the environment in an array with emphasis on which tensors it consists of for a given cell. (c) Contraction resulting in a single tensor representing the left part of the environment. (d) Update of the environment.

2.6.1 Real time evolution

The TEBD algorithm is based on the *Suzuki-Trotter decomposition* [56–58], which allows for approximation of matrix exponentiation. The first order decomposition is given by

$$e^{\lambda(A+B)} = \lim_{n\to\infty}(e^{A/n}e^{B/n})^n \tag{2.63}$$

or, analogously

$$e^{\lambda(A+B)} = e^{\lambda A}e^{\lambda B} + \mathcal{O}(\lambda^2), \tag{2.64}$$

where $\lambda \ll 1$ is a parameter, and A and B are matrices, which in general do not commute generating the error represented by the last term in the above equation. Of course we can extend this formula to the case, in which we are dealing with a much larger number of summed elements. For example, we can use this approximation to transform the evolution operator given by a Hamiltonian with nearest-neighbours interactions. This Hamiltonian can be written as $\hat{H} = \sum_{i=1}^{L-1}\hat{h}_{i,i+1}$, where $\hat{h}_{i,i+1}$ are the interaction terms between the neighbouring nodes, while the corresponding evolution operator as

$$\hat{U}(t) = e^{-\frac{i}{\hbar}\hat{H}t} = e^{-\frac{it}{\hbar}\sum_{i=1}^{L-1}\hat{h}_{i,i+1}}. \tag{2.65}$$

Following Eq. (2.64) we can divide the total evolution time t into N smaller time steps τ. Then, the evolution operator over the time τ can be written as

$$\hat{U}(\tau) = e^{-\frac{i}{\hbar}\hat{H}\tau} = \prod_{i=1}^{L-1} e^{-i\hat{h}_{i,i+1}\tau} + \mathcal{O}(\tau^2), \tag{2.66}$$

while the whole evolution would consist of N applications of operator $\hat{U}(\tau)$. We can see that in order to increase the final precision of the simulation we can decrease the time step τ, while simultaneously increasing their total number N.

To carry out such an evolution of a state stored as an MPS in the right-canonical form, we could apply a sequence of operators $e^{-\frac{i}{\hbar}\hat{h}_{i,i+1}\tau}$ on each pair of neighbouring sites, starting on the left edge of the chain and proceeding rightwards. Following the steps described in Section 2.4.2, before application of each operator we would merge two neighbouring sites, contract the resulting 2-site orthogonality center with $e^{-\frac{i}{\hbar}\hat{h}_{i,i+1}\tau}$, and finally split the subsequent tensor to restore the initial structure of the MPS, with one tensor per node. To keep the canonical form intact, during the splitting via SVD we could incorporate the singular values into the $V^\dagger$ matrix, so that no additional operation is needed before the application of the following operator. All these steps are illustrated in Fig. 2.31.

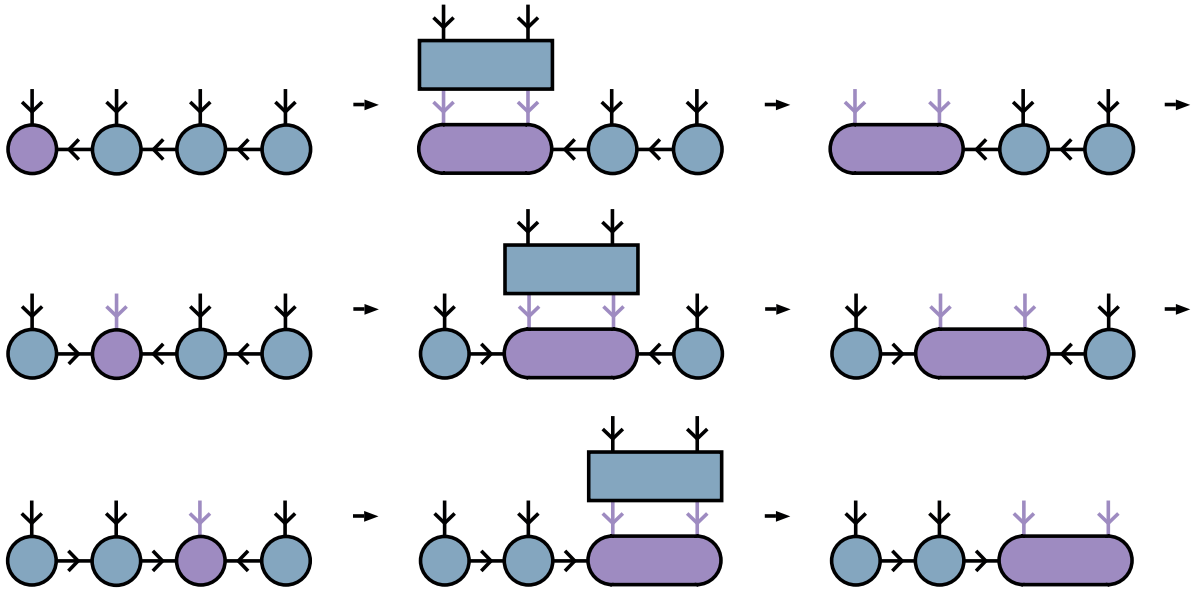


Figure 2.31: Basic version of the TEBD algorithm.

However, the error resulting from the noncommutativity of bond Hamiltonians can be diminished with the use of second-order Trotter-Suzuki decomposition, utilizing the forward and backward multiplication of constituent operators, giving us

$$\hat{U}(\tau) = \prod_{i=1}^{L-1} e^{-\frac{i}{\hbar}\hat{h}_{i,i+1}(\tau/2)} \prod_{j=L-1}^{1} e^{-\frac{i}{\hbar}\hat{h}_{j,j+1}(\tau/2)} + \mathcal{O}(\tau^3). \tag{2.67}$$

This method is illustrated in Fig. 2.32.a. At first sight it would seem that the second-order decomposition would require two times more applications of operators in comparison to the first order version. However, using specific characteristics of the Hamiltonian in question and aspects of the tensor networks discussed in this work, we can get around this problem.

Firstly, we can notice that the bond operators on even bonds commute with each other, thanks to which they can be applied in parallel. The same is true for the odd-bond operators.

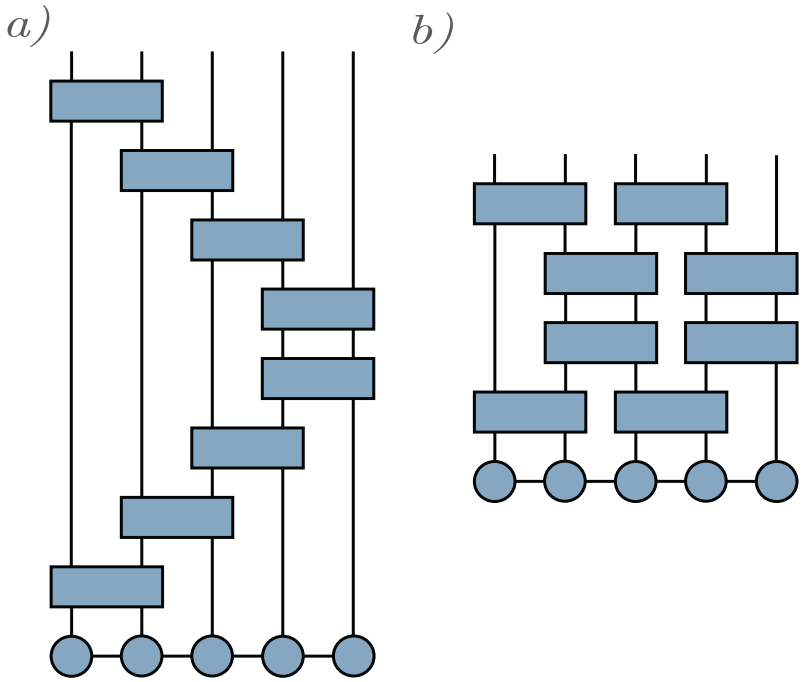


Figure 2.32: (a) Naive application of the operators resulting form a second-order Trotter-Suzuki decomposition. (b) Parallelization of application of operators shown in the sub-figure (a).

As a result we can reorganize the operators shown in Fig. 2.32.a, evolving the MPS over a single time step, into just 4 *layers*, which are shown in Fig. 2.32.b. Vidal representation is the best fit for this arrangement of operators, because thanks to it we can avoid unnecessary SVDs that would have to be conducted to shift the orthogonality center over the bonds, on which no operator is applied in given layer. Also, by separating the even interactions from the odd ones in the definition of the Hamiltonian $\hat{H} = \hat{H}_{even} + \hat{H}_{odd}$, we can denote the corresponding layers of operators as $e^{-\frac{i}{\hbar}\hat{H}_{even}(\tau/2)}$ and $e^{-\frac{i}{\hbar}\hat{H}_{odd}(\tau/2)}$.

Secondly, the odd-bond operators appearing in the second and third layers in Fig. 2.32.b can be grouped together, if no properties of the MPS are measured in-between the layers [28]. As a result we can reformulate the evolution operator over a single time step as

$$\hat{U}(\tau) = e^{-\frac{i}{\hbar}\hat{H}_{even}(\tau/2)} e^{-\frac{i}{\hbar}\hat{H}_{odd}\tau} e^{-\frac{i}{\hbar}\hat{H}_{even}(\tau/2)} + \mathcal{O}(\tau^3). \tag{2.68}$$

Finally, we can notice that we can also combine the layers of even-bond operators of the subsequent layers, again, if no properties of the MPS are measured between them. Thanks to this observation, we can write down the evolution operator over the time t as

$$\begin{aligned}\hat{U}(t) &= e^{-\frac{i}{\hbar}\hat{H}_{even}(\tau/2)} e^{-\frac{i}{\hbar}\hat{H}_{odd}\tau} e^{-\frac{i}{\hbar}\hat{H}_{even}(\tau/2)} e^{-\frac{i}{\hbar}\hat{H}_{even}(\tau/2)} \dots \\ &\dots e^{-\frac{i}{\hbar}\hat{H}_{even}(\tau/2)} e^{-\frac{i}{\hbar}\hat{H}_{even}(\tau/2)} e^{-\frac{i}{\hbar}\hat{H}_{odd}\tau} e^{-\frac{i}{\hbar}\hat{H}_{even}(\tau/2)} + \mathcal{O}(N\tau^3) \\ &= e^{-\frac{i}{\hbar}\hat{H}_{even}(\tau/2)} e^{-\frac{i}{\hbar}\hat{H}_{odd}\tau} e^{-\frac{i}{\hbar}\hat{H}_{even}\tau} \dots e^{-\frac{i}{\hbar}\hat{H}_{even}\tau} e^{-\frac{i}{\hbar}\hat{H}_{odd}\tau} e^{-\frac{i}{\hbar}\hat{H}_{even}(\tau/2)} + \mathcal{O}(N\tau^3). \end{aligned} \tag{2.69}$$

Thus, the second-order decomposition of operator $\hat{U}(t)$ requires the application of just one more layer compared to the first-order decomposition (using the even-odd grouping of operators), which is a completely acceptable price given the increase in precision we

obtain. All the steps involved in reordering of the operators resulting in Eq. (2.69) are shown graphically in Fig. 2.33.

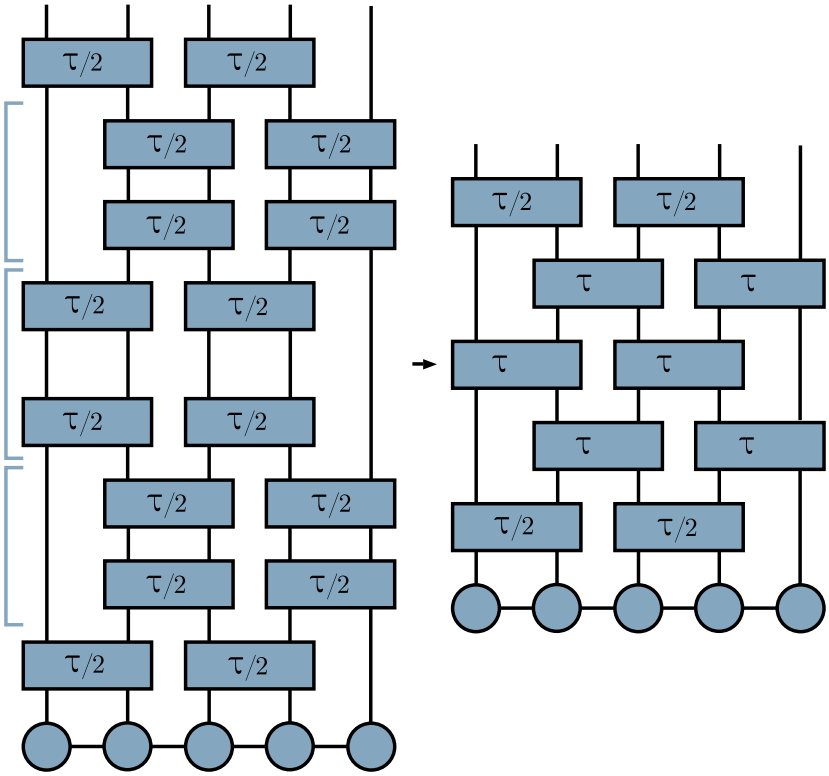


Figure 2.33: Merging of layers occurring in the second order Trotter-Suzuki decomposition.

2.6.2 Imaginary time evolution

By introducing a small change into the above method we can use it not only to evolve a given quantum state stored as an MPS, but also obtain the ground state of chosen Hamiltonian $\hat{H}$. Let us assume that $\hat{H}$ is time independent. In such a case, we can get the energy basis by solving the eigenvalue problem

$$\hat{H}|\psi_n\rangle = E_n|\psi_n\rangle, \tag{2.70}$$

where ψ_n are the eigenvectors, while E_n are the the energy eigenvalues. Then, we can choose a random initial state $|\psi_{random}\rangle$, which can be expressed in the above energy basis, at time t_0, as follows

$$|\psi_{random}(t_0)\rangle = \sum_n c_n|\psi_n\rangle, \tag{2.71}$$

where c_n are the expansion coefficients. Now, we can evolve this state up to the moment t, arriving at

$$|\psi_{random}(t)\rangle = \sum_n c_n e^{-iE_n(t-t_0)/\hbar}|\psi_n\rangle. \tag{2.72}$$

From Eq. (2.72) we can see that the contributions to the superposed state, coming from each of the basis eigenvectors, oscillate throughout the whole evolution with the frequency

proportional to $E_n/\hbar$. However, if we replace the real time t with its imaginary substitute $\tau = it$, and assume $\tau_0 = 0$, we may conduct an *imaginary time evolution* leading to the following state

$$|\psi_{random}(\tau)\rangle = \sum_n c_n e^{-E_n\tau/\hbar}|\psi_n\rangle. \tag{2.73}$$

This time, the expansion coefficients of the energy eigenstates do not oscillate, but rather decay exponentially fast with the rate proportional to $E_n/\hbar$. Because the ground state energy is, by definition, the smallest one among all E_n, its decay rate will be the slowest. Then, for large value of τ the ground state will be projected out of the initial random state, provided that it was part of the $|\psi_{random}(0)\rangle$, which is usually the case.

In order to implement the imaginary time evolution through the TEBD algorithm, we choose a set of time steps of decreasing length $\{\tau_1, \tau_2, \ldots, \tau_m\}$. Starting with τ_1 we generate the evolution operators $e^{-\frac{\tau_1}{\hbar}\hat{h}_{i,i+1}}$ (instead of $e^{-\frac{i\tau_1}{\hbar}\hat{h}_{i,i+1}}$). For the fixed value of τ_1 we run consecutive iterations of TEBD, until the change in energy of the state is smaller than the chosen threshold. The number of iterations cannot be guessed in advance, and is dependent solely on the convergence of energy. After reaching the point of minimal fluctuations, we decrease the time step length to τ_2, and proceed in a similar fashion. After conducting this procedure for all time steps, from τ_1 up to τ_m, the obtained state should be the projected ground state.

The final point worth highlighting in the context of imaginary time evolution via TEBD is that the second-order Suzuki-Trotter decomposition is typically used, albeit in its simplest version, as illustrated visually in Fig. 2.32.a.

2.7 Density Matrix Renormalization Group

Although effective, the imaginary time evoulution described in the preceding section has certain drawbacks. Firstly, it uses the bond operators, which makes it applicable only to systems with nearest-neighbours interactions. Secondly, it tends to get caught in local minima and not provide the true ground state energy. In this section we demonstrate a variational algorithm called *Density Matrix Renormalization Group* (DMRG) [28, 35–38], which allows for inclusion of long-range interactions and converges to the low energy values much faster than TEBD.

2.7.1 Finite version

The goal of this method is to find the state $|\psi\rangle$, given as an MPS that minimizes the energy of some Hamiltonian $\hat{H}$, represented in the form of an MPO. This state can be found by extremizing the expectation value of energy, which can be written as follows

$$\frac{d}{d|\psi\rangle}\frac{\langle\psi|\hat{H}|\psi\rangle}{\langle\psi|\psi\rangle} = 0. \tag{2.74}$$

We can get rid of the denominator in the fraction by using the Lagrangian multiplier technique, arriving at the folowing formula

$$\frac{d}{d|\psi\rangle}(\langle\psi|\hat{H}|\psi\rangle - \lambda\langle\psi|\psi\rangle) = 0. \tag{2.75}$$

The tensors, being the constituents of the $|\psi\rangle$ state, appear in Eq. (2.75) in a product form, making it a highly nonlinear problem. We can bypass this issue by following an approach similar to the one presented in Section 2.5.2 that is by optimising just a single tensor at a time, while fixing all the remaining ones. In this way, we can update consecutive tensors by sweeping through the lattice back and forth. Thus, in each iteration of the algorithm instead of finding the solution to Eq. (2.75), we will rather extremize

$$\frac{\partial}{\partial (T^{\dagger})_{p_i'}} (\langle\psi|\hat{H}|\psi\rangle - \lambda\langle\psi|\psi\rangle) = 0. \tag{2.76}$$

Assuming no normalization of the MPS, we can illustrate Eq. (2.76) using diagram notation as shown in Fig. 2.34.

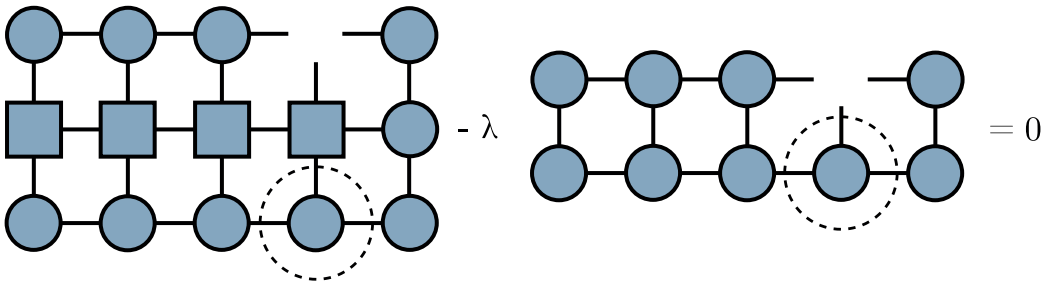


Figure 2.34: Diagrammatic representation of a single site update in the DMRG algorithm, when the MPS does not satisfy the canonical form.

Let us transform the tensor networks depicted in Fig. 2.34 in order to find an elegant way of finding the solution to Eq. (2.76). Again, as shown in Section 2.5.2, we can reshape the tensor T^{p_i} into a vector v of size $\chi^2 p$, while the remaining part of the left-most tensor network appearing in Fig. 2.34 can be contracted and reshaped into an $\chi^2 p \times \chi^2 p$ matrix M. This matrix is frequently referred to as an *effective Hamiltonian*. Because we have already decided to reshape T^{p_i} into v, we also need to find a proper transformation for the residual tensors included in the tensor network representing the calculation of the overlap. In fact, we can treat them as another matrix N. All of these conversions (shown graphically in Fig. 2.35) allow us to restate Eq. (2.76) as a *generalized eigenvalue problem*

$$Mv - \lambda N v = 0. \tag{2.77}$$

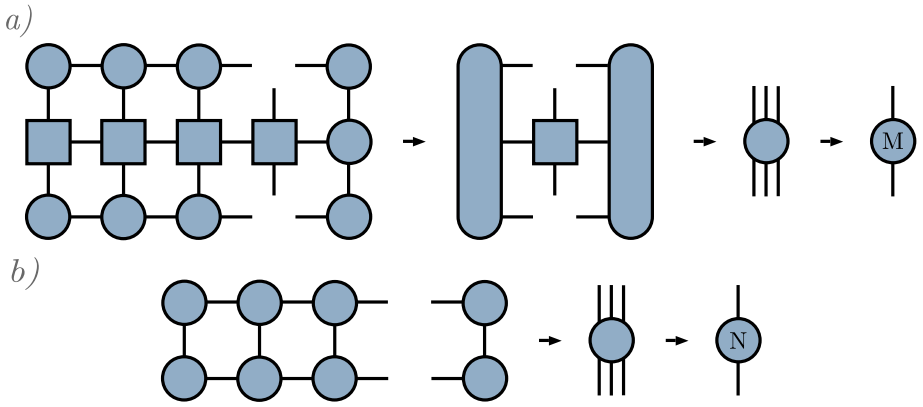


Figure 2.35: Generation of the (a) M and (b) N matrices appearing in Eq. (2.77).

Since it may be challenging to solve Eq. (2.77) numerically, we shall attempt to further simplify it. As usual, we can achieve immense gains by making use of the canonical form of the MPS. By keeping the orthogonality center at the ith site we can notice that the N matrix simplifies to the identity, which allows us to rewrite Eq. (2.77) as

$$Mv - \lambda v = 0, \tag{2.78}$$

which is a standard eigenvalue problem. The corresponding tensor networks are depicted in Fig. 2.36.

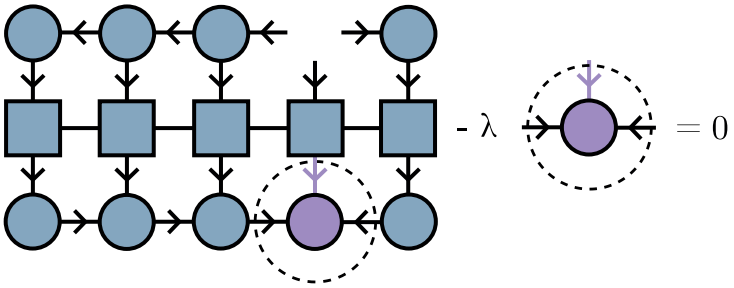


Figure 2.36: Update of a single site in the DMRG utilising the canonical from of the MPS.

Due to the size of the matrix M, usage of the exact diagonalization technique in order to find solutions of Eq. (2.78) might be too computationally expensive. Instead of that, it is recommended to use Lanczos algorithm to obtain the eigenstate with the smallest eigenvalue, corresponding to the energy, which is in fact the ultimate goal of the DMRG method. After reshaping the resulting vector v into a tensor T'^{p_i}, we might proceed to update the next tensor, by shifting the orthogonality center left or right. The whole procedure given above is performed on each site of the chain in sweeps going from left to right and backwards, until the energy converges.

A number of remarks regarding the implementation ought to be taken in this place. Firstly, this algorithm might be extended to find not only the ground state of given system, but also other low lying states. We can achieve this by initially running the standard DMRG and saving the resulting state $|\psi_0\rangle$. Then, we begin the search for the first excited state. It can be done by enforcing the orthogonality of the state resulting from the Lanczos technique, while solving the eigenvalue problem on a given site, to the analogous vector taken from $|\psi_0\rangle$. This is a commonly used extension of the Lanczos method.

The second observation concerns efficient calculation of the effective Hamiltonian M. It would be expensive to contract all of the component tensors in order to obtain M at each site. For this reason, usually each site has assigned its left and right environment, consisting of the appropriate parts of the MPO, sandwiched between the bra and ket MPSs. For example, the left environment of the ith site would include all tensors on sites from 1 to $i-1$ of $\hat{H}$, $|\psi\rangle$ and $\langle\psi|$, resulting in a tensor with three dangling legs. By analogy, the right environment would incorporate tensors on sites from $i+1$ up to L, also generating a tensor with three legs. Then in order to calculate M at given site it is sufficient to contract the left environment with the ith tensor of the MPO, followed by another contraction with the right environment. Construction and method of storage of these environments is shown in Fig. 2.37.a-c.

Of course, left and right environments should be updated between optimizations of consecutive sites. Let us assume that at some point of the ground state search we are proceeding rightwards. Then, after the optimization at the ith site and the following shifting of the orthogonality center is finished, we can contract the $A_{[i]}$ tensor with the left environment,

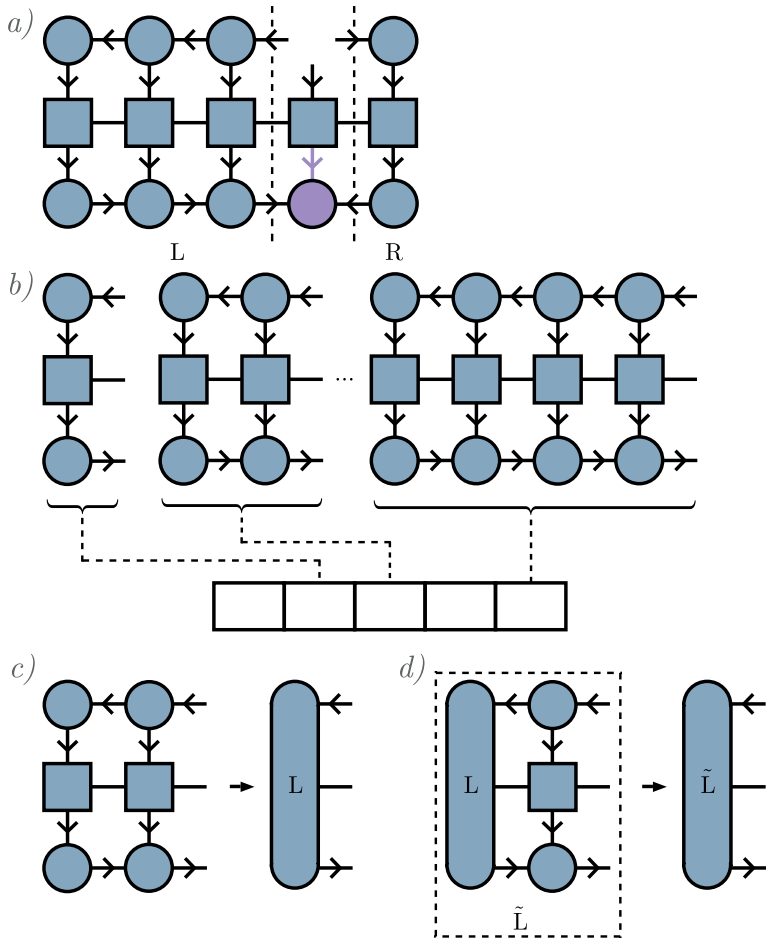


Figure 2.37: (a) Indication of the left and right parts of the effective Hamiltonian used in the DMRG algorithm. (b) Storing of the left environment of each site, with explicitly shown tensors being its constituents. (c) Contraction leading to a single tensor representing the left environment. (d) Update of the environment.

along with $W_{[i]}$ tensor of the MPO and $A^{\dagger}_{[i]}$. As a result, we obtain the left environment of the $(i+1)$th site. In this scenario, we do not have to update the right environment, because it would have changed anyway after the optimization of the $(i+)$th site is finished. Analogously, if we were sweeping leftwards, we would contract the $B_{[i]}$, $B^{\dagger}_{[i]}$ and $W_{[i]}$ tensors with the right environment and not update the left one. The update of the environment is shown in Fig. 2.37.d.

The final modification of the algorithm worth discussing is the two-site update. It should be noted that the single-site version has no possibility of increasing the sizes of virtual bonds. To address this issue the so called *mixers* are used, which, however, will not be discussed in detail in this work. Let us just mention that they allow for an increase in the bond-sizes of the horizontal legs in the MPS, and make it possible to escape local energy minima. The two-site update does not suffer from this problem, and in general manifests better convergence than the single-site version. The graphical depiction of this modification of the DMRG is shown in Fig. 2.38.a.

In this context, the shifting of the orthogonality center also becomes more intuitive, as it is simply performed during the splitting of the two-site tensor. The update of environments is also akin to the previously presented one. The left environment is updated by contraction with $A_{[i]}$, $W_{[i]}$ and $A^{\dagger}_{[i]}$ (done only when moving rightwards), while the right environment by

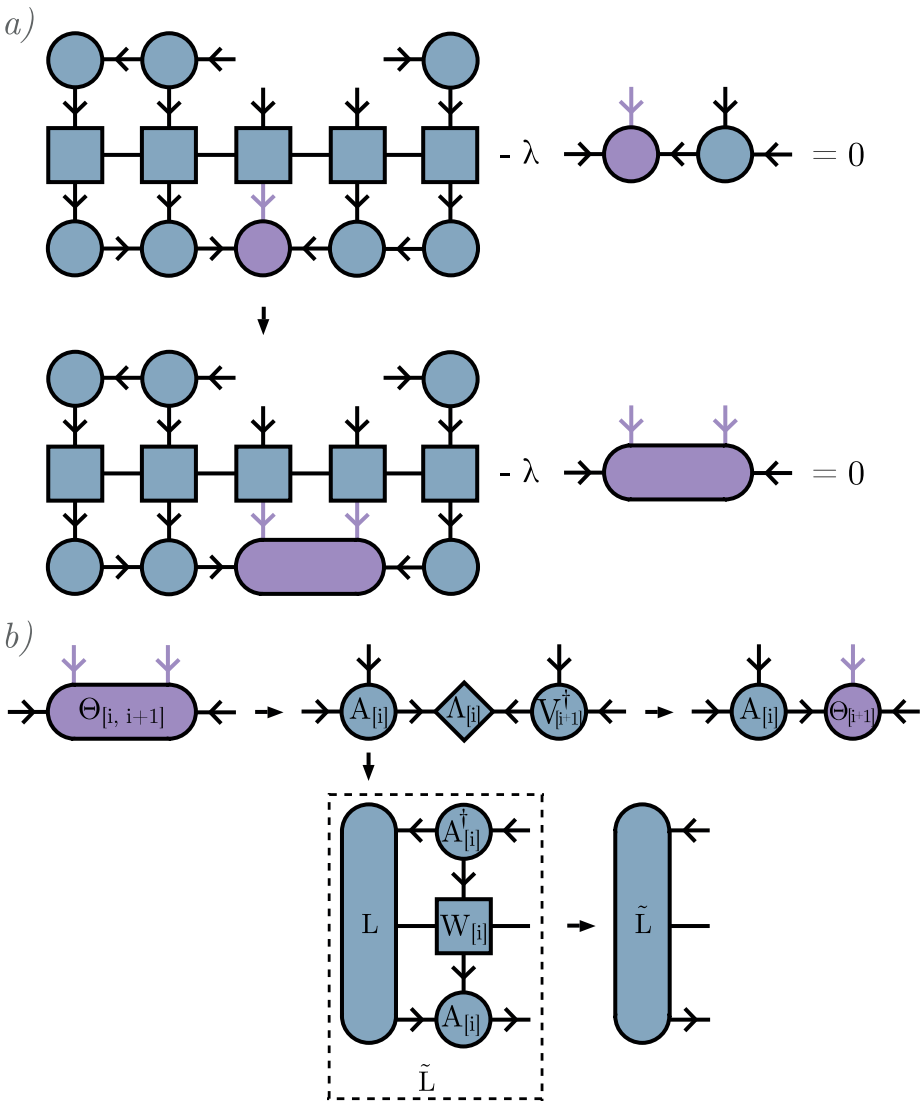


Figure 2.38: (a) The two-site version of the DMRG algorithm. (b) Shifting of the orthogonality center followed by the environment update, performed after the optimization step of the DMRG is finished.

contraction with $B_{[i+1]}$, $W_{[i+1]}$ and $B^{\dagger}_{[i+1]}$ (conducted only while moving leftwards), where $A_{[i]}$ and $B_{[i+1]}$ are obtained from the SVD of the two-site orthogonality center. Shifting along with the update of the environments is shown in Fig. 2.38.b.

2.7.2 Infinite version

Although extremely useful in the analysis of finite systems, the DMRG method making use of the tensor network formalism showed in the previous section does not closely resemble the original algorithm suggested by White [35,36], which allows to obtain the ground state of a given model in the thermodynamic limit. In this section, we will describe the infinite DMRG (iDMRG), which will be an intuitive extension of the finite version, enabling simulation of chains with $L \to \infty$. We will begin the explanation with a conceptual approach, which will be further expanded on when we concentrate on implementation details.

Let us assume that the system under study has translationally invariant unit cell, consist-

ing of L sites. At this point we can immediately emphasize that the unit cell of the algorithm might be different from the unit cell of the MPS. The infinite algorithm begins in the same way as the finite version. After the initialization of the system and environments (shown in Fig. 2.39.a) we sweep through the chain in a standard way. However, between the sweeps we increase the system size by inserting a copy of the "central" part of the chain into each of the environments (illustrated in Fig. 2.39.b). We then extend the range of sweeps, to incorporate all sites in the new, larger system.

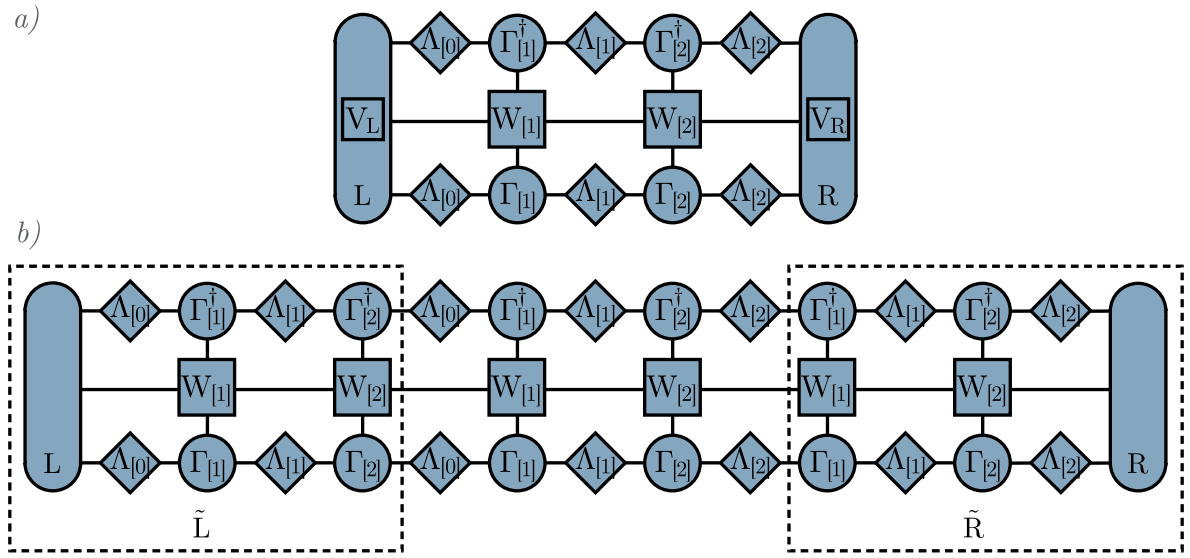


Figure 2.39: (a) Initialization of the infinite version of the DMRG method. (b) Conceptual view of the extension of the system, by insertion of a single copy of the current unit cell of the algorithm into each of the environments.

From this abstract framework, it would seem that, starting with an arguably poor initial guess, we would improve our approximation of the bulk unit cell with each extension of the system. However, this would come at the expense of an increasing memory footprint. Fortunately, this issue can be resolved by wise use of dummy tensor legs, environments and indexing modulo L [38].

While introducing the MPS formalism, we stated that the left-most $\Gamma_{[1]}$ tensor and the right-most $\Gamma_{[L]}$ are matrices, with only one virtual leg each. However, we can insert a dummy leg of size 1 to each of these tensors, the left one for $\Gamma_{[1]}$ and right one for $\Gamma_{[L]}$. This would amount to simply increasing the dimensionality of the array representing the tensor, without addition of any new entries. As a result, each tensor in an MPS constructed in this fashion would be an order-3 one. Thanks to this solution, it is now possible to perform contraction of $\Gamma_{[1]}$ with $\Gamma_{[L]}$ over the dummy legs, to obtain a two-site tensor.

Second thing to notice, is that after initial growth of the virtual legs of environments, with each contraction with a *column* consisting of tensors coming from the ket state, the MPO, and the bra state, those bond-sizes eventually saturate at χ (thanks to the truncation of Schmidt values). This means that after reaching such a state of environments we can contract arbitrarily many columns into them, one at a time, without increasing their total size. To fully understand the implementation to be presented, let us denote the number of columns contracted with given environment as its *age*.

With the above setup we can add one last modification to the finite DMRG method. In the standard, two-site version of the algorithm, the last update of the rightward sweep was

performed on the $\Theta_{[L-1,L]}$ tensor. However, with the addition of dummy legs and usage of indices modulo L, we may take another step and optimize the $\Theta_{[L,1]}$ tensor, which will be followed by a standard update of environments. This operation conceptually corresponds to creation of the orthogonality center from the last site of the central system, and the left-most one from the copy of the chain that should be incorporated into the right environment. Yet, with the extensions presented above we do not need to make any copies of the system.

To fully illustrate that with this approach each environment truly incorporates L new columns at the end of each sweep, we show in Fig. 2.40 the ages of the left and right environments throughout the first two sweeps of iDMRG. Transition between each state of the environments is done via the update of the two-site orthogonality center. Moreover, each case in which there was no need to store environment at site i was denoted simply by the "-" symbol.

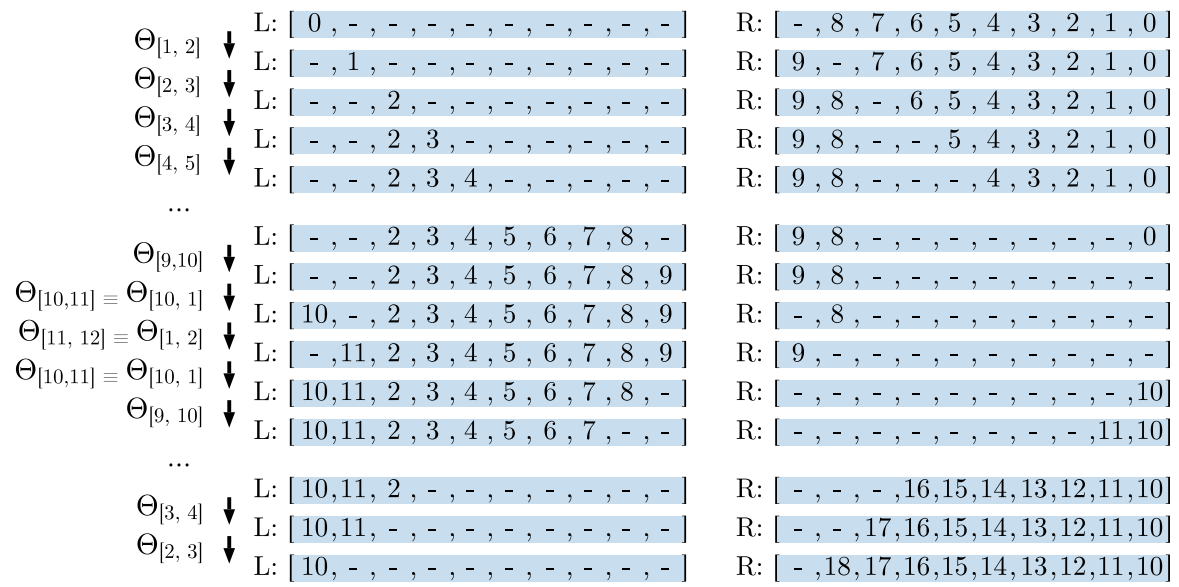


Figure 2.40: Change of the environment ages during the first two sweeps of the iDMRG algorithm.

One might wonder whether the above approach preserves the canonical form. Indeed, this becomes particularly clear, when we use the Vidal representation. With its help we can obtain a two-site tensor consisting of $\Gamma_{[1]}$, the singular values $\Lambda_{[0]} \equiv \Lambda_{[L]}$ assigned to the dummy legs, and $\Gamma_{[L]}$. For this tensor to become a true orthogonality center $\Theta_{[1,L]}$ we need to incorporate in it the $\Lambda_{[1]}$ and $\Lambda_{[L-1]}$ singular values.

The last topic worth discussing in the context of iDMRG is the convergence criterion. As the system size grows between consecutive iterations, it is pointless to determine whether to end the algorithm basing on the total energy of the system, as it can be done in the finite DMRG. Rather than that, we can measure the average energy per site, and terminate the calculations after reaching the saturation of this value.

2.8 Charge conservation

Computational cost of operations performed on tensors of various kinds can be significantly reduced, when these objects preserve the highly sparse block-diagonal structure. Such a

situation occurs, for example, when a given Hamiltonian $\hat{H}$ commutes with an unitary U. In this scenario, $\hat{H}$ can be written in the basis of U, and both of them can be diagonalized simultaneously.

We can define a general *charge rule* [38], which enforces the block structure of tensors and selects the entries, which can be discarded. Let us exemplify it on the case of a spin-1/2 system. In this scenario, we can pick the basis of the $\hat{S}^z$ operator and denote it as $\{|\alpha\rangle\} = \{|\uparrow\rangle, |\downarrow\rangle\}$. The eigenvalues corresponding to each of the basis vectors can be rescaled (to avoid numerical errors) according to the $q = 2\hat{S}^z/\hbar$ formula, which gives the values of 1 and -1 assigned to $|\uparrow\rangle$ and $|\downarrow\rangle$, respectively. The obtained q values are the *charges* in question (also reffered to as *quantum numbers*), and similarly as the $\{|\alpha\rangle\}$ basis can be written down in the form of a vector as $\bar{q} = (1, -1)$.

Let us write down three spin operators in the $\{|\alpha\rangle\}$ basis, where we relabel each basis vector by its corresponding charge

$$\hat{S}^z = \begin{array}{c} \\ 1 \\ -1 \end{array}\!\!\begin{array}{c} \begin{array}{cc} 1 & -1 \end{array} \\ \begin{pmatrix} 1/2 & 0 \\ 0 & -1/2 \end{pmatrix} \end{array}, \quad \hat{S}^+ = \begin{array}{c} \\ 1 \\ -1 \end{array}\!\!\begin{array}{c} \begin{array}{cc} 1 & -1 \end{array} \\ \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} \end{array}, \quad \hat{S}^- = \begin{array}{c} \\ 1 \\ -1 \end{array}\!\!\begin{array}{c} \begin{array}{cc} 1 & -1 \end{array} \\ \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix} \end{array}. \tag{2.79}$$

We can note that for each of these operators we can find a value Q, which satisfies the following rule

$$\bar{q}^{[1]}_{a_1} - \bar{q}^{[2]}_{a_2} \neq Q \implies \hat{S}^{a_1}_{a_2} = 0. \tag{2.80}$$

The Q in the above formula is called the *total charge*, and for $\hat{S}^z$ is equal to 0, for $\hat{S}^+$ is 2, and for $\hat{S}^-$ is -2. Eq. (2.80) was defined for simple matrices, while in general we would like to extend it to the case of higher-dimensional tensors. To achieve this goal let us recall that any matrix can be expressed in Penrose notation as a shape with two legs. Moreover, we can assign the incoming arrow to the first bond (labeled as a_1), and an outgoing one to the second (denoted as a_2), as we already did multiple times in this work. Finally, we can modify Eq. (2.80) by including a sign ζ, corresponding to the direction of an arrow of given leg of the tensor. We will follow the convention assigning a +1 value to the incoming legs, and -1 to the outgoing ones. Thus, with this new notation we can reformulate Eq. (2.80) as

$$\zeta^{[1]}\bar{q}^{[1]}_{a_1} + \zeta^{[2]}\bar{q}^{[2]}_{a_2} \neq Q \implies \hat{S}^{a_1}_{a_2} = 0. \tag{2.81}$$

Even though introduction of ζ values seems redundant at first sight, it allows for a simple generalization of the charge rule to the case of order-n tensors, which can be written as

$$\forall\, a_1, a_2, \ldots a_n : \zeta^{[1]}\bar{q}^{[1]}_{a_1} + \zeta^{[2]}\bar{q}^{[2]}_{a_2} + \ldots + \zeta^{[n]}\bar{q}^{[n]}_{a_n} \neq Q \implies T_{(a_1,a_2,\ldots,a_n)} = 0. \tag{2.82}$$

In the above equation we also utilized the fact that good quantum numbers are additive. In Fig. 2.41 we explicitly show, how this property translates into the assignment of charges to the bonds of an MPS.

In Table 2.1 we show how the charge rule for $Q = 0$ can be applied in the case of the second tensor shown in Fig. 2.41. We can plainly see the sparse structure of the tensor, as only 6 out of 16 entries can be non-zero.

The last thing worth mentioning in the context of conservation of quantum numbers, is that formally any operator with $Q \neq 0$ does not preserve the charge of the state it acts upon. However, it retains the block structure, and therefore can be used during the generation of a charge conserving MPO. The operators which cannot be naively used for that purpose, are the ones without a well defined total charge, which is the case, e.g., with $\hat{S}^x$ or $\hat{S}^y$.

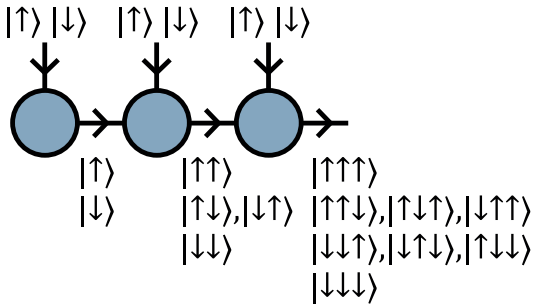


Figure 2.41: Conservation of quantum numbers for the first three tensors of an MPS. The charges are assigned to the bonds from left to right, following the direction of arrows. In the case of virtual legs the basis vectors with the same quantum number were grouped together.

Incoming		Outgoing - Right			
Left	Top	$\|\uparrow\uparrow\rangle$ (2)	$\|\uparrow\downarrow\rangle$ (0)	$\|\downarrow\uparrow\rangle$ (0)	$\|\downarrow\downarrow\rangle$ (-2)
$\|\uparrow\rangle$ (1)	$\|\uparrow\rangle$ (1)	$\neq 0$	0	0	0
$\|\uparrow\rangle$ (1)	$\|\downarrow\rangle$ (-1)	0	$\neq 0$	$\neq 0$	0
$\|\downarrow\rangle$ (-1)	$\|\uparrow\rangle$ (1)	0	$\neq 0$	$\neq 0$	0
$\|\downarrow\rangle$ (-1)	$\|\downarrow\rangle$ (-1)	0	0	0	$\neq 0$

Table 2.1: Identification of entries in the central tensor from Fig. 2.41, which can be discarded for $Q = 0$. The left two columns store the basis vectors of the incoming legs (and their corresponding charges), while the top row defines analogous values for the outgoing leg. The 0 and $\neq 0$ values present in the table should be thought of as the components of an order-3 tensors, stored at the intersection of the three indices.

2.9 Simulation of 2D systems via 1D methods

We will conclude this chapter by demonstrating how 1D methods can be used to model the 2D systems, which are ultimately the subject of this work. This task can be achieved by simply projecting a 2D system onto a 1D MPS, as shown in Fig. 2.42.

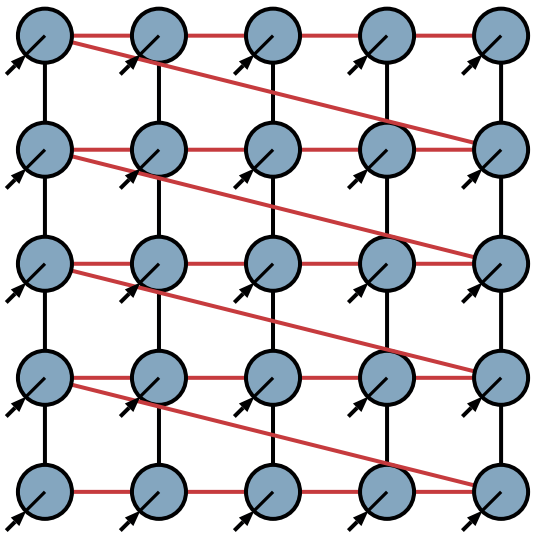


Figure 2.42: Projection of a 2D system onto a 1D "snake" MPS.

We can immediately see that this mapping introduces certain long-range interactions between the sites of the chain, which are neighbors in the 2D lattice, but are farther apart in the MPS. Because of that the naive TEBD method cannot be used for this class of systems. Moreover, MPOs representing the long-range interactions have larger legs than the short-range ones. As a result, application of such an MPO to an MPS increases the bond-size of the latter. This in turn increases the computation time and makes it more difficult to obtain convergence, e.g., while using the DMRG method.

THIRD CHAPTER

SIMULATION OF 2D PHYSICAL MODELS WITH 1D ISOMETRIC TENSOR NETWORKS

In this chapter we will use 1D isometric tensor networks to simulate two strongly correlated physical models. In the first section we will use finite DMRG, without conservation of quantum numbers, to obtain the ground state of the transverse field Ising (TFI) model on a square lattice. Although this model was extensively studied in the past, it will serve us as a benchmark for methods tailored specifically for 2D systems that will be presented in Chapter 5. The purpose for which we perform these calculations also justifies the resignation from the conservation of quantum numbers, as the methods to be presented in the further part of this work do not yet use this formalism.

In the second section of this chapter we will use both finite and infinite versions of DMRG to analyse magnetic phases of the spin-3/2 Heisenberg XXZ model on a honeycomb lattice, which can effectively model the monolayer of CrI_3. This material has been reported to exhibit the ferromagnetic order in the off-plane axis, which has received a lot of attention from the condensed matter physics community, and suggests the usefulness of its thorough examination. Results of these studies were published in Ref. [59].

3.1 Transverse field Ising model on a square lattice

The Hamiltonian of the transverse field Ising model on a square lattice is defined as

$$\hat{H} = -\sum_{\langle i,j\rangle, i<j} J\hat{\sigma}_i^x\hat{\sigma}_j^x - g\sum_i \hat{\sigma}_i^z, \tag{3.1}$$

where $\langle i, j\rangle$ denotes iteration over the nearest neighbours in the lattice. We obtain the ground states of this model for the fixed value of $J = 1$ and varying g.

The main goal of this section is to obtain benchmark results with the use of an MPS method, with which we will be able to compare two-dimensional algorithms developed in the further part of this work. Because of that, we are not interested in systems having little to no entanglement, as it is known that these can be simulated efficiently with 1D methods. Rather than that, we will focus on systems with moderate amount of entanglement, which at the same time can be found in each of the system sizes considered (in this place we will study only rectangular $L \times L$ lattices).

Thus, in order to get test cases of comparable difficulty for different values of L, we needed to firstly locate critical points g_c, as these may fall upon different values of the g-space, due to the properties of finite size systems. For each obtained ground state MPS, we measured the von Neumann entanglement entropy for all bi-partitions of the chain, and picked among those values the maximal one, denoted as S_{max}. This allowed us to assign one S_{max} value to each g, and locate the point of maximal entanglement for each system size L.

We run the DMRG calculations for $g \in \{0.5, 1.0, ..., 3.5, 4.0\}$, and maximal bond-size χ increasing gradually by 100 every 40 sweeps, and eventually stopping at 1200, to get the first vague location of the critical point. Later, after finding the area in which the peak of entanglement was present in our results, we run additional calculations, this time incrementing g in steps of length 0.1. Combination of all of the described results allowed us to generate the plot of maximal entanglement S_{max} for each value of g, which is shown in Fig. 3.1. For each L we marked the g_c with a star symbol.

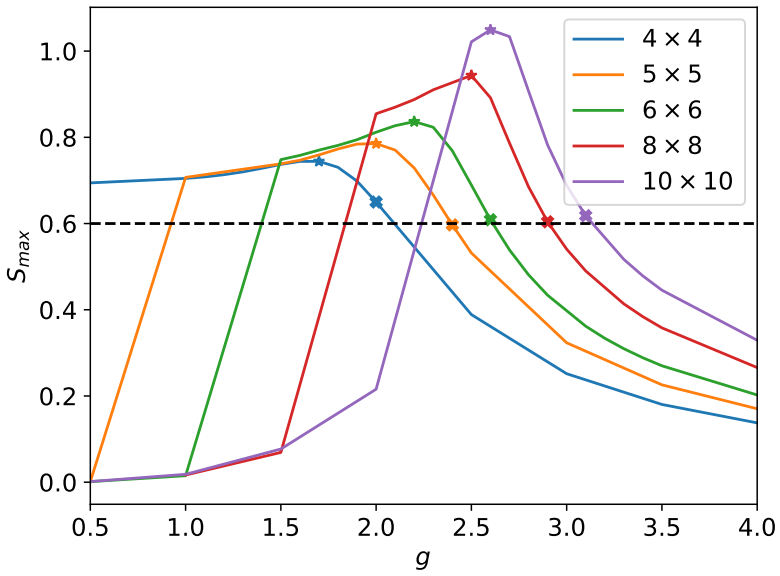


Figure 3.1: Maximal entanglement in the ground state MPS of the TFI model, for varying values of g. The critical points are marked with star symbols, while cases of moderate entanglement, being the benchmarks for methods to be presented, are marked with crosses.

To make sure that the ground states which we obtained from the DMRG are correct, we checked the scaling of the critical value g_c, which is shown in Fig. 3.2. In the thermodynamic limit we obtain the value of 3.236 (the intersection of the fitted line with the vertical axis in the plot), which is in good agreement with the value of 3.044 obtained from infinite methods [39–42], considering the finite size of the lattices we studied.

Knowing that the results obtained from the DMRG are correct, we moved on to selecting the benchmark cases for the 2D methods. We picked the ground states with $S_{max} \approx 0.6$, which are marked with crosses in Fig. 3.1. In Chapter 5 we will focus our analysis on the number of variational parameters used in a 2D tensor network, while trying to reproduce these results, therefore it is of great importance for us to know, how precisely can we represent the selected ground states with MPSs storing as few parameters as possible. To get this information we conducted variational compression of the MPSs obtained from the DMRG, reducing the maximally allowed bond-size by 1 in each step, and performing 4 sweeps across the whole

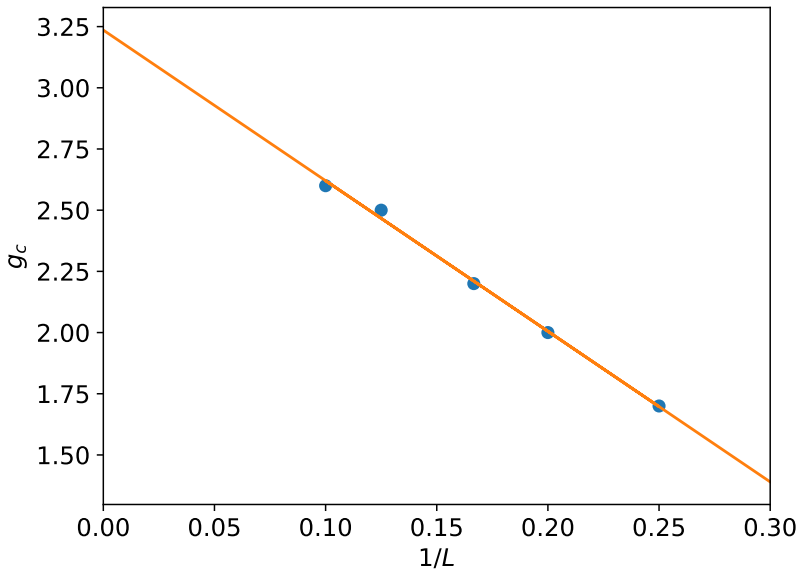


Figure 3.2: Scaling of the critical value g_c. We can see the leveling off of the g_c values for increasing system size L.

chain. The energies and maximal entanglement entropy of states resulting from the last 1000 steps of compression of the ground state for $g = 3.1$ on a 10×10 lattice are illustrated in Fig. 3.3 and Fig. 3.4, respectively.

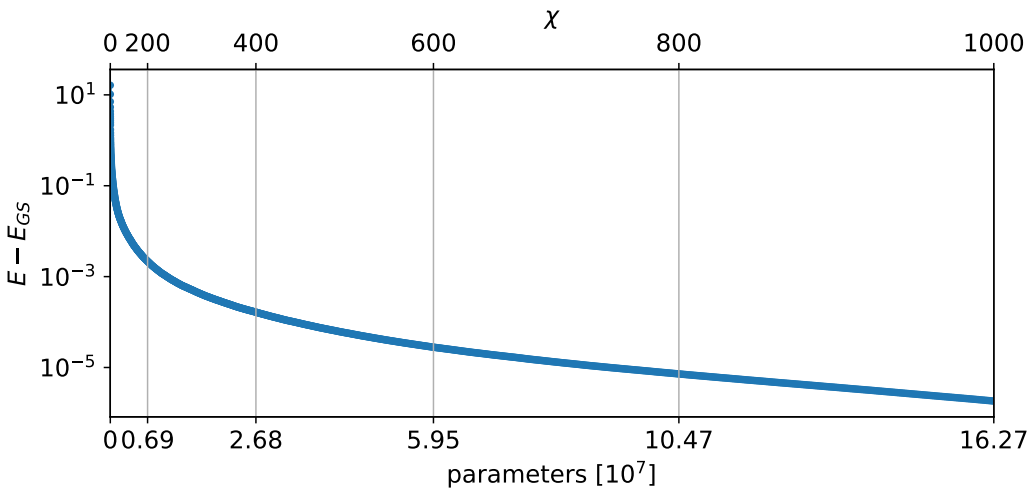


Figure 3.3: Differences between the energies of the MPSs resulting from compression, and the ground state energy in the TFI model with $g = 3.1$ on a 10×10 lattice, obtained with DMRG. The plot should be read from right to left, as firstly the values for large bond-sizes χ were obtained.

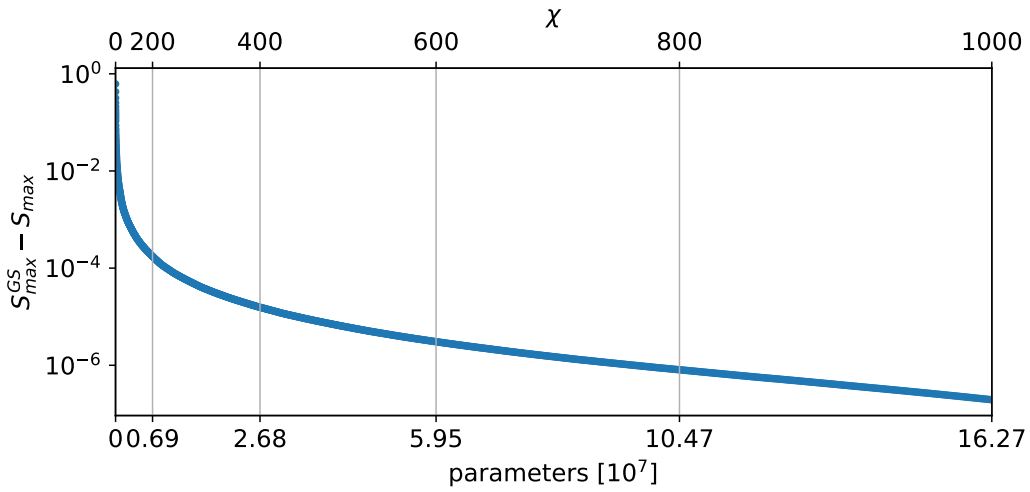
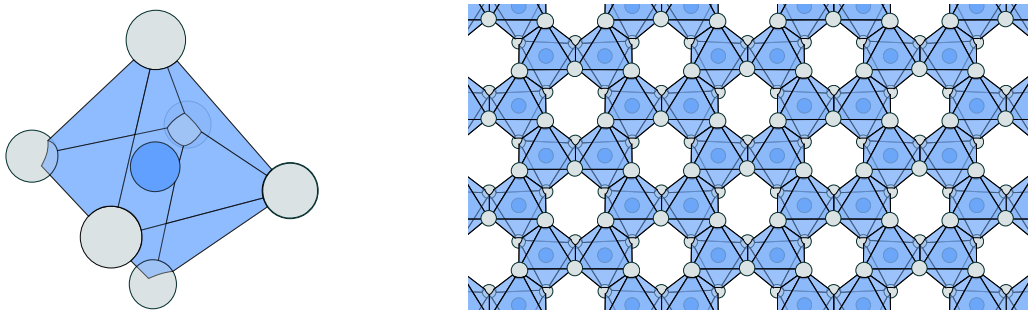


Figure 3.4: Differences between the maximal entanglement entropy appearing in the ground state of the TFI model for $g = 3.1$ on a 10×10 lattice (obtained via DMRG), and analogous property of states resulting from the variational compression.

3.2 Heisenberg XXZ model on a honeycomb lattice

Recent years have seen a significant amount of theoretical and experimental research into the diverse class of two-dimensional van der Waals crystals, which include semiconductors, superconductors, semi-metals, topological insulators, charge density waves materials, ferroelectrics, and magnetics [60–71]. These materials have numerous potential uses in cutting-edge types of electronics, such as spintronics, valleytronics, and optoelectronics [62, 72–76]. Among these materials, the monolayers of CrI_3 and $Cr_2Ge_2Te_6$ have been discovered to have ferromagnetic order [77, 78], while monolayers of $FePS_3$ manifest antiferromagnetic order [79, 80]. In several other materials, the magnetic order has likewise been predicted and then empirically verified [77, 81–93].

In this section we will focus on the analysis of the aforementioned monolayer of CrI_3. Its crystalline structure consists of chromium atoms arranged in a honeycomb lattice (shown in Fig. 3.5b), each having an octahedral environment with iodine atoms located at the edges (Fig. 3.5a).



(a) (b)

Figure 3.5: (a) Chromium atom with its octahedral iodine environment. (b) The honeycomb crystalline lattice of CrI_3 [59].

It has been shown that the magnetic order in CrI_3 can be described by the spin-3/2 XXZ

Heisenberg model [43, 94], whose Hamiltonian is defined as

$$\hat{H} = -\left(J \sum_{\langle i,j\rangle, i<j} \bar{S}_i \cdot \bar{S}_j + \lambda \sum_{\langle i,j\rangle, i<j} \hat{S}_i^z \hat{S}_j^z + \sum_i D(\hat{S}_i^z)^2 \right), \tag{3.2}$$

where the Z axis was chosen as the off-plane direction. The J parameter is associated with the Heisenberg isotropic exchange interaction between spin particles. λ stands for the anisotropic super-exchange that results from the spin-orbit interactions of ligand I atoms across the $\simeq 90$ degree $Cr-I-Cr$ bonds. Finally, the D term describes the single ion anisotropy that results from the interaction between the spin-orbit coupling and the distorted octahedral environment of the Cr atoms.

It should be noted that by the Mermin-Wagner theorem [95] any long-range ordering could not emerge from Hamiltonian in Eq. (3.2) if not for the λ and D terms, which break the spin-rotational invariance. By means of *ab-initio* calculations, parameters for isolated CrI_3 have been determined, and are equal to $D = 0$ meV, $J = 2.2$ meV and $\lambda = 0.09$ meV [43, 94]. For this set of variables this material exhibits the ferromagnetic order in the off-plane (Z) axis, below the Curie temperature $T_C = 45$ K [78]. However, it has been shown that other phases can be achieved in CrI_3 by introducing defects [96], strain [97, 98] and charge doping [98, 99], which effectively enhance magnetic anisotropy.

In this place we will analyse the magnetic ordering of the Heisenberg model ground state in a wide range of parameter space, and compare obtained results with classical predictions. Hamiltonian given in Eq. (3.2) can exhibit four different magnetic phases: in-plane ferromagnetic, in-plane antiferromagnetic, off-plane ferromagnetic and off-plane antiferromagnetic which are shown in Fig. 3.6. When the system parameters are changed, these phases compete with one another, with the ferromagnetic order being preferred for $J > 0$, and the antiferromagnetic one for $J < 0$. We will use these four product states as the classical approximation of the model, and pick the one giving the smallest expectation energy of the Hamiltonian as the ground state.

To obtain the ground state of the fully quantum Hamiltonian we firstly conducted calculations using finite DMRG, for the lattice with periodic boundary conditions along both of the basis vectors, making it effectively a torus. Because of the big local Hilbert space size, being equal to 4 for a spin-3/2 particle, we utilized the conservation of quantum numbers, with the total value of spin in the Z axis being the preserved property. With this setup, for each sector of the Hamiltonian we obtained the ground state and first two excited ones. After the calculations for each sector were finished, we gathered all of the results and chose two smallest eigenvalues (and corresponding state-vectors), as the ground state and the first excited state energies of the whole Hamiltonian. Finally, to check the scaling of the results predicted by finite DMRG we performed separate computations using iDMRG in the infinite cylinder geometry.

We firstly obtained the ground state of the isolated CrI_3 on a lattice consisting of 9 unit cells, giving in total 18 spin-3/2 particles. The obtained energy of $E_{GS} = -139.1175$ meV was equal to the one predicted by the classical approximation, as expected from a simple product state. Then, we investigated the properties of the system for varying parameters of the Hamiltonian. As the impact of the single ion anisotropy (corresponding to the D variable) is substantially smaller than the one coming from the exchange interaction between particles [43], we carried out the calculations for just three different values of $D \in \{-0.4, 0.001, 0.4\}$, while focusing on a larger parameter space of the remaining coefficients $J, \lambda \in \{-0.5, -0.45, ..., 0.45, 0.5\}$. Because the number of all considered combinations of parameters is equal to 1323 we re-

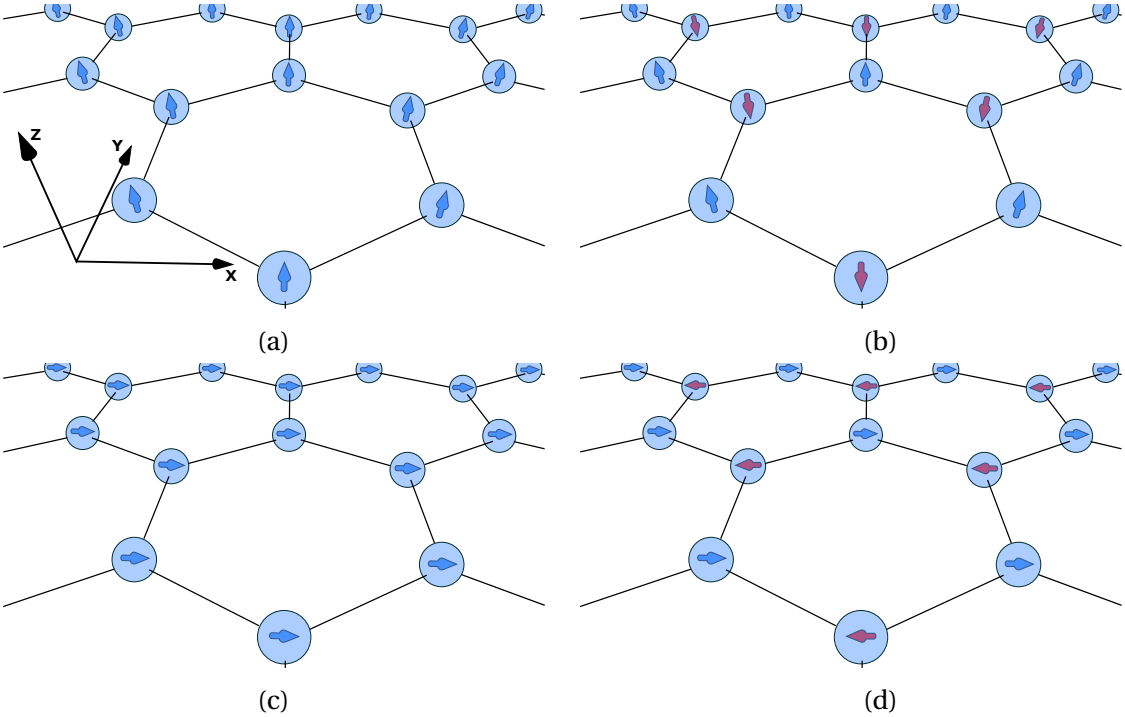


Figure 3.6: Four possible magnetic orderings appearing in Eq. (3.2). (a) Ferromagnetic off-plane. (b) Antiferromagnetic off-plane. (c) Ferromagnetic in-plane. (d) Antiferromagnetic in-plane [59].

stricted the lattice size to 4 unit cells, giving in total 8 spin-3/2 particles. Although this system size seems to be small, once again we need to take into account big local Hilbert space size. If we were to conduct a similar analysis using exact diagonalization (or Lanczos algorithm), the storage of the Hamiltonian alone would require roughly 35 GB of memory. This fact, in combination with a large number of investigated combinations of parameters, would make the whole analysis infeasible.

In Fig. 3.7 we show the magnetic phases predicted by the classical approximation of the Hamiltonian. In order to recognize the phases of state-vectors obtained from the DMRG calculations, and compare them with classical predictions, we firstly measured the average value of spin in the Z axis, which is illustrated in Fig. 3.8. Due to the fact that, for instance, the ferromagnetic and antiferromagnetic phases in the XY plane both have average values of spin in the Z axis equal to 0, this value alone would not be sufficient to differentiate between all possible phases. The additional information needed is the average in-plane correlation. For any chosen bond in the lattice, this value is defined as $(\langle \hat{S}_i^+ \hat{S}_j^- \rangle + \langle \hat{S}_i^- \hat{S}_j^+ \rangle)/2$, where i and j are the indices of the nearest neighboring sites. This property is depicted in Fig. 3.9.

The average value of spin in the Z axis and in-plane correlations appear in four different combinations. Firstly, the spins can be fully aligned in the off-plane direction with no coexisting correlation in the XY plane. This situation corresponds to the ferromagnetic order in the Z axis (top-right corner of the phase diagram). Secondly, the average spin value in the off-plane axis and in-plane correlations are both equal to 0. This information alone is not sufficient to draw a definite conclusion regarding the corresponding phase, thus we looked closely at the expectation value of spin at each site. The spins were fully aligned in the Z axis in an alternating fashion, giving the antiferromagnetic order (bottom-left corner). The last two cases correspond to a situation with the average value of spin in the off-plane axis being

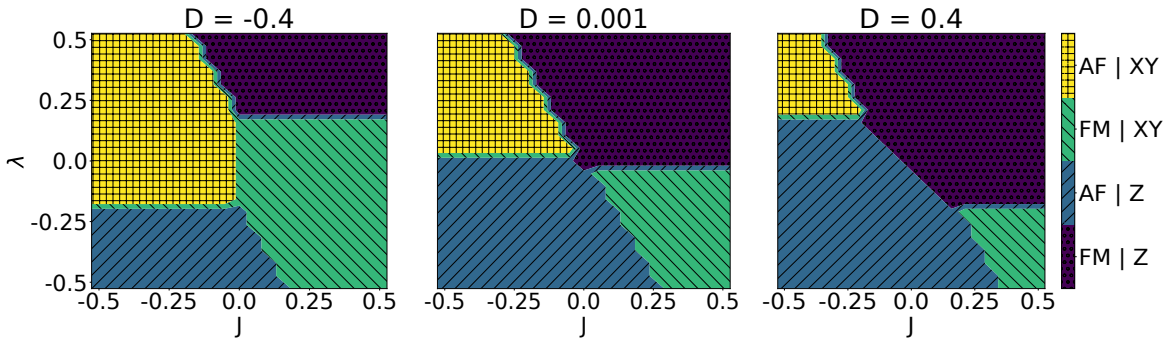


Figure 3.7: Magnetic phases predicted by the classical approximation [59].

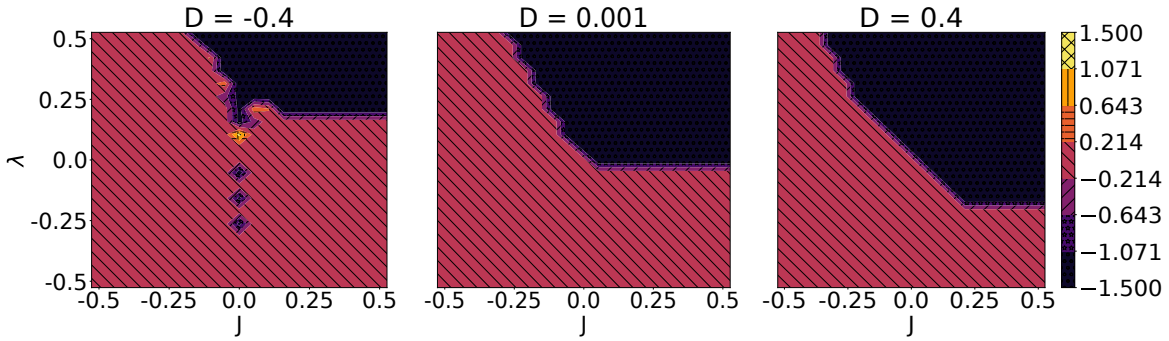


Figure 3.8: The average value of spin in the Z axis [59].

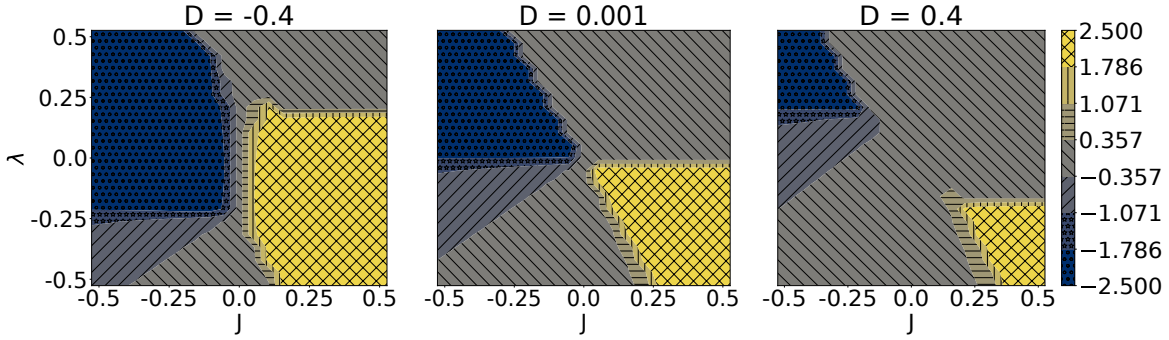


Figure 3.9: The average correlation in the XY plane between the nearest neighbouring spins [59].

equal to 0. The only factor differentiating between these two cases is the correlation in the XY plane. The positive one gives the ferromagnetic order (bottom-right corner), while the negative one results in the antiferromagnetic phase (top-left corner). We can therefore notice that the magnetic phase of Hamiltonian in Eq. (3.2) can be approximated with high fidelity within the investigated parameter space.

To get a better picture of the properties of the system under study we also investigated the energy gap and average entanglement entropy present within the MPS (the average was taken over all bisections of the chain), which are shown in Fig. 3.10 and Fig. 3.11, respectively. The energy gap is present within the off-plane phases and increases with the magnitude of D (as expected from the formalism adopted in the definition of the Hamiltonian in Eq. (3.2)),

while the in-plane phases are gapless. On the other hand, entanglement is present in all of the phases, with the exception of the ferromagnetic one in the Z axis.

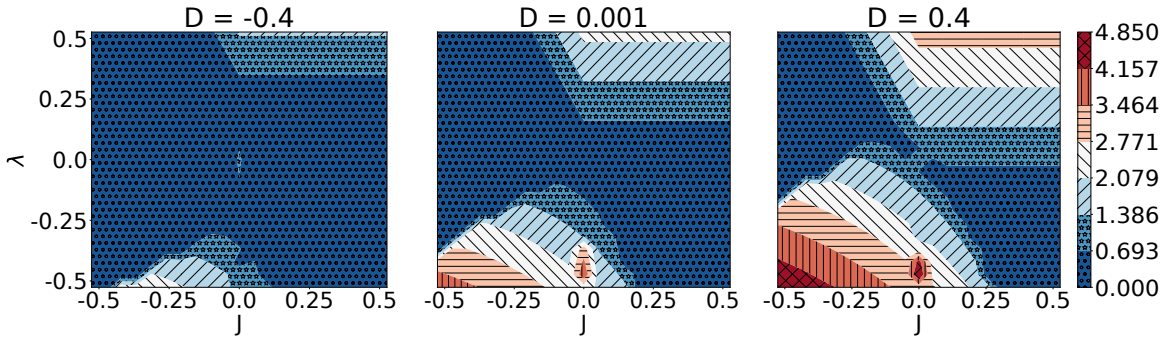


Figure 3.10: The energy gap between the ground state and the first excited one [59].

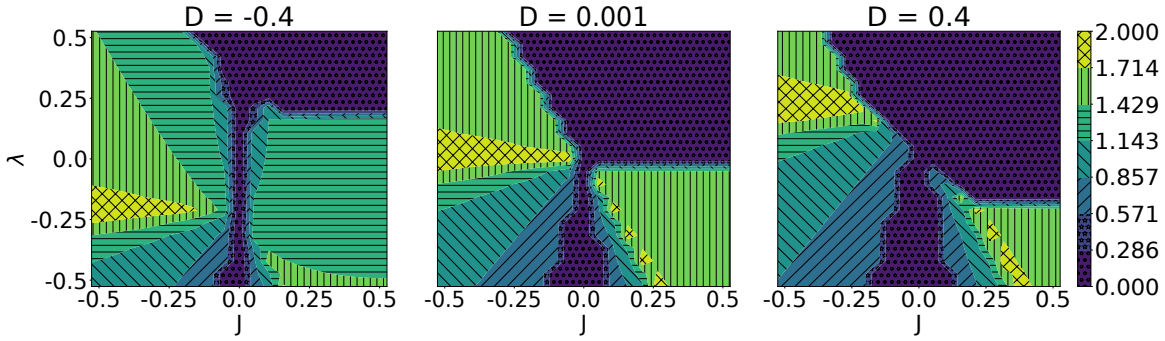


Figure 3.11: The average entanglement entropy in the XXZ model [59].

The last property that we analysed was the correlation energy, defined as the difference between the energy resulting from the DMRG calculations and the classical approximation. It is illustrated in Fig. 3.12. We can see that it is the largest in magnitude in the case of in-plane phases, and its maxima coincide with the maxima of entanglement entropy.

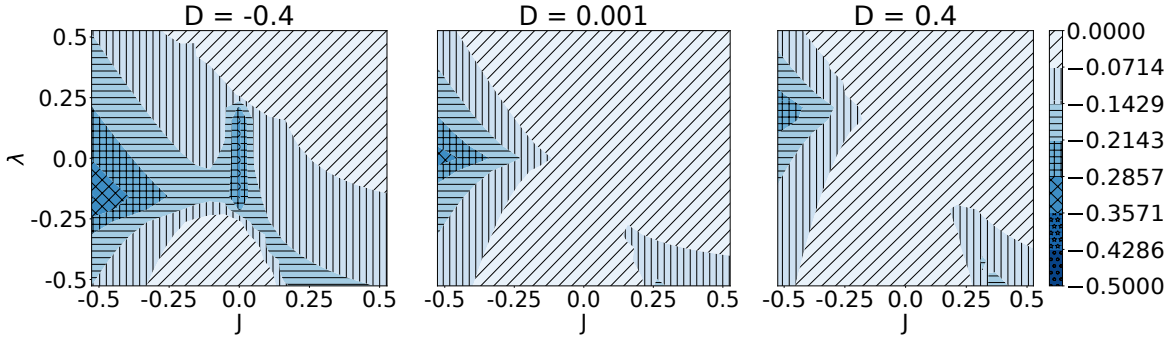


Figure 3.12: The correlation energy [59].

To check the scaling of the properties obtained from finite DMRG we ran the iDMRG on an infinite cylinder, on a set of parameters corresponding to each corner of the three types of phase diagrams presented above. For each such case we conducted the calculations

three times, for increasing values of bond-size $\chi \in \{1000, 2000, 3000\}$. The chosen unit cell of the algorithm consisted of four sites. We also restricted calculations to the subspaces with $\langle \hat{S}^z \rangle = 0$ and $\langle \hat{S}^z \rangle = 3/2 \cdot 4 = 6$, because these were the only candidates for the true ground state. With this setup, we reproduced the results given by the finite method. Mean energy per site, average value of spin in the Z axis and mean in-plane correlations are listed in Tables 3.1 to 3.3.

D	J	λ	DMRG	iDMRG		
				$\chi = 1000$	$\chi = 2000$	$\chi = 3000$
-0.4	-0.5	-0.5	-2.61813	-2.61394	-2.61394	-2.61394
	-0.5	0.5	-1.52755	-1.49697	-1.49697	-1.49697
	0.5	-0.5	-1.52755	-1.49697	-1.49697	-1.49697
	0.5	0.5	-2.475	-2.475	-2.475	-2.475
0.001	-0.5	-0.5	-3.48795	-3.48618	-3.48618	-3.48618
	-0.5	0.5	-1.78903	-1.76248	-1.76248	-1.76248
	0.5	-0.5	-1.78903	-1.76248	-1.76248	-1.76248
	0.5	0.5	-3.37725	-3.37725	-3.37725	-3.37725
0.4	-0.5	-0.5	-4.36585	-4.36492	-4.36492	-4.36492
	-0.5	0.5	-2.10218	-2.07989	-2.07989	-2.07989
	0.5	-0.5	-2.10218	-2.07989	-2.07989	-2.07989
	0.5	0.5	-4.275	-4.27499	-4.27499	-4.27499

Table 3.1: Mean energy per node [59].

D	J	λ	DMRG	iDMRG		
				$\chi = 1000$	$\chi = 2000$	$\chi = 3000$
-0.4	-0.5	-0.5	0	0	0	0
	-0.5	0.5	0	0	0	0
	0.5	-0.5	0	0	0	0
	0.5	0.5	1.5	1.5	1.5	1.5
0.001	-0.5	-0.5	0	0	0	0
	-0.5	0.5	0	0	0	0
	0.5	-0.5	0	0	0	0
	0.5	0.5	1.5	1.5	1.5	1.5
0.4	-0.5	-0.5	0	0	0	0
	-0.5	0.5	0	0	0	0
	0.5	-0.5	0	0	0	0
	0.5	0.5	1.5	1.5	1.5	1.5

Table 3.2: Average value of spin in the Z axis [59].

D	J	λ	DMRG	iDMRG		
				$\chi = 1000$	$\chi = 2000$	$\chi = 3000$
-0.4	-0.5	-0.5	-0.42120	-0.39787	-0.39787	-0.39787
	-0.5	0.5	-2.35530	-2.33065	-2.33065	-2.33065
	0.5	-0.5	2.35530	2.33065	2.33065	2.33065
	0.5	0.5	0	0	0	0
0.001	-0.5	-0.5	-0.31161	-0.30305	-0.30305	-0.30305
	-0.5	0.5	-2.38443	-2.35844	-2.35844	-2.35844
	0.5	-0.5	2.38443	2.35844	2.35844	2.35844
	0.5	0.5	0	0	0	0
0.4	-0.5	-0.5	-0.25103	-0.24679	-0.24679	-0.24679
	-0.5	0.5	-2.34179	-2.31411	-2.31411	-2.31411
	0.5	-0.5	2.34179	2.31411	2.31411	2.31411
	0.5	0.5	0	0	0	0

Table 3.3: Average correlation in the XY plane [59].

FOURTH CHAPTER

SIMULATION OF QUANTUM CIRCUITS WITH 1D ISOMETRIC TENSOR NETWORKS

In the previous chapter with the use of MPS methods we obtained the groudstates of the TFI model for a chosen parameter set, which will serve us as benchmark for our work in consecutive chapters, when testing the 2D tailored algorithms. However, these states are characterized by a moderate amount of entanglement. We would like to evaluate how effectively the 2D methods introduced in this work can describe states with high levels of entanglement, in order to fully explore their potential.

For that purpose, we will attempt to simulate random quantum circuits. We will do this for two reasons. First off, this task was created specifically to generate as much entanglement in a quantum system with as few operations as possible. Secondly, 1D algorithm used to simulate these circuits can be extended in a straightforward way to the 2D case, which alleviates all the issues associated with different rates of convergence of different methods (e.g. as is in the case of TEBD and DMRG).

The aforementioned random quantum circuits were used by Google Inc. in their famous "quantum supremacy experiment" [44]. At first, Google claimed that reproducing their results by performing simulation on the biggest existing supercomputer would take 10,000 years. This initial hypothesis was overstated, according to subsequent research [100–102], however, running such a simulation still requires an exponential amount of resources relative to the number of considered qubits.

We will firstly introduce the quantum supremacy task in a simplified 1D version, along with the MPS-based algorithms allowing for its simulation. Then, we will extend these concepts to the 2D variant, paying attention to any nuances between the two versions.

4.1 Quantum supremacy task in 1D

The quantum circuit that can be executed on a 1D quantum computer, which we will initially attempt to simulate, is shown in Fig. 4.1. It consists of alternating *layers* of single- and two-qubit quantum gates. Using the nomenclature introduced by Google, we will refer to a pair consisting of a single-qubit gate layer, followed by a two-qubit gate one, as *cycle*.

Each single-qubit gate is chosen randomly from a set $\{\sqrt{X}, \sqrt{Y}, \sqrt{W}\}$, where $W = (X + Y)/\sqrt{2}$. Additionally, once a given gate is picked in one layer, it cannot be selected in the

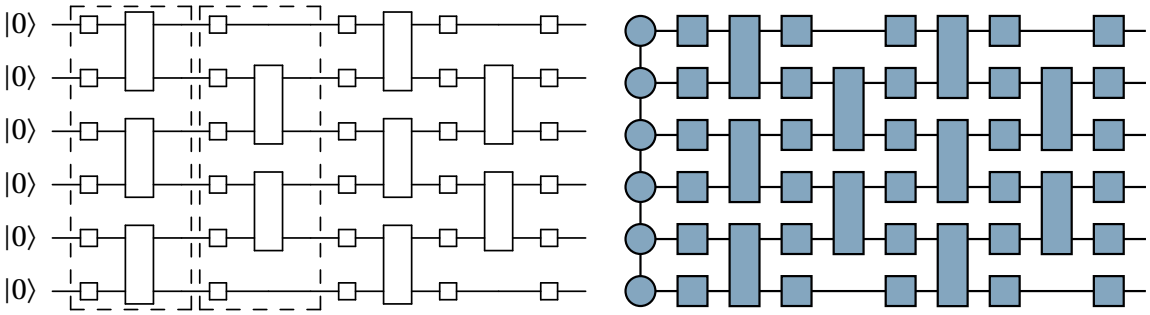


Figure 4.1: (Left) Random quantum circuit executed on a 1D quantum computer. First two cycles are marked with dashed lines. (Right) Tensor network representation of the same quantum circuit.

subsequent layer of single-qubit gates. In the case of two-qubit gates just a single type of gate will be used throughout the whole circuit. However, as we will see shortly, the type of chosen multi-qubit gate will have a huge impact on the difficulty of the task. The three kinds of gates that will be used in this work are *CNOT*, *CZ* and *ISWAP*. Finally, two consecutive layers of multi-qubit gates will act on different pairs of qubits. For example, if the second layer was consisting of gates that should be applied on even pairs of qubits, the fourth one will act on the odd ones.

From Fig. 4.1 we can see that the illustrated circuit can be directly translated into the tensor network framework, if we simply represent the qubits as an MPS rotated by 90 degrees from its previously used orientation. None of the quantum gates need any additional transformations. Moreover, we can see a resemblance between the application of quantum circuit onto a set of qubits, and the MPS-based techniques. This observation gave rise to the first algorithm simulating *noisy* quantum computers [45].

In fact, the algorithm works exactly in the same way as the TEBD method, with the only difference being that the evolution operators are replaced with quantum gates. Although a DMRG-based method was also proposed [103], we will reproduce only the former one, as it will be easier to extend it to two dimensions.

Let us remind that to avoid exponential explosion of resources used during the execution of TEBD, we truncate the bonds resulting from SVDs. This is the place, where we are approximating the simulation, by reducing the entanglement to yield a finite *fidelity* of the application of a two-site gate. It should be noted that each single-qubit gate can be applied perfectly, as it cannot introduce any entanglement in the system. However, it can shuffle the state, which can later result in an increase in entanglement, when a two-qubit gate is applied.

Using the terminology from Ref. [45] we will call the fidelity of application of the nth two-site gate as f_n. Let us also denote the truncated state after the application of n two-qubit gates as $|\psi_T(n)\rangle$. We can immediately note that $|\psi_T(n)\rangle \overset{\text{SVD}}{\approx} U|\psi_T(n-1)\rangle$, where U denotes the two-site operation. Following Ref. [45], the two-qubit fidelity is defined as

$$f_n = |\langle\psi_T(n)|U|\psi_T(n-1)\rangle|^2. \tag{4.1}$$

Thanks to the canonical form of the MPS, we can estimate f_n by the sum of squares of singular values kept after the truncation, divided by the sum of squares all singular values [45], which can be written as

$$f_n = \Big(\sum_{i=1}^{\chi} s_i^2\Big) \Big/ \Big(\sum_{i=1}^{2\chi} s_i^2\Big). \tag{4.2}$$

If the MPS was not in the canonical form, we would have to contract all tensors stored in the two chains to calculate f_n, which once again shows huge gain resulting from the usage of this formalism.

However, the mere gain from estimating the precision of a two-qubit gate application would be negligible, if we could not correlate it in any way with the real fidelity of our noisy simulation in relation to an exact realisation of a quantum circuit. If we denote a perfect state resulting from application of n two-qubit gates as $|\psi_P(n)$ (that is a state, on which no truncations were conducted), we can write down the *multi-qubit fidelity* $\mathcal{F}(n)$ as

$$\mathcal{F}(n) = |\langle\psi_P(n)|\psi_T(n)\rangle|^2. \tag{4.3}$$

Although it was essential to use $|\psi_P(n)\rangle$ in the above definition, for large enough systems we would not be able to obtain it by means of classical simulation. In fact, this is exactly the state that we would like to approximate. Ideally, we would want to use consecutive f_n values, rather than having to rely on $|\psi_P(n)\rangle$. We can achieve this task by defining an orthonormal basis $\{|\alpha\rangle\}$, whose first vector is given as $|1\rangle \equiv |\psi_T(n-1)\rangle$. Then, we can write $|\psi_P(n-1)\rangle$ in this basis, assuming that the simulation is performed on N qubits, as

$$\begin{aligned} |\psi_P(n-1)\rangle &= \sum_{\alpha=1}^{2^N} p_\alpha|\alpha\rangle = p_1|1\rangle + \sum_{\alpha=2}^{2^N} p_\alpha|\alpha\rangle \\ &= \sqrt{\mathcal{F}(n-1)}|\psi_T(n-1)\rangle + \sum_{\alpha=2}^{2^N} p_\alpha|\alpha\rangle, \end{aligned} \tag{4.4}$$

where p_α are coefficients, which should not be confused with probabilities (p standing for perfect). We can conduct a similar procedure for the exact and truncated states after the application of the next U gate, resulting in

$$|\psi_P(n)\rangle = \sum_{\alpha=1}^{2^N} p_\alpha U|\alpha\rangle = \sqrt{\mathcal{F}(n-1)}U|\psi_T(n-1)\rangle + \sum_{\alpha=2}^{2^N} p_\alpha U|\alpha\rangle, \tag{4.5}$$

$$\begin{aligned} |\psi_T(n)\rangle &= \sum_{\alpha=1}^{2^N} t_\alpha U|\alpha\rangle = t_1 U|1\rangle + \sum_{\alpha=2}^{2^N} t_\alpha U|\alpha\rangle \\ &= \sqrt{f_n}U|\psi_T(n-1)\rangle + \sum_{\alpha=2}^{2^N} t_\alpha U|\alpha\rangle, \end{aligned} \tag{4.6}$$

where t_α coefficients are analogous to the p_α ones (t standing for truncated). After defining $|\psi_P(n)\rangle$ and $|\psi_T(n)\rangle$ in the above way, we can simply put them in Eq. (4.3), giving us

$$\begin{aligned} \mathcal{F}(n) = |\langle\psi_P(n)|\psi_T(n)|^2 &= \Big|\sum_{\alpha=1}^{2^N} p_\alpha^*\langle\alpha|U^\dagger \sum_{\alpha'=1}^{2^N} t_{\alpha'}U|\alpha'\rangle\Big|^2 \\ = \Big|\sum_{\alpha,\alpha'=1}^{2^N} p_\alpha^* t_{\alpha'}\langle\alpha|U^\dagger U|\alpha'\rangle\Big|^2 &= \Big|\sum_{\alpha=1}^{2^N} p_\alpha^* t_\alpha\Big|^2 \end{aligned}$$

$$= \left| \sqrt{\mathcal{F}(n-1) f_n} + \sum_{\alpha=2}^{2^N} p_\alpha^* t_\alpha \right|^2 \tag{4.7}$$

Because the quantum state must be normalized, and using that $p_\alpha \sim 1/\sqrt{2^N}$ [45], we can note that the last term in Eq. (4.7) is negligible, giving us

$$\mathcal{F}(n) \approx \mathcal{F}(n-1) f_n. \tag{4.8}$$

This recurrent relation can be in turn extended as follows

$$\mathcal{F}(n) \approx \prod_{i=1}^{n} f_i, \tag{4.9}$$

showing that the multi-qubit fidelity is approximated by the product of two-site operations. To check the precision of prediction given by Eq. (4.9) we launched simulations on three different circuits. All of them had the same architecture, as described above, but differed in the type of two-qubit gate used. By simply changing the type of this multi-qubit gate, we could easily increase the difficulty of the task. Thus, the circuits with *CNOT* gates were the easiest to simulate, *CZ* gates resulted in an intermediate difficulty, and the *ISWAP* gates were the most challenging ones, as would be expected from the way the quantum supremacy task was designed [104, 105]. In Figs. 4.2 and 4.3 we show the multi-qubit fidelity during each step of the execution of circuits containing 50 cycles, for a system consisting of 25 qubits. For each simulation we used two different bond-sizes $\chi = 256, 512$. It can be seen that the difficulty of the circuit clearly translates into the slope of the curves, and that increase in bond-size gives higher fidelity.

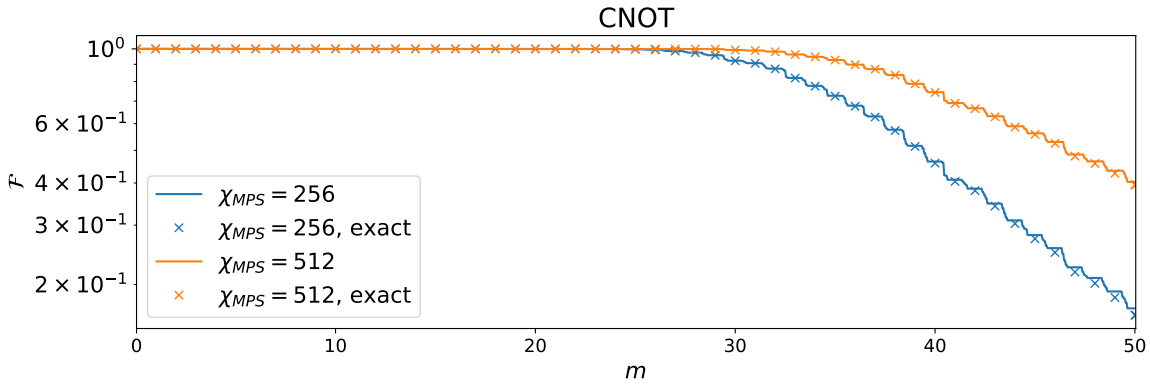


Figure 4.2: Multi-qubit fidelity of simulations of random quantum circuits with *CNOT* gates, for a chain consisting of 25 qubits, using the MPS-based method. The lines correspond to predictions given by Eq. (4.9), while the overlaps with exact calculations are marked with crosses.

We compared our results with an exact simulation obtained with the use of *cirq* library for relatively small system sizes. In Figs. 4.2 and 4.3 the marked overlaps between the state vectors given by *cirq* simulator, and the ones resulting from the contraction of the MPSs, are very close to the predictions given by Eq. (4.9), which proves the usefulness of this formula.

Besides the multi-qubit fidelity we would also like to define a second tool, with which we could evaluate the performance of our simulators compared to real quantum computers. We can achieve this by simply introducing the *average two-qubit fidelity* f_{avg}, which can be written as

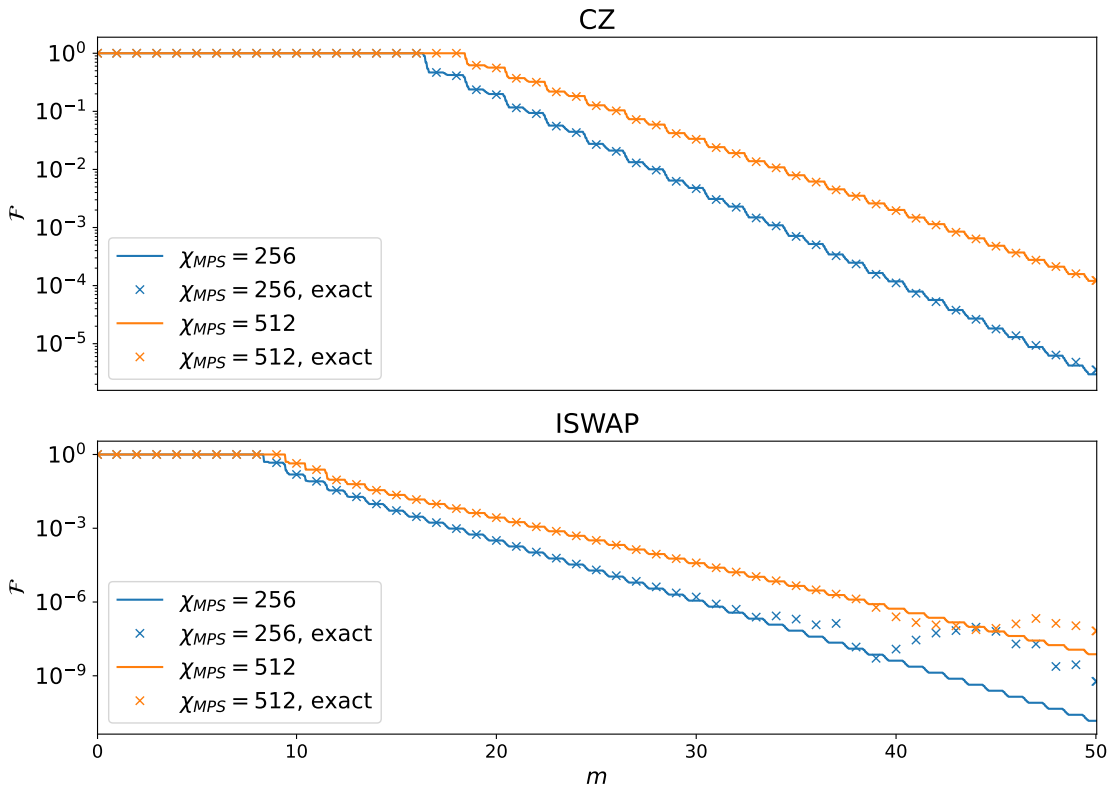


Figure 4.3: Multi-qubit fidelity of simulation of random quantum circuits with *CZ* and *ISWAP* gates, also for a chain of 25 qubits.

$$f_{avg} = \sqrt[n]{\mathcal{F}(n)}. \tag{4.10}$$

In Table 4.1 we show average fidelities for the same simulations as illustrated in Figs. 4.2 and 4.3.

χ_{MPS}	*CNOT*	*CZ*	*ISWAP*
256	0.997	0.979	0.959
512	0.998	0.985	0.969

Table 4.1: Average two-qubit fidelity for simulations of random quantum circuits applied on a chain of 25 qubits. Each circuit consisted of 50 cycles, which gives 600 two-qubit gates in total.

Considering the fact that for an exact simulation of these circuits with the MPS-based algorithm we would require the bond-size of 4096, we can still obtain an excellent average fidelity. Moreover, this precision should also increase when simulating circuits implementing real algorithms, as these are not designed to be extremely hard, but rather are targeted to solve actual problems.

4.2 Quantum supremacy task in 2D

In this section we will generalise the approach of Ref. [45] to the 2D case. Throughout all this thesis we will be considering only rectangular lattices, due to which the two-site gates appearing in any given layer can only occur in four possible configurations. Each such configuration corresponds to the type of pair of qubits, on which the gate can be applied. Thus, the four possible arrangements of bonds are: vertical odd (denoted as A), vertical even (B), horizontal even (C) and horizontal odd (D). As these configurations interspersed in different ways can result in tasks of varying difficulty, we will follow just the $ABCDCDAB$ pattern, which was reported to be the most challenging one [44]. In situations, in which there are more cycles in the circuit than in the presented sequence, this pattern is simply repeated as many times as necessary. In Fig. 4.4.a we show an example grid of qubits with marked types of bonds, on which the multi-qubit gates are applied.

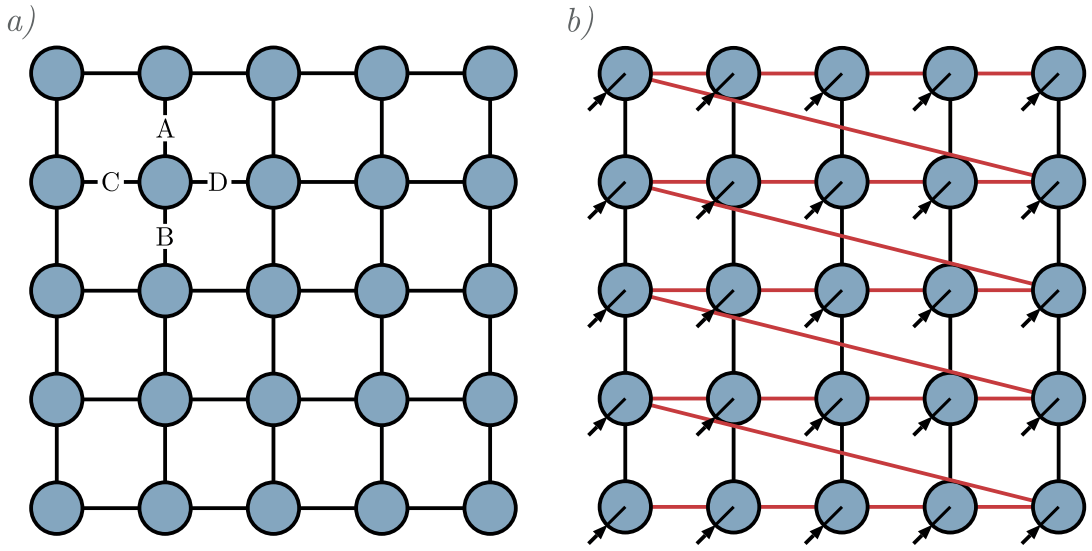


Figure 4.4: (a) Types of bonds, on which two-qubit gates are applied in a given layer. (b) Mapping of the 2D lattice onto a 1D MPS.

To obtain benchmark results of simulation, with which we will compare two-dimensional methods developed in the further part of this work, we conducted calculations with the use of MPSs. For that purpose we projected the rectangular lattice onto a 1D chain in a way shown in Fig. 4.4.b. With this mapping we were able to apply each layer of horizontal gates without any additional steps needed, while in the case of vertical ones we used a fixed number of *SWAP* operations per each gate. In spite of the fact that there have been proposed methods involving contractions of multiple sites at once [45], in our analysis we restricted ourselves to combinations of at most two tensors at any given moment, to make comparison of different methods easier.

Results of simulation of random circuits on a 5×5 lattice for $\chi = 256, 512$ are shown in Fig. 4.5. In this case we performed calculations for just 20 cycles. We can see that even though we used the same bond-sizes as in the case of 1D system, the fidelity drops significantly faster. This is caused by the need for the use of *SWAP* gates, which increase the difficulty of simulation. Moreover, by applying the vertical gates we are introducing long range correlations, which are hard to grasp with the use of 1D MPSs.

Similarly as in the case of simulation of a 1D system, in Table 4.2 we show the average two-qubit fidelity of each simulation of a circuit applied onto a 5×5 lattice.

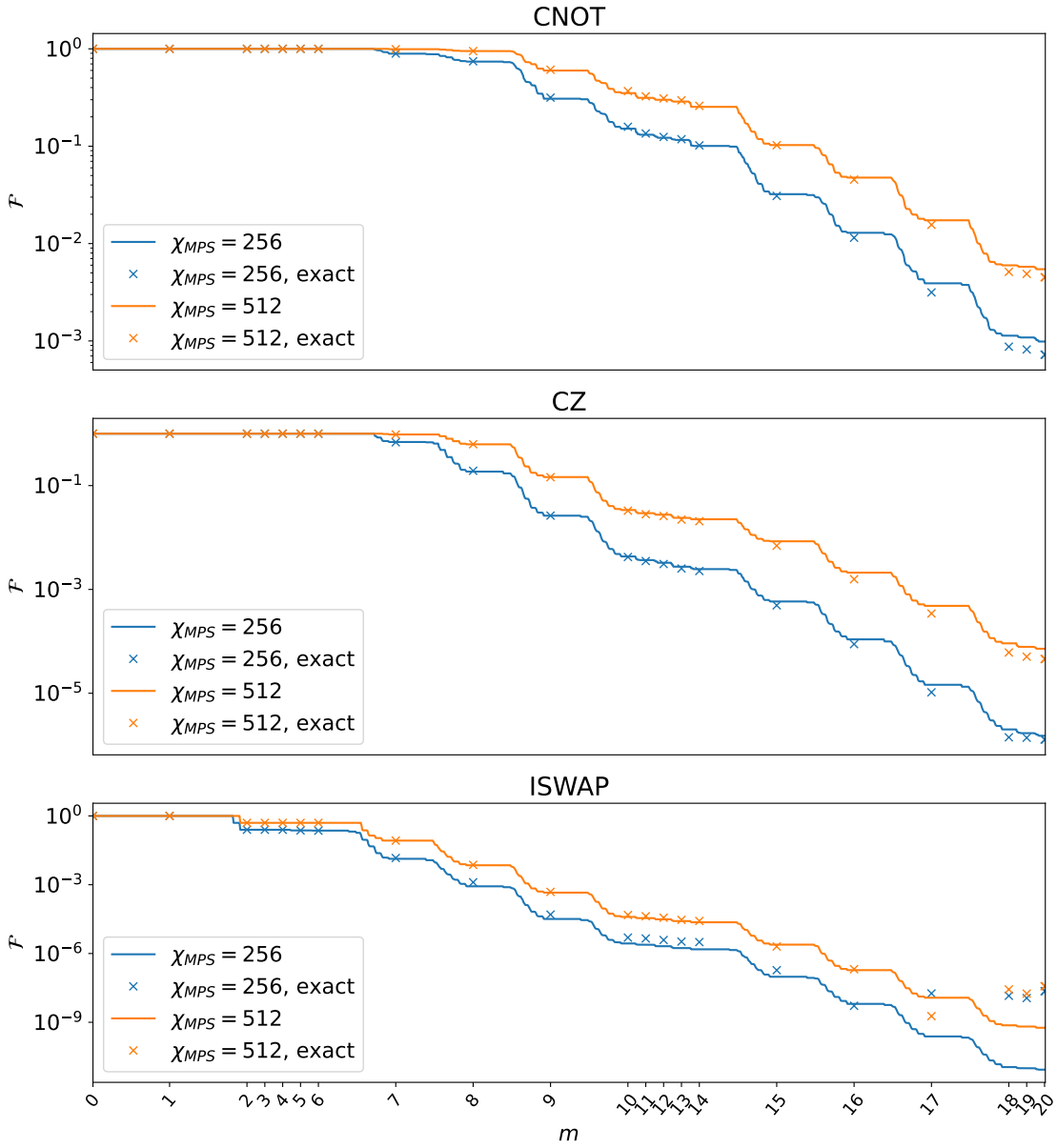


Figure 4.5: Multi-qubit fidelity of simulation of random quantum circuits with *CNOT*, *CZ* and *ISWAP* gates applied on a 5×5 lattice. The calculations were performed with the use of MPS-based method. From the spacing between different cycles on the horizontal axis it can be seen that the application of vertical layers requires significantly more operations than the horizontal ones.

χ_{MPS}	*CNOT*	*CZ*	*ISWAP*
256	0.965	0.935	0.88
512	0.974	0.953	0.899

Table 4.2: Average two-qubit fidelity for simulations of random quantum circuits applied on a 5×5 lattice. The total number of multi-site gates is equal to 200 (not including *SWAP* operations required by the MPS simulator).

FIFTH CHAPTER

ISOMETRIC TENSOR NETWORKS IN 2D

Up to this point we focused only on methods basing on single-dimensional tensor networks. However, other kinds of tensor networks were developed, which are operating in higher-dimensional spaces. Among them are *Tree Tensor Networks* (TTN) [106–113], *Multiscale Entanglement Renormalization Ansatz* (MERA) [114–116] and *Projected Entangled Pair States* (PEPS) [29, 30].

In TTNs the distance between any pair of sites in the lattice scales as $\mathcal{O}(\log N)$, where N is the total number of nodes. Because of that TTNs can capture longer-range correlations in comparison to the MPSs, as the correlation functions usually decay exponentially fast with respect to the path length [108]. An example TTN is shown in Fig. 5.1.

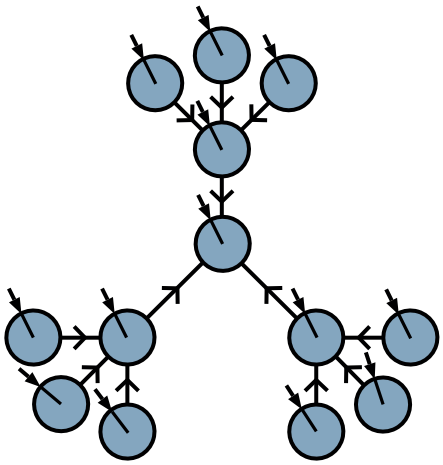


Figure 5.1: Example of a Tree Tensor Network.

PEPS are generalized MPSs, which are characterized by a regular lattice of higher rank tensors where nearest neighbours are connected by virtual legs. Usually, the arrangement of these legs matches the geometry of the lattice under study. The power of PEPS lies in the ability of capturing the ground states of 2D systems with significantly smaller bond-size than the one needed when using MPS. However, the cost of *exact* calculation of properties of such a tensor network grows exponentially, because of which usually approximation methods are used for that purpose.

Recently different methods of *canonization* of PEPS were presented [31–34], which by imposing the isometry conditions onto the tensor network reduces its cost of contraction. In this work we will follow the methods introduced in Refs. [32, 34] and refer to a canonized PEPS as an *Isometric Tensor Network* (isoTNS). In Fig. 5.2.a we present an example of a PEPS, while Fig. 5.2.b illustrates its isoTNS version.

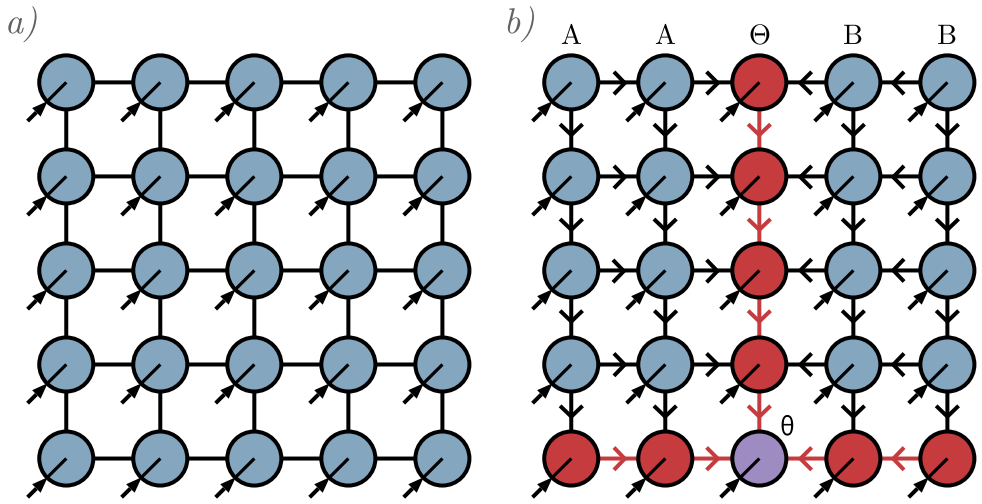


Figure 5.2: (a) PEPS for a 5×5 lattice. (b) isoTNS for the same lattice. Orthogonality row and column Θ are highlighted in red, while the orthogonality center θ is marked in purple.

As can be seen in Fig. 5.2, in the case of isoTNSs we need to extend the concept of orthogonality center to orthogonality row and column. These two surfaces can be easily recognized while looking at the illustration, by having only incoming bonds connecting them with the rest of the lattice. Moreover, the position of orthogonality center inside the orthogonality surface is not fixed, as it can be easily moved with the use of SVD. However, shifting of the orthogonality center outside its current orthogonality surface is not possible, and instead of that the whole orthogonality row (or column) is moved. This procedure will be discussed in detail in Section 5.3.

5.1 Initialization of the isoTNS

isoTNS can be initialized in a number of different ways. One of them can be achieved by the so called *MPS to isoTNS* algorithm [32], which transforms an MPS with each constituent tensor being a grouped site, containing multiple physical indices, into a full fledged isoTNS with just a single tensor per each node. However, in this work we will not need this procedure, so we will use for the purpose of initialization much simpler approach. Specifically, we will generate random, normalized tensors with physical leg of size d and the remaining virtual bonds of size 1, whenever a random starting state from a given algorithm is required. Thus, this will be the strategy used in the case of imaginary time evolution.

In the case of simulating quantum computations we will simply create the product state $|00...0\rangle$ (which usually serves as a starting point to all quantum algorithms) by generating separately $|0\rangle$ states on each site, and reshaping them to match the dimensionality required by the isoTNS. So, each vector will be also transformed into a tensor with physical leg of size d and remaining virtual bonds of size 1. isoTNS created in this way has just one singular value per each virtual bond (equal to 1), thanks to which it fulfills the isometry condition.

5.2 Computing observables of an isoTNS

Computation of observables with the use of an isoTNS is particularly efficient, especially in the case of single-site expectation values, since it amounts to contracting the orthogonality center with its Hermitian adjoint and an operator. When it comes to calculating two-site observables, for nearest neighboring sites it can be also be achieved at a low cost, because we only need to make sure that the two nodes in question are lying inside the orthogonality surface, and can be simply contracted. The rest of the computations is analogous to the case of single-site operators [28]. However, for sites farther apart it is necessary to contract larger number of nodes, similarly to the case of an MPS when calculating long range correlations [28].

5.3 Shifting of the orthogonality surface via Moses Move

In order to be able to optimize any given tensor in the lattice we need to introduce a procedure allowing for shifting of the orthogonality surface. This method will be called the *Moses Move* [32]. We will exemplify its use on the case of orthogonality column, however shifting of orthogonality row is analogous.

As mentioned above, the input to the Moses Move consists of the lth orthogonality column, which will be denoted as Θ^l, with the orthogonality center θ stored at its bottom tensor. The algorithm will output two new columns: A^l and Θ. The first one will be the new lth column, fulfilling the isometry condition, while Θ can be thought of as an extracted ancilla column, not directly assigned to any physical position in the lattice. Its further contraction with the B^{l+1} column will result in a new orthogonality column Θ^{l+1} similarly as it takes place in MPS, when we contract singular values with given tensor to shift the orthogonality center.

We begin the procedure by shifting θ to the top node by conducting a series of SVDs. For each tensor to by factorized we firstly combine all of its legs, besides the virtual one attached from the direction in which we would like to move θ. Assuming that the upper bound on the bond-size for isometries is equal to χ, and for the orthogonality surface η, reshaping of the tensor located at the bottom of the lattice gives a matrix of size $d\eta^2 \times \eta$, while for the tensor located in the bulk this transformation results in a $d\eta\chi^2 \times \eta$ one.

After reaching the top of the lattice we begin the core part of the Moses Move. At its each step, we will split one tensor into three new ones. This splitting is preceded by combining of tensors legs into three groups. By the A group we will denote legs attached from the bottom, which means that in fact this group will consist only of the bottom, virtual leg. The B group will contain all bonds attached from left and also the physical one. Finally, the C group will incorporate all remaining virtual legs attached from right.

After the first reshaping is finished we conduct another one, by combining the A and C legs. As a result we obtain a matrix of size $\dim(B) \times \dim(AC)$. This matrix is factorized via SVD, which gives U_1, S_1 and $V_1^\dagger$ matrices. After the incorporation of singular values into $V^\dagger$, we are left with U_1 and θ_1 matrices connected by a single bond. This bond is splitted into two new ones, each of size at most χ, the product of which gives the original bond-size. So, for instance, if the original matrices U_1 and θ_1 are of shapes 4×4 and 4×8, respectively, the connecting leg of size 4 is divided into two, each of size 2. These new legs will be denoted as B_R and B_L. The splitting is achieved by reshaping the matrices into order-3 tensors.

Subsequently we are reshaping θ_1 from a $\dim(B_R) \times \dim(B_L) \times \dim(AC)$ tensor into an order-4 one of size $\dim(A) \times \dim(B_R) \times \dim(B_L) \times \dim(C)$. It is done in order to find an unitary

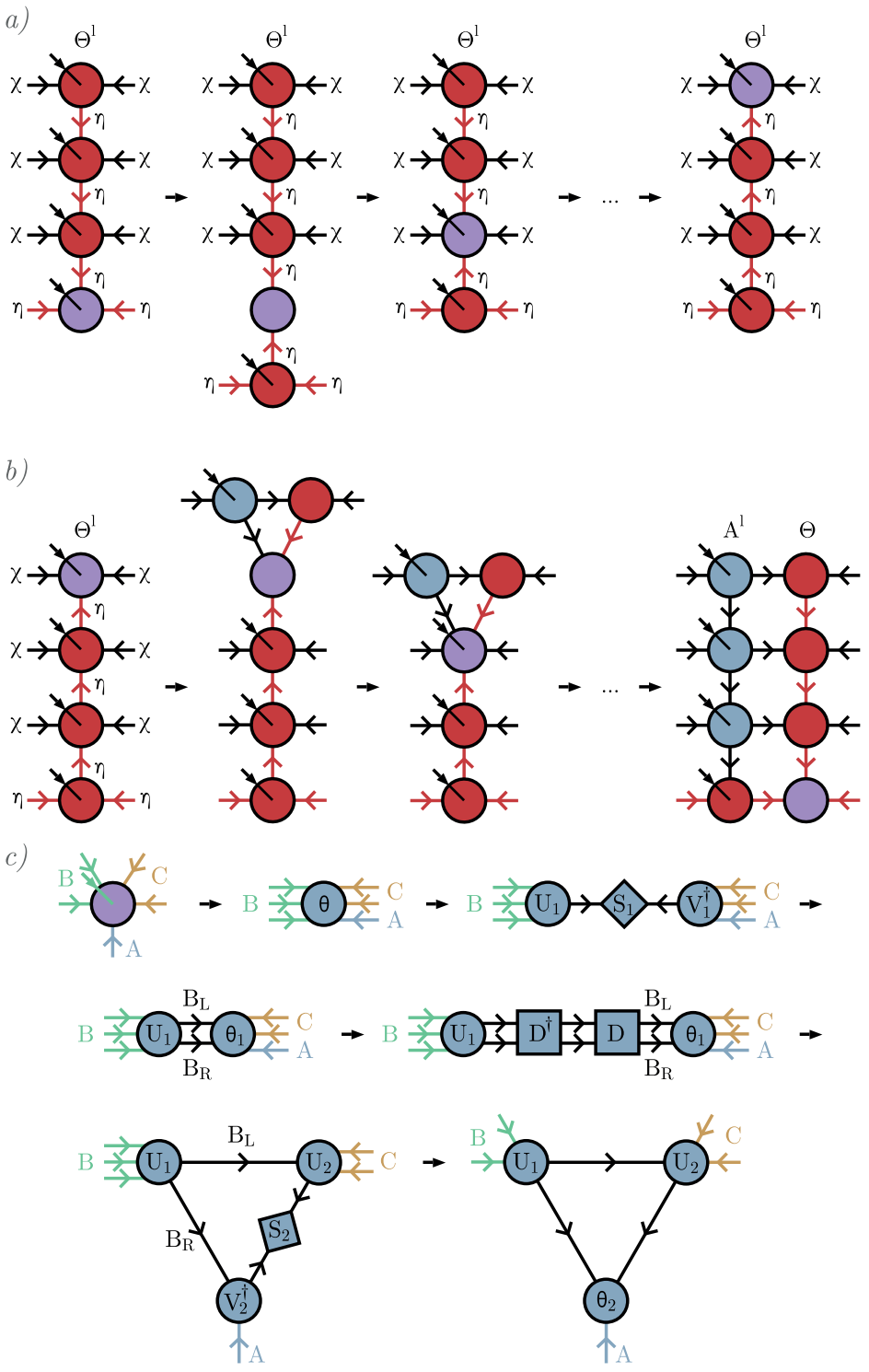


Figure 5.3: a) Shifting of the orthogonality center to the top of the lattice. b) A conceptual view of the Moses Move. c) Detailed splitting procedure dividing a single tensor into three new ones.

disentangler D minimizing the entanglement entropy $S_{AB_R:B_LC}$, thus the one between the AB_R and B_LC degrees of freedom. Finding of such disentangler is a well known problem, which might be solved with the use of conjugate gradient method [34, 117].

When D is found, it is contracted with θ_1 over the B_R and B_L legs, while its Hermitian

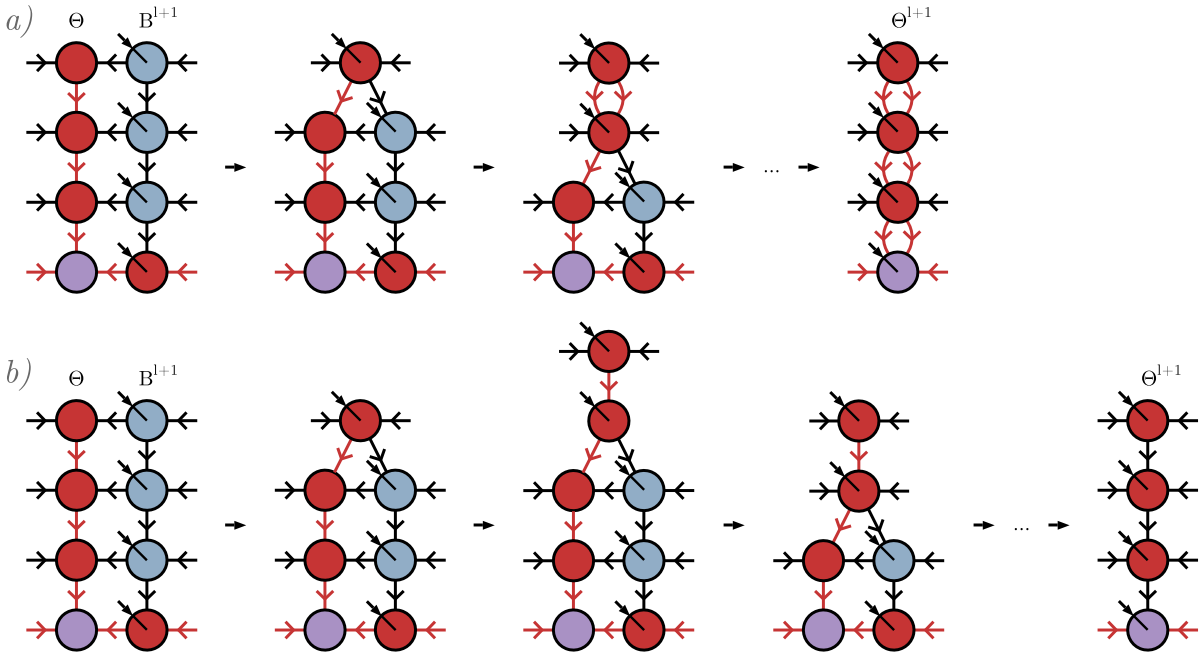


Figure 5.4: a) Basic merging of Θ and B^l columns. b) Contracting of columns involving an immediate truncation of vertical bonds.

adjoint $D^\dagger$ is contracted with the U_1 tensor. The following step of the procedure consists of reshaping of θ_1 tensor into a matrix of size $\dim(AB_R) \times \dim(B_L C)$, which is further factorized by SVD into U_2, S_2 and $V_2^\dagger$. If the disentangler was not used previously, this step could result in a large number of non-negligible singular values, the disposal of which would give a substantial truncation error.

By incorporating S_2 into $V_2^\dagger$ we obtain the new orthogonality center θ_2. The last step of tensor division involves splitting of all legs of U_1, U_2 and θ_2, which were previously combined. After this procedure is finished, θ_2 might be contracted with the tensor lying below the initial tensor that was divided by the Moses Move. When this is done, the whole algorithm might be repeated. By conducting it on each tensor of the initial Θ^l column we are "unzipping" it, and obtain new A^l and Θ columns. All of the steps described above are shown in Fig. 5.3.

After the Θ column has been extracted, it can be further contracted with the B^{l+1} column located to the right. It can be achieved in two ways. Firstly, tensors in each row of Θ can be simply contracted with the corresponding ones from B^{l+1}. This would result in a new column Θ^{l+1}, in which each tensor would have double vertical legs. These can be truncated later during the following update of the Θ^{l+1} column, e.g., while shifting the orthogonality center upwards.

A second way to merge these two columns involves an immediate truncation of vertical bonds. After the first contraction of tensors from the Θ and B^l columns is finished, the resulting tensor is reshaped into a matrix by grouping its horizontal legs with the physical bond on one side, and the vertical legs on the other. The factorization that follows resembles shifting of the orthogonality center upwards, shown in Fig. 5.3.a, however this time we are proceeding downwards. While conducting the SVD the vertical bond can be truncated and the singular values are incorporated into the bottom tensor. During the next step of merging, instead of contracting just two tensors, we are also contracting the third one, being the result of the procedure described above. The following steps are analogous to the first one, however during reshaping of tensor into a matrix, the top vertical legs are also included into the group

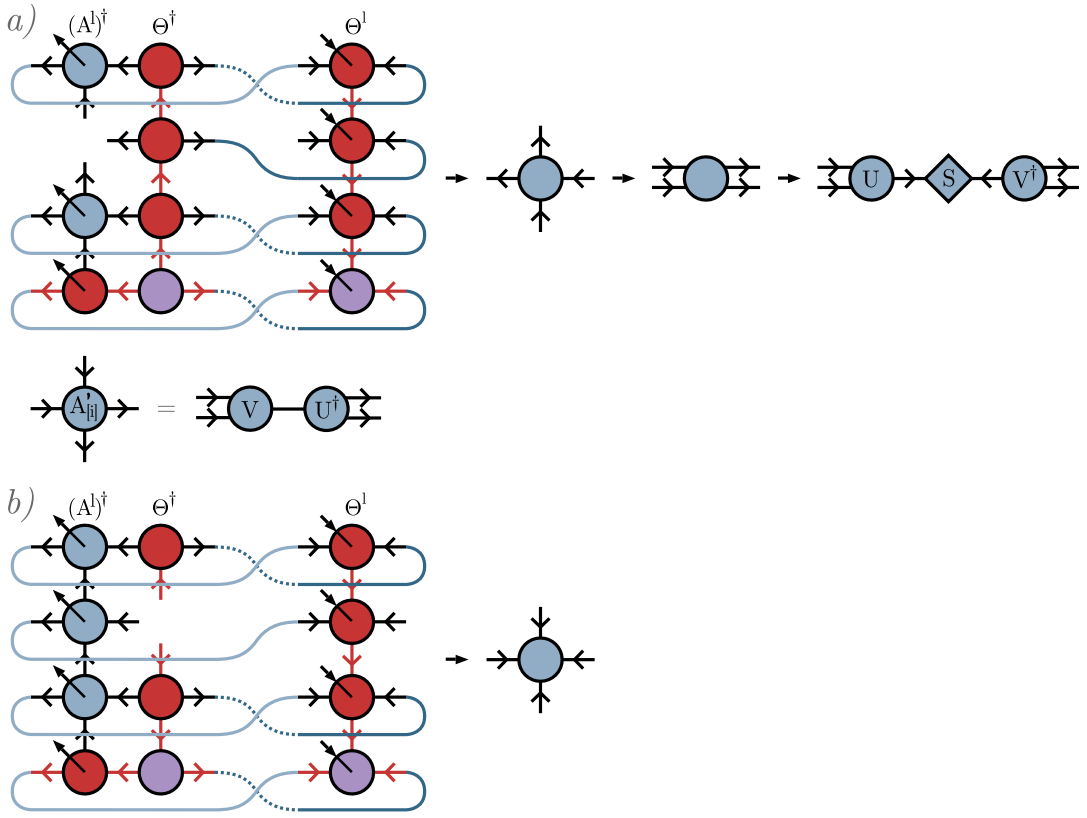


Figure 5.5: a) Update of the A^l column by the Evenbly-Vidal algorithm. b) Optimization of the Θ column.

consisting of horizontal and physical ones. Both of the column contracting techniques are depicted in Fig. 5.4.

Although the Moses Move presented in this section was moving rightwards, a leftwards shifting is analogous. Moreover, the application of Moses Move on rows can be achieved by simply rotating the whole lattice and using the same procedure.

5.4 Variational solution of $\Theta^l = A^l\Theta$

Although result given by the Moses Move might not be the optimal one, it can be improved by means of variational optimization techniques. Firstly, we can increase the overlap $\langle A^l\Theta|\Theta^l\rangle$ by updating the tensors of the A^l column. This can be achieved via the Evenbly-Vidal algorithm [118, 119]. It is based on the idea of linearization of function to be optimized by keeping all tensors fixed, except the one being updated during given iteration. In our case, the function in question is the aforementioned overlap $\langle A^l\Theta|\Theta^l\rangle$. We can denote the updated tensor as $A_{[i]}$ and the rest of the tensor network, forming the environment of this tensor, as $E_{A_{[i]}}$. Then, by sweeping through the A^l column we are updating each of its tensors as $A_{[i]} \leftarrow A'_{[i]} = VU^\dagger$, where the V and $U^\dagger$ matrices come from the factorization of the environment through SVD $E_{A_{[i]}} = USV^\dagger$.

The optimization of the Θ column can be thought of as variational compression, which was described in detail in Section 2.5.2. The only difference lies in creation of environment of the tensor being updated. In this case, it will be similar to the one used in the Evenbly-Vidal algorithm, but a different tensor is removed from the tensor network.

Both of these optimization techniques are depicted in Fig. 5.5.

5.5 Imaginary time evolution in two dimensions

With the procedures outlined in the preceding sections, we can now easily introduce the algorithm of imaginary time evolution making use of the isoTNS. In fact, it will be almost identical to the Moses Move, with the only difference lying in the initial shifting of the orthogonality center. In the Moses Move we achieved this by the mere use of SVDs. However, these factorizations can be replaced with the application of evolution operators, followed by the calculation of expectation values of bond operators.

As a result, the application of this method on a given column is similar to a 1D TEBD algorithm. After sweeping through each column we can rotate the whole lattice and proceed with the following updates, this time over the rows. After four rotations we obtain a lattice in its original orientation, which finishes one full step of the algorithm. Summarizing, this method consists of two nested versions of a single-dimensional TEBD, where it got its name *TEBD*^{2}.

The application of evolution operators on a single column is depicted in Fig. 5.6.a, while the conceptual representation of the whole TEBD^2 algorithm is shown in Fig. 5.6.b.

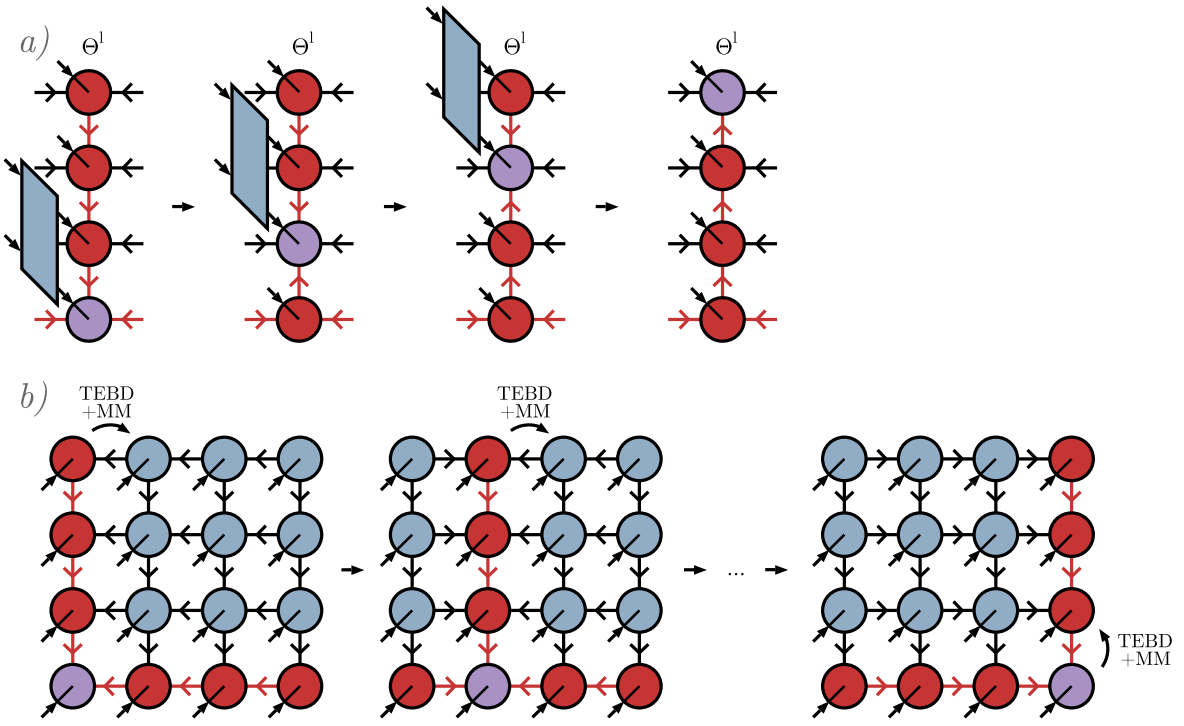


Figure 5.6: a) Update of a single column of the isoTNS in a TEBD-like fashion. b) A schematic outlook of the TEBD^2 algorithm.

To estimate the cost of the algorithm we firstly need to make certain assumptions regarding the size of η. As a matter of principle we will simply use $\eta = 6\chi$, as it gave the best numerical results. With this setup we can note that the cost of both SVDs conducted during the Moses Move is of the same order. The first one takes $\mathcal{O}(d\eta^2\chi^3 \cdot d\chi^2) = \mathcal{O}(d^2\eta^2\chi^5)$ FLOPS, while the second one $\mathcal{O}(\eta^2\chi^3 \cdot \eta\chi) = \mathcal{O}(\eta^3\chi^4)$. As η is a simple linear function of χ we can immediately see that the cost of both factorizations is proportional to χ^7 [32,34] . The following

contraction of the orthogonality center (resulting from the division of the initial tensor into three new ones), with the tensor lying below, costs $\mathcal{O}(d\eta^3\chi^3) < \mathcal{O}(\chi^7)$.

While merging Θ column with the B^{l+1} one, the most costly contraction is carried out in the bulk, and takes $\mathcal{O}(d\eta^2\chi^5) \propto \mathcal{O}(\chi^7)$ FLOPS. Thus, the cost of the entire procedure shifting the orthogonality surface is $\mathcal{O}(\chi^7)$.

Whereas, in the 1D-TEBD method the most expensive operation is the division of the two-site tensor into two separate ones, and its cost is $\mathcal{O}(d^2\eta^2\chi^4 \cdot d\eta\chi^2) = \mathcal{O}(d^3\eta^3\chi^6) \propto \mathcal{O}(\chi^9)$ [34].

5.6 Benchmark results of the TEBD^2 algorithm

We used the TEBD^2 algorithm trying to reproduce the results obtained in Section 3.1, i.e., the ground states of the TFI model on a square lattice with approximately the same maximal entanglement entropy for varying system sizes. As the convergence of the algorithm over the consecutive iterations was very slow, we followed a similar approach as the one used in Ref. [32]. Instead of running the algorithm for a given time-step τ until reaching convergence, as it is usually the case when using the TEBD algorithm, we rather conducted a fixed number of iterations per given τ, followed by the decrease of the length of τ. The number of iterations was inversely proportional to τ. In our calculations we used a second-order Trotter-Suzuki decomposition to minimize the energy error. The obtained results are shown in Fig. 5.7.

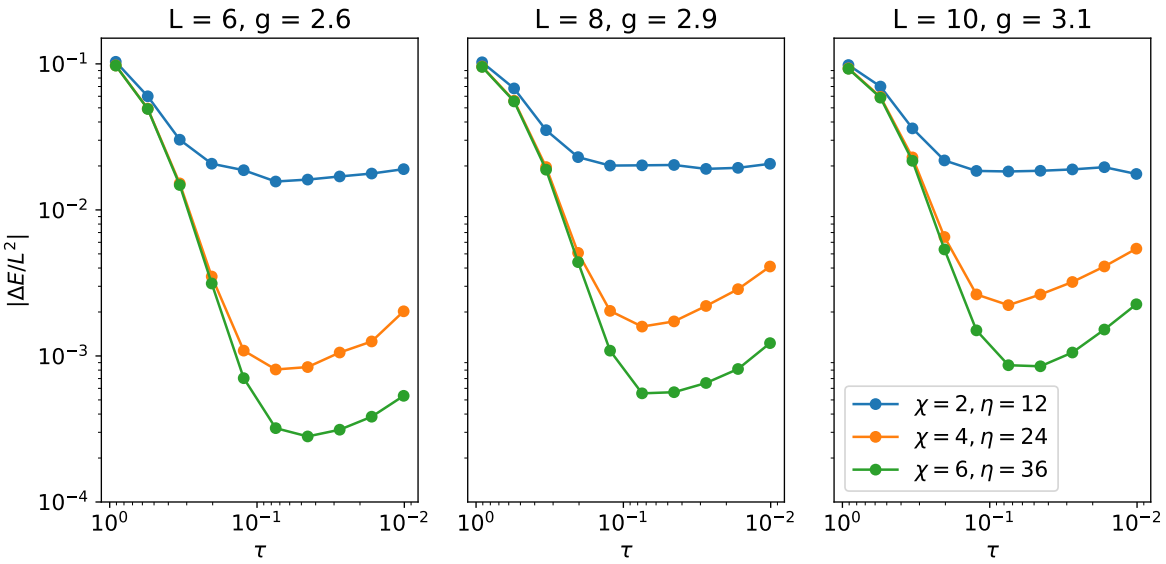


Figure 5.7: Errors of mean energies per site for the TFI model. The true ground states was obtained by large scale 1D-DMRG calculations.

From Fig. 5.7 we can see that after reaching its minimum, the error of energy density starts to diverge. This is caused by the use of the Moses Move, which introduces small inaccuracies during its each iteration.

5.7 Shifting the entire orthogonality surface into the bulk

Up until this point we were introducing the procedures firstly presented in Refs. [32, 34]. From now on we will show our own modifications of these methods, created for the purpose of this work.

The first alteration allows for shifting of the whole orthogonality surface into the bulk, which allows for a significant reduction of bond-sizes needed while carrying out numerical calculations. It should be emphasized that in the initial definition of the Moses Move and TEBD^2 one of the orthogonality surfaces was kept at the edge of the lattice, while the other one was moved across the system. For instance, throughout all examples shown in previous sections we were always keeping the orthogonality row at the bottom edge of the lattice, while the orthogonality column was shifted.

To see the consequences of this constraint imposed on the lattice let us use a certain analogy. We can view the graph representing the system as an instance of the *flow network* [120] in which all of the nodes are sources, with the only exception being the orthogonality center, which acts as a sink. Then, the flow going through the lattice corresponds to the maximal possible entanglement. Moreover, for each node (besides the orthogonality center), the product of sizes of the incoming legs must be equal to the product of sizes of the outgoing ones. If we concentrate for a moment solely on the virtual legs, we can see that their outgoing flow must be d times larger than the incoming one, where this d factor comes from the temporarily ignored physical bond.

Using this formalism we can easily estimate the maximal bond-size of 4096 needed to represent exactly a 5×5 lattice of qubits, with the orthogonality surface located on the edge. This system is depicted in Fig. 5.8.a. However, if we were able to shift the whole orthogonality surface into the bulk, we would be able to reduce the bond-size required to just 64. This situation is illustrated in Fig. 5.8.b. We should immediately note that the largest tensors in both of the lattices in Fig. 5.8 store the same number of variational parameters. However, the above strategy for estimating bond-sizes assumes that we are in possession of huge computational resources and do not perform any truncations. When executing real algorithms, we will always discard some singular values to avoid an exponential explosion of needed memory. Then, in the scenario shown in Fig. 5.8.b the pruning of variational parameters is much more flexible, and potential SVDs should cost much less than in the case shown in Fig. 5.8.a.

To be able to shift the entire orthogonality surface into the bulk we can firstly note that the lattice shown in Fig. 5.8.a can be interpreted as a top-left part of a larger system. The virtual legs connecting this subsystem with its environment would be the incoming ones, and all of them would be of size 1. From this perspective, the TEBD^2 launched on this lattice can be viewed as time evolution of just a single part of a larger system, completely independent of its environment.

By extending this reasoning we can modify the original TEBD^2 algorithm to operate on four different parts of the lattice separately. The only modification needed is to take into account the dangling legs when approaching the edge of the sublattice, which connects it with the remaining part of the system. With this adjustment we can run TEBD^2 in a clockwise fashion, sequentially going through all four quarters of the lattice. This procedure is shown in Fig. 5.9.

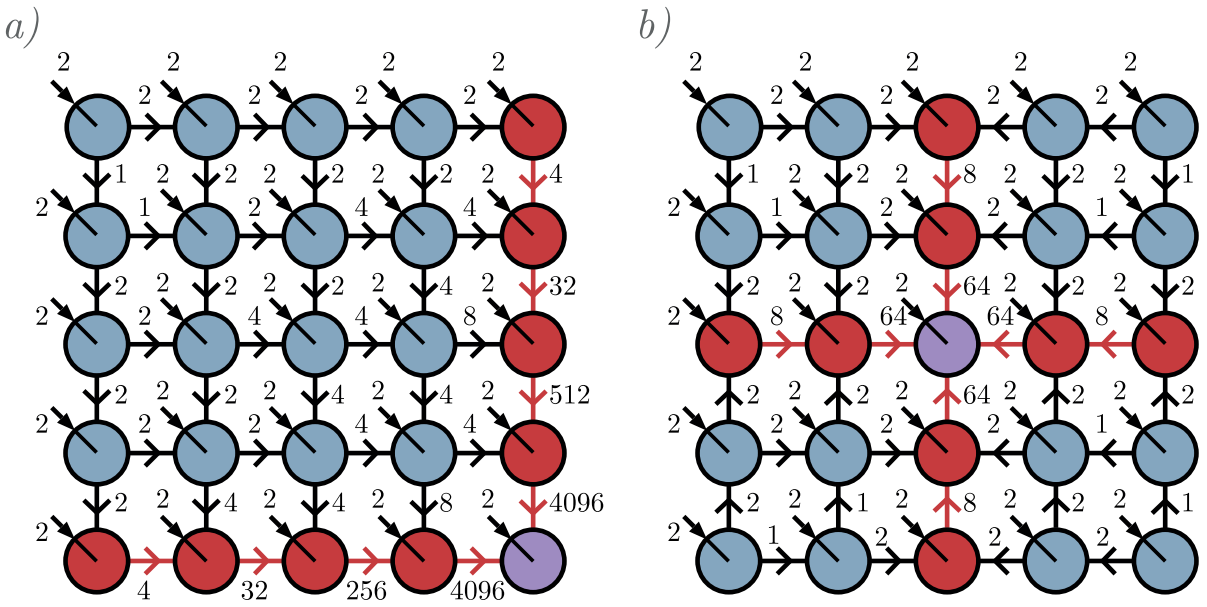


Figure 5.8: Comparison of different bond-sizes needed to represent exactly a 5×5 lattice of qubits with different locations of the orthogonality surface.

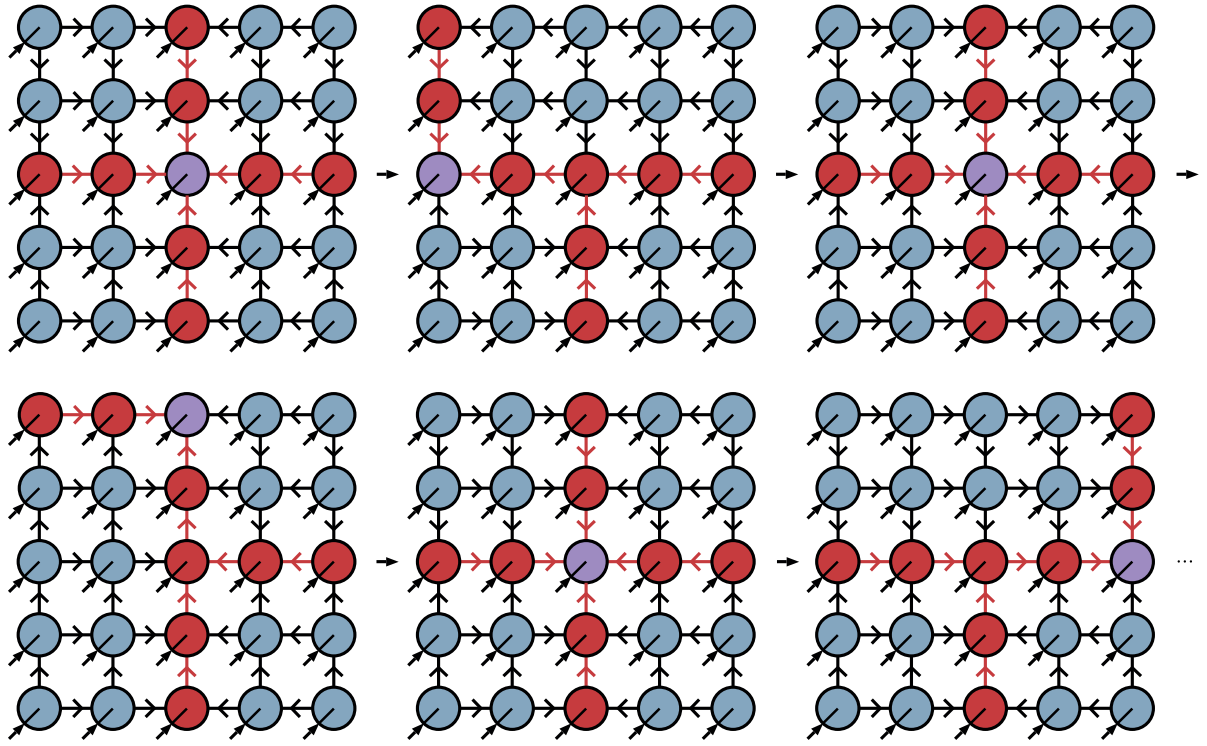


Figure 5.9: TEBD^2 algorithm with the orthogonality surface located in the bulk.

5.8 Tree-like isometric tensor networks

Since we wanted to use our algorithms operating on isoTNSs to simulate computations on quantum computers, we tried to predict the multi-qubit fidelity in a way analogous to that shown in Eq. (4.9). Unfortunately, as the Moses Move introduces additional inaccuracies, not directly related to the errors attributed to the application of operators, we were not able to approximate this value with high precision (neither by using SVDs, nor by means of

variational solution of the $\Theta^l = A^l\Theta$ problem).

This prompted us to slightly modify the structure of the lattice. We decided to keep the sizes of chosen legs to be always equal to 1, effectively forbidding the flow of entanglement in loops between any two nodes in the system. As a result, we obtained a tree-like version of the isoTNS, with "empty" bonds connecting the branches. Also, each path connecting any node with the central site has an "L" shape. Finally, this choice of the lattice arrangement allowed us to pick just a single restriction of the bond-size, being equal to χ, which allows to avoid the ambiguity associated with selecting the correct relationship between χ and η values, as was the case in the initial definition of the isoTNS. Example of a tree-like isoTNS is illustrated in Fig. 5.10.

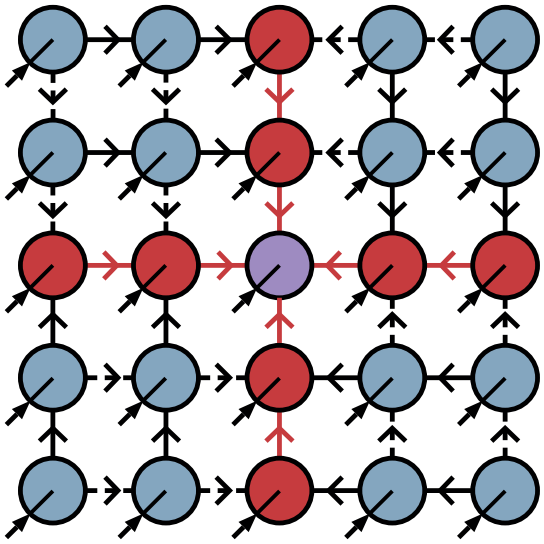


Figure 5.10: Structure of the tree-like isoTNS on a 5×5 lattice. The dashed lines correspond to bonds of size 1.

By choosing this structure of the lattice we were able to apply operators on non-empty bonds, and further move the orthogonality surface with the use of the Moses Move. This time, the predicted fidelities were accurate, as the Θ surface extracted in the process of shifting consisted of tensors with just a single entry, enforcing it to be equal to 1. As a result, further contraction of Θ tensors with the ones stored in the B^{l+1} surface was trivial, and did not disturb the precision of our predictions. Moreover, we were able to stop using the disentangler as a result of the introduction of these modifications, as one of the legs of the D unitary would be of size 1, which makes the optimization redundant. The application of the Moses Move on a modified lattice is presented in Fig. 5.11.

However, the above method does not allow to update the "empty" bonds. To solve this issue we developed a new technique, involving the contraction of tensors on neighboring branches. We will exemplify it on the case of application of operators on vertical bonds in the top-left quarter of the system. Beginning in the centre of the lattice, we are contracting two sites into one, effectively starting the merging of two branches. After having applied an operator on the resulting two-site tensor, we shift the orthogonality center by SVD in the outwards direction. We decided to loosen restrictions imposed on the bond-size for this step of the algorithm and keep at most χ^2 singular values, after the factorization is finished. We do that in order to not dispose of too much entanglement stored in two branches, which at this point are being fused into one. We further contract the ancilla tensor (the one without

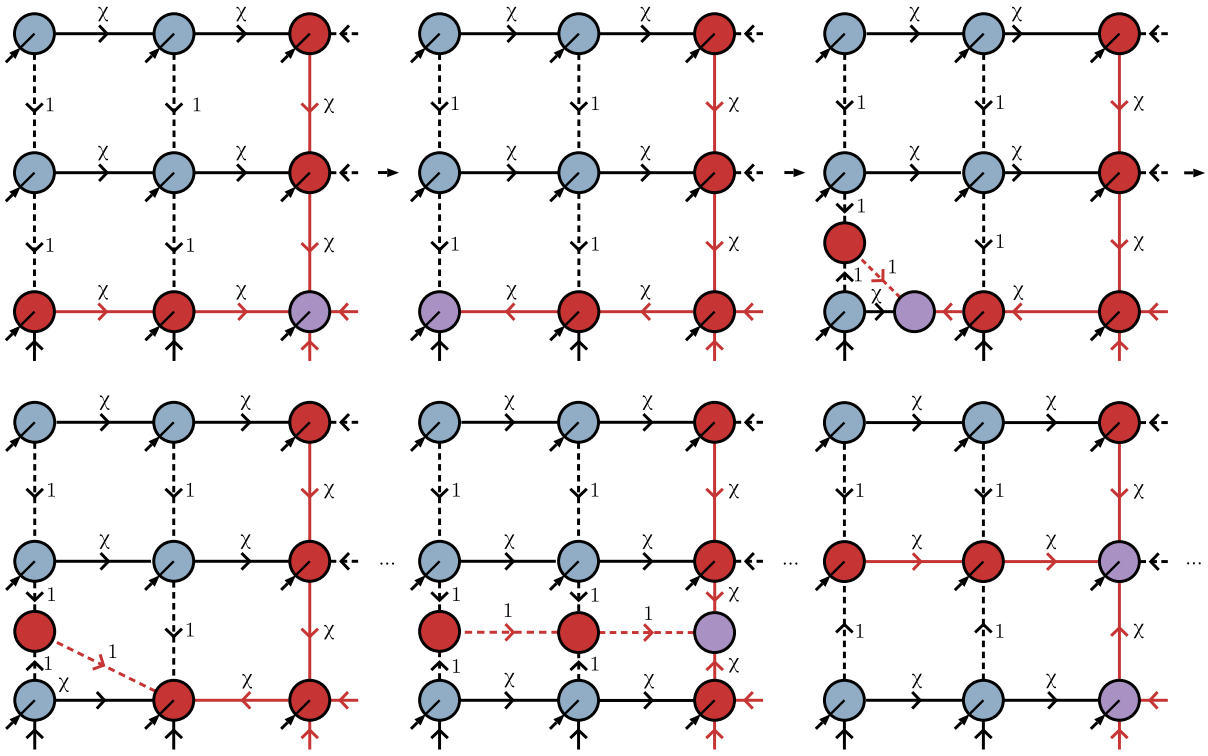


Figure 5.11: The Moses Move procedure conducted on a tree-like isoTNS.

any physical leg) with the following two tensors lying on its path. We proceed in this fashion until reaching the leaf nodes.

At this point we start the division of branches, combined with a simultaneous shifting of the orthogonality surface. Using SVD we firstly "detach" the bottom site, and further reshape resulting tensors in order to split in two the bond connecting them. As the vertical bonds in this example were chosen to be always of size 1 (besides the central column of the whole lattice), the reshaping step can be simply achieved by an addition of a trivial leg to both of the tensors. Then, we shift the orthogonality center inwards, by obtaining with SVD an ancilla tensor, which is further contracted with a two-site one located closer to the center of the system. We sweep in this fashion inwards, until reaching the two-site tensor from which the merging of branches began, and simply divide it by SVD, which finishes one step of shifting of the orthogonality surface. All steps involved in this procedure are illustrated in Fig. 5.12.

By repeating the outward and inward sweeps multiple times we can apply operators on all vertical bonds, concurrently shifting the orthogonality surface to the top-edge of the lattice. We then proceed with an analogous method to move it back to the center of the system, which allows for subsequent update of the remaining parts of the lattice.

We will conclude this section with the analysis of the computational complexity of techniques being performed on the tree-like isoTNS. For both the Moses Move and the two-site update the most expensive operations are being conducted on the site located in the center of the lattice, due to the highest number of non-empty bonds attached to it. In the case of the Moses Move, the cost of factorization of this site is $\mathcal{O}(d\chi^4 \cdot \chi) = \mathcal{O}(d\chi^5)$, while for the two-site update it is equal to $\mathcal{O}(d^2\chi^5 \cdot d\chi^2) = \mathcal{O}(d^3\chi^7)$. Moreover, the contraction of the central site of the lattice with an orthogonality center, resulting from the chosen shifting technique, is of the same order as its division via SVD, and is equal to $\mathcal{O}(d\chi^5)$ and $\mathcal{O}(d^2\chi^7)$, respectively for the Moses Move and the two-site method. Finally, the most expensive operation associated with the application of an operator is the factorization of the updated two-site tensor, and

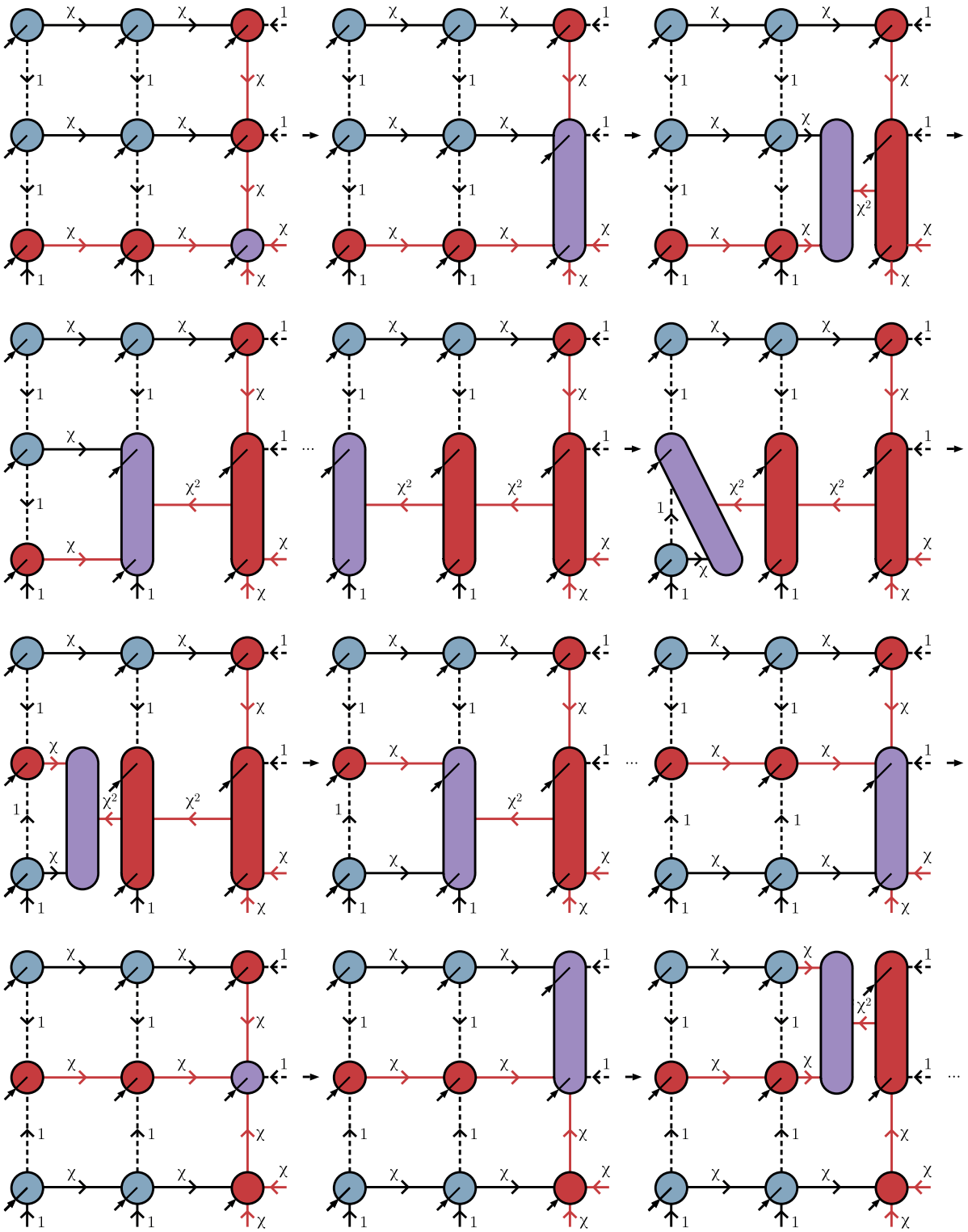


Figure 5.12: A two-site method allowing for simultaneous application of operators on "empty" bonds and shifting of the orthogonality surface. Application of two-site operators was omitted for clarity.

its cost is equal to $\mathcal{O}(d^2\chi^5 \cdot d\chi^2) = \mathcal{O}(d^3\chi^7)$. We can thus see that the cost of shifting of the orthogonality center across the lattice in the tree-like isoTNS is of the same order as in the case of the Moses Move performed on the basic isoTNS ($\propto \mathcal{O}(\chi^7)$), while the number of FLOPS needed during the application of an operator on a two-site tensor is of smaller order

in the tree-like version (also $\propto \mathcal{O}(\chi^7)$).

5.9 Benchmark results for the tree-like isoTNS

To benchmark performance of the TEBD^2 algorithm using the tree-like isoTNS we return to the task of obtaining the ground state of the TFI model. This time we also used a second order Trotter-Suzuki decomposition while generating the time evolution operators. However, because during the update of a single quarter of the lattice we are firstly sweeping outwards, and then inwards, we divided each evolution step over time τ into two separate ones, each corresponding to a time-step of length $\tau/2$. Consequently, the first and last layers of operators applied in every iteration of the TEBD^2, which originally corresponded to a $\tau/2$ time-step, were replaced with two layers evolving the system over the time $\tau/4$.

With the use of modifications introduced in previous two sections we observed that the convergence was much more stable than in the case of the original TEBD^2 algorithm. Because of that we were reducing the time-step length only when the energy converged for the currently used value of τ, similarly as it is done in the case for classical TEBD.

In Fig. 5.13 we show energy calculated with the modified TEBD^2 as a function of the time-step τ. As a reference point we used the ground state energies obtained in Section 3.1 with the use of 1D-DMRG.

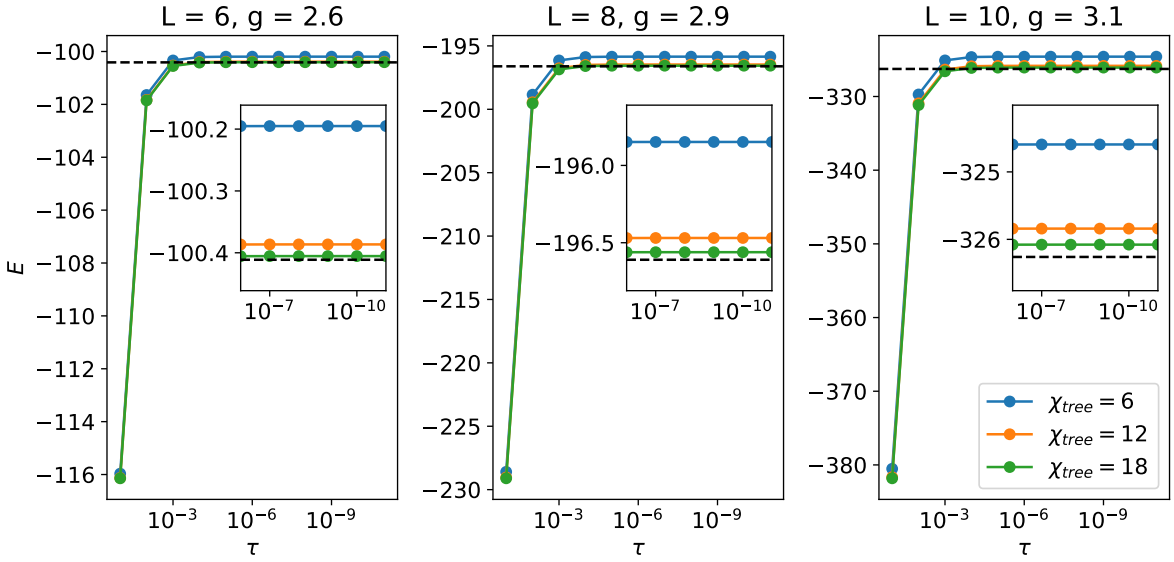


Figure 5.13: Ground state energies of the TFI model on a $L \times L$ lattice, obtained with the TEBD^2 algorithm operating on a tree-like isoTNS, per time-step τ. Each dot corresponds to the last value of energy calculated using given τ. Insets depict the energies for very small values of τ. The bond-sizes chosen for the purpose of simulations were equal to 6, 12 and 18.

We also wanted to compare the precision with which the true ground state can be represented using the tree-like isoTNS and an MPS, when we limit the number of variational parameters stored in each of these structures. For that purpose we conducted multiple simulations using the modified TEBD^2, progressively increasing the maximally allowed bond-size. On the other hand, we conducted large scale 1D-DMRG calculations to obtain the ground state in the form of an MPS, which was subsequently gradually compressed. The details on the latter approach were presented in Section 3.1.

In Fig. 5.14 we show results of error densities of the energy as a function of the number of used variational parameters, for the two techniques described above. We can see that a certain gain from the usage of the tree-like isoTNS was observed only for a very small number of used parameters. However, it is known that imaginary time evolution manifests worse convergence than the DMRG algorithm, which can partially explain this phenomenon. A similar analysis with the use of a DMRG approach, which was already proposed in the context of isoTNSs [34], could be very fruitful and would allow to definitively determine, whether a similar gain in precision of representation of the true ground state can be achieved using the tree-like isoTNS for large number of used variational parameters.

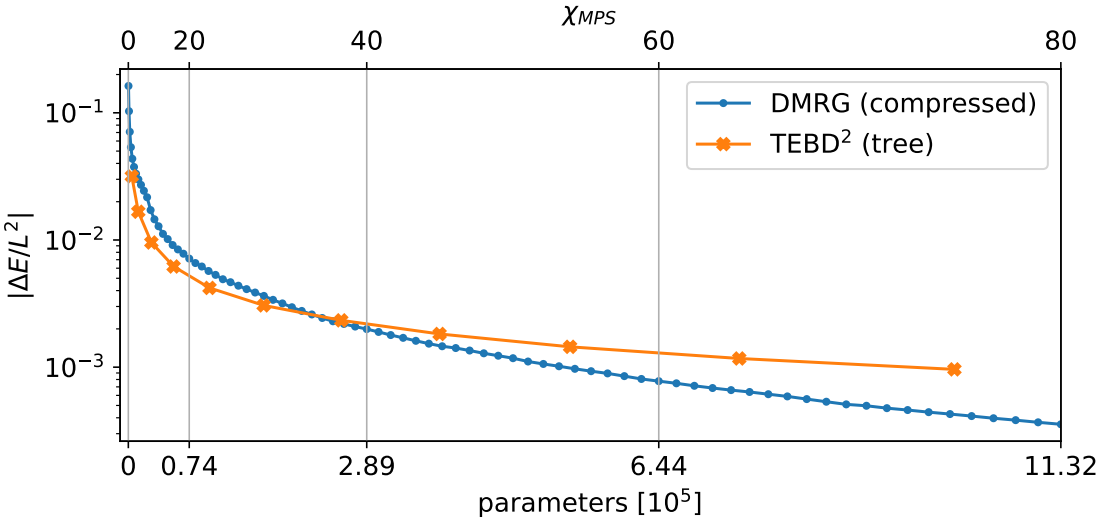


Figure 5.14: Error density of energy as a function of the number of used variational parameters in the TFI model, for modified TEBD^2 algorithm and state obtained with 1D-DMRG, which was further variationally compressed. Calculations were performed for a 10×10 lattice and $g = 3.1$. The horizontal axis above the plot shows values of χ corresponding to a given number of variational parameters, when an MPS-based technique is used. However, this serves only as a reference point, and in the case of tree-like isoTNS different values of χ were used.

Finally, we compared the results given by the modified TEBD^2 and its original counterpart. For that purpose we also investigated, how the energy obtained with each algorithm varies with respect to the number of used parameters. In the case of the tree-like isoTNS we took into account only the final value returned by this method, as the energies for consecutive values of τ were approaching the true ground state energy from below and intersected with it at certain point, which can be seen in Fig. 5.13. On the other hand, in the case of the original TEBD^2 we took both the minimum and final values of energy, because we have seen that for small enough values of τ the energy starts to diverge. Outcomes of this comparison are shown in Table 5.1.

First thing to notice, is that for similar number of used parameters the minimum values of energy calculated with the original TEBD^2 method are smaller than the corresponding ones obtained with the modified version of the algorithm. However, with the increasing number of parameters in the tensor network this difference diminishes, and we expect that asymptotically the energy obtained with both of these techniques would be comparable. Moreover, the energy obtained with the TEBD^2 on a tree-like isoTNS is significantly smaller than the final one obtained with the original method, which highlights the usefulness of

	isoTNS - tree		isoTNS - benchmark			
Parameters	χ_{tree}	Final energy	χ	η	Minimum energy	Final energy
4232	4	-323.0923				
7840			2	12	-324.4980	-316.4407
12096	6	-324.5932				
28008	8	-325.3048				
54992	10	-325.6417				
80688			4	24	-326.0377	-316.9485
98568	12	-325.8405				
164304	14	-325.9545				
258536	16	-326.0271				
287444			6	36	-326.1756	-317.0089
377928	18	-326.0781				
536360	20	-326.1160				
741800	22	-326.1435				
1002984	24	-326.1644				

Table 5.1: Energies obtained with two version of the TEBD^2 algorithm with respect to the number of variational parameters stored in the tensor network. The simulations were performed for a TFI model on a 10×10 lattice with $g = 3.1$.

the modifications proposed in this work. The last aspect worth mentioning is that as the bond-size increases, the energies given by both methods converge towards the true ground state energy.

5.10 Simulation of 2D quantum computers via tree-like isoTNS

In the final section of this chapter we return to the task of noisy simulation of quantum computers. This time we will use for that purpose the TEBD^2 algorithm on a tree-like isoTNS, where we simply replace the time evolution operators with quantum gates.

We firstly checked, if our prediction of the multi-qubit fidelity $\mathcal{F}$, calculated as a product of fidelities of consecutive SVDs, is in line with the square of the overlap between the exact and truncated states. In Fig. 5.15 we show that in fact our approximations are accurate, on an example of random quantum circuits applied on a 5×5 lattice with various types of two-qubit gates. It should be noted that because we are updating a single quarter of the lattice at a time, firstly sweeping outwards, and then inwards, we are applying gates coming from two consecutive layers (whenever it is possible). As a result, after finishing operations on a given quarter we effectively apply just about $1/4$ of all the gates from two consecutive cycles. We apply the remaining gates by proceeding in a clockwise fashion, updating other three quarters of the lattice. Thus, during one whole step of the algorithm we apply two cycles of the circuit. Because of this property random quantum circuits seem to be a perfect fit for our method, as the layers of horizontal and vertical gates always come in pairs.

However, this feature of our algorithm also comes with a certain inconvenience. The *cirq* exact simulator gives access to the state vector only after the application of gates in the whole layer is finished, due to which we were comparing our truncated state with the perfect one

only after every second cycle. Despite this, we see an excellent precision of our approximation of $\mathcal{F}$, which can be seen in Fig. 5.15.

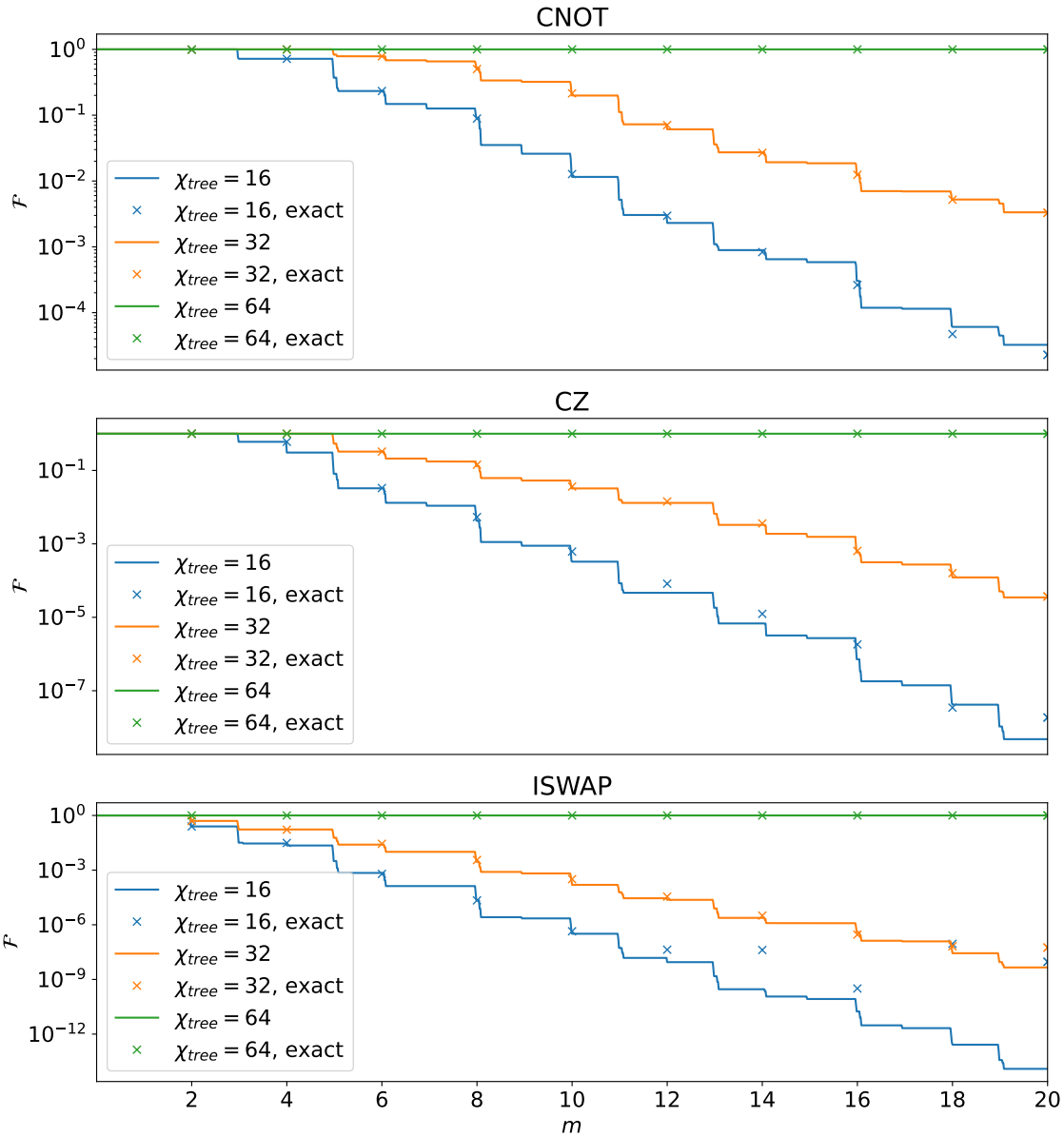


Figure 5.15: Multi-qubit fidelity of the simulation of random quantum circuits with the *CNOT*, *CZ* and *ISWAP* gates applied on a 5×5 lattice. The calculations were performed with an algorithm operating on a tree-like isoTNS, for varying bond-sizes. The square of the overlap between the truncated and exact states is marked with crosses.

Knowing that we can precisely predict the multi-qubit fidelity of simulation performed with our method, we moved on to checking how does the average two-qubit fidelity f_{avg} scale with the size of the system. For that purpose we chose three values of the bond-size χ_{tree} equal to 64, 68 and 72, and gradually increased the width and length of the lattice. Concurrently, for all simulations conducted with the algorithm operating on a tree-like isoTNS, we performed analogous computations using the MPS-based method. For each instance of MPS calculations we used the bond-size χ_{MPS} giving a slightly larger number of variational parameters kept in the tensor network, as compared to the tree-like isoTNS. For example, for a 6×5 lattice and $\chi_{tree} = 72$, an analogous number of variables is stored in an

MPS with $\chi_{MPS} = 1454$. We chose to use an MPS with a bit larger number of parameters to check, whether a gain in precision with simultaneous compression of the quantum state can be obtained with our method. Results of the average two-qubit fidelity for the two described methods are depicted in Fig. 5.16.

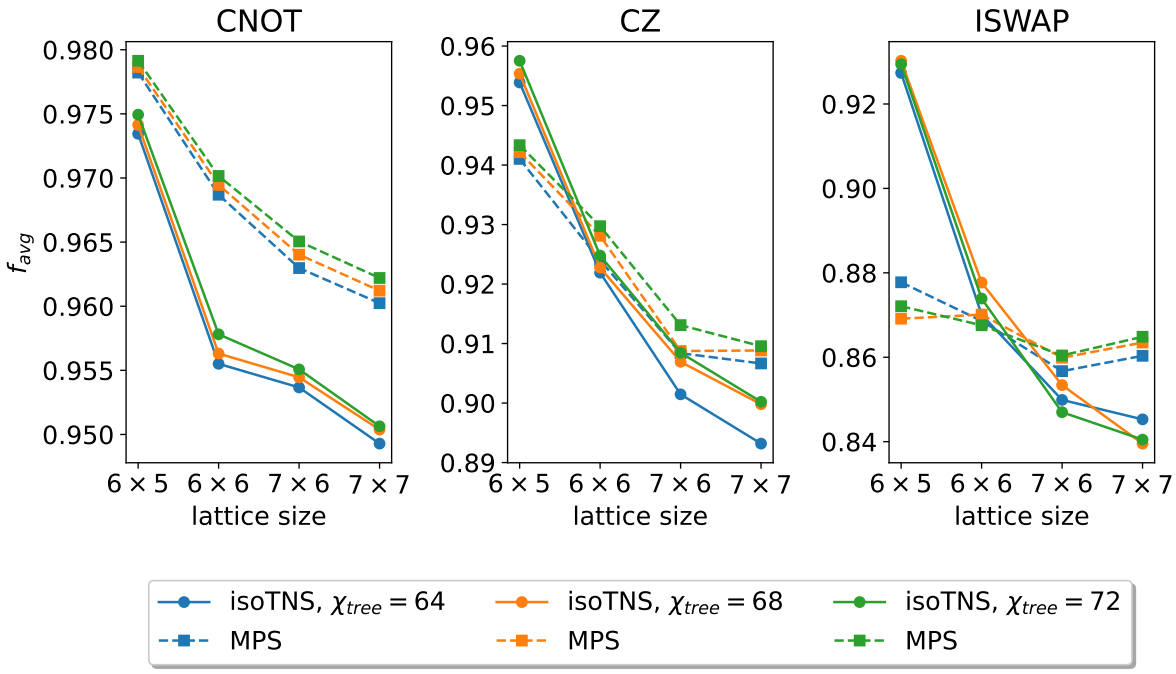


Figure 5.16: Comparison of the average two-qubit fidelity for the algorithm using a tree-like isoTNS, and an MPS-based method.

Surprisingly, even though in the case of the MPS-based simulations we are conducting large number of SWAP operations in order to apply vertical gates, the average precision of this method scales better than the one of the algorithm operating on a tree-like isoTNS. Nonetheless, results of the two methods give very high average fidelity, if one considers how small is the fraction of parameters stored in the final tensor network, as compared to an exact simulation. For example, in the case of a 7×7 lattice and circuit including the *CNOT* gates, the MPS-based method gave $\sim 96\%$ average two-qubit fidelity and the tree-like isoTNS algorithm $\sim 95\%$ one, for just $\sim 10^{-7}$ variables being stored in the final state. It should be highlighted that during the application of gates and shifting of the orthogonality surface it is needed to operate on a larger parameter space (which will be explained in depth shortly), however it is still incomparably smaller than the full Hilbert space.

Another observation that can be drawn from Fig. 5.16 is that, as expected, the average two-qubit fidelity gradually decreases as we increase the difficulty of the task, by replacing the *CNOT* gates in the circuits with the *CZ* and *ISWAP* ones. However, the slopes of curves corresponding to both methods are similar in the case of the *CNOT* and *CZ* gates, and differ drastically for the *ISWAP* gates. This prompted us to investigate the individual fidelities of SVDs for the two techniques under study, while simulating a circuit containing the *ISWAP* gates, applied on the largest tested lattice of size 7×7. These values are presented in Fig. 5.17.

We can see that the fidelities corresponding to the MPS-based method never fall below the threshold of $1/2$. This phenomenon occurs due to the properties of SVD and the structure of the largest two-site tensor appearing in an MPS. Let us recall that for an $a \times b$ matrix,

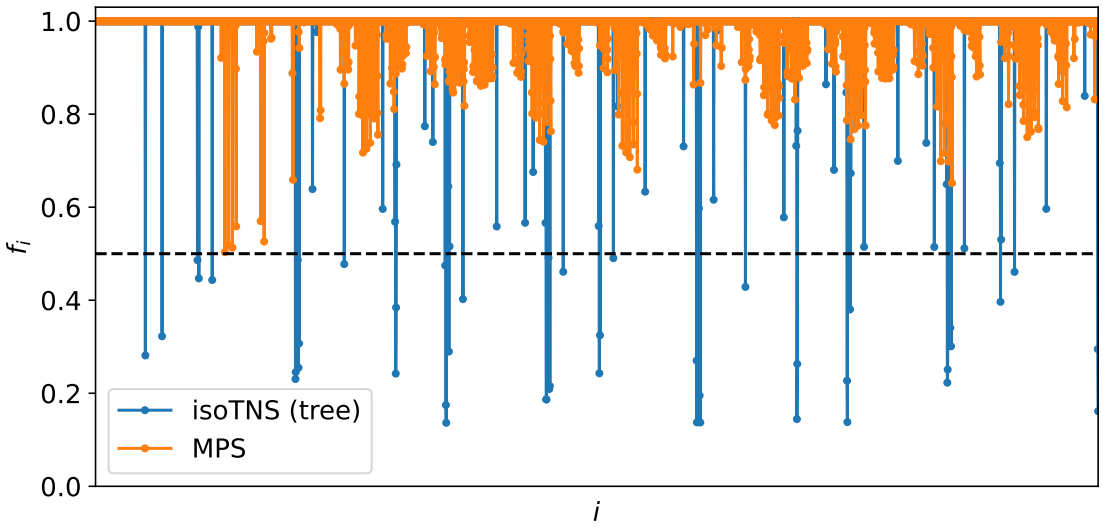


Figure 5.17: Individual fidelities of SVDs conducted during the execution of an algorithm operating on a tree-like isoTNS with $\chi_{tree} = 72$, and the MPS-based method using $\chi_{MPS} = 943$. The random quantum circuit was applied on a 7×7 lattice. Lower bound on the fidelity of SVDs performed on an MPS is marked with a dashed line.

SVD always calculates at most $\min(a, b)$ singular values. Thus, for the largest tensor in the chain (depicted in Fig. 5.18.a) being of size $d\chi \times d\chi$ we always calculate $d\chi$ singular values, from which we keep only the χ most relevant ones. As $d = 2$ in the case of qubits, we can plainly see that the lower bound on the fidelity of SVD performed on a chain is equal to $1/2$. However, this constraint does not apply to two-site tensors appearing in trees and PEPS. Example of such an object is shown in Fig. 5.18.b. Because multiple legs are combined on both ends of this tensor, in order to form an $a \times b$ matrix which can be further factorized, the minimum among a and b is always some power of χ. As we further keep only the χ largest singular values we can see that in this scenario there is a possibility of a much larger leakage of information, than is the case with MPSs.

Another issue resulting from the contraction of two separate sites into one is the increase in memory needed to store the resulting tensor, when operating on trees and PEPS systems. This problem does not arise in the context of MPSs, as two tensors containing $d\chi^2$ variables each take up the same amount of space as the two-site tensor holding $d^2\chi^2$ parameters (when working with qubits). In contrast, in the case of PEPS the number of variables stored in two single-site nodes is equal to $2 \cdot d\chi^4$, while for a two-site tensor it amounts to $d^2\chi^6$. Thus, we can see a significant increase in space needed to store such an object. This, in turn, explains the inferior scaling of memory footprint of our method with respect to an MPS-based technique.

Knowing the bottleneck of our algorithm we will end this chapter with suggestions for improvements and alternative approaches to the task of simulation of two-dimensional quantum systems. All the methods developed in this work rely exclusively on the SVD technique to assess the multi-qubit fidelity of our calculations, which, as we have seen, causes a number of problems. Although, recently a different strategy was proposed [103], which might allow to overcome these issues. The authors use an approach similar to the DMRG algorithm, in which the MPO representing the Hamiltonian is substituted with several layers of a quantum circuit. As a result, rather than calculating the two-qubit fidelity as

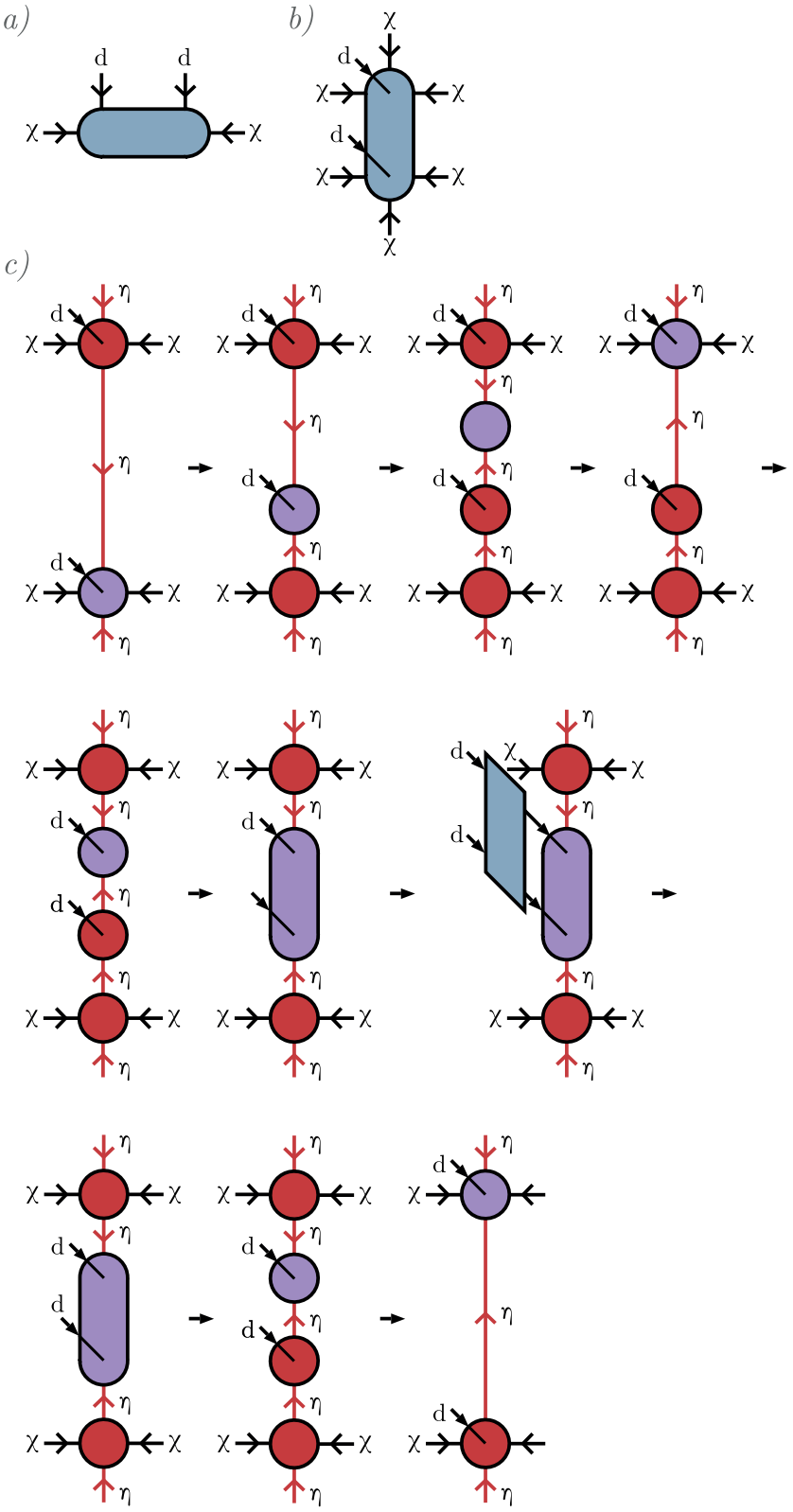


Figure 5.18: a) The largest tensor in an MPS. b) The largest tensor in a general PEPS. c) Application of a two-site quantum gate in the case of PEPS with the use of reduced tensors technique.

the sum of squares of preserved singular values, we obtain it as the norm of the last tensor

updated during the final sweep of the optimization.

It should be noted that our modified TEBD^2 algorithm can be applied on a general TTN. However, due to the necessity of contraction of two sites located in the middle of different branches, it would suffer from the same problems as the ones present in our implementation. This issue can be overcome in two steps. Firstly, we could factorize a four-legged two-site operator into two three-legged ones, e.g., by SVD. Then, we could optimize each tensor occurring on the path connecting the two sites, on which the gate is applied, in a DMRG-like fashion discussed above. As the shifting of the orthogonality center on a single branch (which would be necessary in this scenario) does not increase the maximal bond-size used, the most expensive operation of the whole method would be the calculation of environment of the updated tensor. In the case of a binary tree, in which all tensors besides the leaves are the ancilla ones, by using smart strategy for tensor contraction the cost of such procedure would amount to $\mathcal{O}(\chi^4)$ [121]. Because the distance between any pair of physical sites in a TTN scales logarithmically with respect to the total number of particles, this technique should asymptotically scale better than the MPS-based method, since the decay of the correlation functions with respect to the path length is typically exponentially fast. Numerical studies in this area would be needed to check, whether there are other obstacles that arise when using this algorithm. A potential disadvantage of this approach would be, e.g., worse performance in avoiding local minima when updating just single-site tensors, rather than the two-site ones.

Finally, we suggest possible modifications to the algorithm operating on the basic isoTNS. The first area, in which a significant improvement might be achieved, is the application of the two-site operators. The cost of naive procedure achieving this task, being equal to $\mathcal{O}(\chi^9)$, can be significantly reduced by using the *reduced tensors* method [122]. In this technique, the physical leg of given site is grouped together with a single virtual bond, which, when factorization is conducted, allows for "detachment" of the physical leg along with the ancilla one from the remaining virtual degrees of freedom. Effectively, the tensor corresponding to a physical site is moved along one of the bonds, leaving the other tensor resulting from the SVD as the ancilla one. When this method is performed on the two sites, on which the operator is ought to be applied, the two tensors being the output of this procedure can be contracted, giving as a result an order-4 tensor of size $\mathcal{O}(d^2\chi^2)$. After the update of this node is finished, it can be divided by another SVD, and the resulting tensors can be contracted with the ancilla ones, left from the preparation step of the procedure. As a result, the overall cost of the application of a two-site operator is reduced to $\mathcal{O}(\chi^5)$, which is equal to the cost of both the SVD of a single site, performed at the beginning of the algorithm, and contractions conducted at its end. All of the steps described above are depicted in Fig. 5.18.c. It should be noted that in the course of carrying out the reduced tensors method, at most three truncations are performed, each bounded from below by the fidelity of $1/2$ (in the case of qubits). Thus, the lower limit on the final fidelity of the application of a two-site tensor is equal to $1/8$, due to the multiplicative nature of the accumulation of errors. Numerical studies checking whether this bound is ever reached during the execution of real quantum circuits would be of great value.

However, even the best possible method of applying operators would not be useful to us if we were not able to predict the multi-qubit fidelity of our simulation. As we mentioned previously, we could not approximate this value using local estimates (coming from the SVDs or variational solution of the $\Theta^l = A^l\Theta$ problem), thus we propose here a more "global" approach. The problem lies precisely in estimating the precision, with which the orthogonality surface is moved. In general, this value can be computed by calculating the overlap of the

whole PEPS from before, and after the shifting. In the case of isoTNS this procedure is not as expensive, as the only update of tensors occurs in the initial $\Theta^l B^{l+1}$ columns, yielding the $A^l\Theta^{l+1}$ ones, while the rest of the lattice satisfies the isometry condition. Therefore, the precision of shifting of the orthogonality surface should be properly approximated by the $|\langle\Theta^l B^{l+1}|A^l\Theta^{l+1}\rangle|^2$ value. Nevertheless, the computation of this fidelity would cost $\mathcal{O}(\chi^8)$, and would involve storing the copy of initial and final columns, making it the most expensive part of the whole simulation algorithm.

SIXTH CHAPTER

CONCLUSIONS

The study of the properties of strongly correlated two-dimensional systems is a fascinating area of research, which at the same time poses a huge challenge due to the exponentially increasing amount of resources required to perform their exact classical simulation. A large-scale quantum computer could be potentially used to overcome this problem, however its construction is itself an extremely difficult engineering endeavor. Until such a fault-tolerant machine is built, approximation methods using tensor networks serve as one of the most powerful tools to explore the interesting physics of complex quantum systems.

In this thesis we focused on the algorithms utilising the so-called isometry condition, allowing for significant areas of tensor networks to be ignored when conducting their update, or calculating the expectation values of observables of interest. We began with a pedagogical introduction to the 1D MPSs and presented a number of algorithms, such as variational compression, TEBD and DMRG (both in its finite and infinite versions). We showed how to increase the efficiency of simulation by introducing the concept of charge conservation. We also illustrated how 2D systems might be mapped onto a 1D chain, which in turn allows the use of the aforementioned methods to analyze their properties.

We then used the DMRG technique to analyze two types of physical models. Firstly, we localized the critical point of the transverse field Ising model on a finite $L \times L$ rectangular lattice, for varying values of L, and compared the obtained results with the ones presented in Refs. [39–42] in order to check their correctness. From the entire spectrum of investigated Hamiltonian parameters we selected a set giving a comparable amount of entanglement being present in the system, for increasing size of the lattice. These results served us as a benchmark against which we compared algorithms tailored specifically for the two-dimensional tensor networks, developed in the further part of this work.

Secondly, we investigated the properties of the spin-3/2 XXZ Hamiltonian with a single ion anisotropy on a honeycomb lattice, which was shown to be an effective model of the monolayer of CrI_3 [43]. Isolated CrI_3 manifests the ferromagnetic order in the off-plane axis, with a Curie temperature $T_C = 45$ K [78], while other magnetic phases exhibited by the Hamiltonian can be achieved by the introduction of defects [96], strain [97, 98] and charge doping [98, 99], which enhance magnetic anisotropy. We showed that a classical approximation of this model can be used to predict the type of magnetic ordering with high fidelity in a wide range of the parameter space. The in-plane ferromagnetic and antiferromagnetic phases have correlation energies that are the greatest in magnitude, while the off-plane ferromagnetic

phase has a correlation energy of zero. With the help of our findings, it is feasible to identify the Hamiltonian parameters for which the energy gap is the largest, increasing the stability of the resulting phase.

Afterwards, we turned to the task of noisy simulation of quantum circuits with the use of MPS-based methods, which by reducing the amount of entanglement being present in the system yield a finite precision [45]. We explained, how the two-qubit fidelity can be approximated by the sum of squares of singular values kept during the SVD, and how the multi-qubit fidelity can be obtained by the product of the two-qubit ones. We demonstrated the concept of random quantum circuits, which were used by Google Inc. in their famous quantum supremacy experiment [44]. We showed how the fidelity of simulation of this kind of circuits depends on the type of multi-qubit gates used, being the highest for the *CNOT* gates, and decreasing gradually when utilizing the *CZ* and *ISWAP* ones.

Next, we introduced the two-dimensional Isometric Tensor Networks, proposed initially in Refs. [32, 34]. We gave a detailed description of the Moses Move technique, allowing to shift the orthogonality surface over the whole lattice, and the TEBD^2 algorithm, which in turn can be used to obtain the ground state of a chosen physical model by means of imaginary time evolution. Then, we presented two modifications of the initial framework, designed specifically for the purpose of this work. The first one allows to move the whole orthogonality surface of an isoTNS into the bulk, effectively allowing for a significant reduction of the bond-size required to conduct given computations. The second alteration consisted of two parts. The first one involved the rearrangement of the flow of entanglement through the lattice, making it effectively a tree-like structure. Second one used tensors contraction and SVDs to merge different branches, making it possible to apply the two-site operators on nodes not being directly connected in the initial tree.

We compared both the original version of the TEBD^2 and the modified one utilizing the aforementioned alterations on the task of obtaining the ground state of the transverse field Ising model on a square lattice. As a reference point we used the results calculated with the use of the DMRG algorithm. We have seen that the adjusted version of the TEBD^2 exhibited much more stable convergence than the basic one. As the initial implementation diverged for small enough length of the time-step, we kept track of both the minimum value of energy it gave during the whole run-time, as well as the final one. We anticipate from the altered TEBD^2 to give asymptotically the same energy (with respect to the number of variational parameters stored in the final tensor network) as the minimal one obtained with the basic version. Whereas in the case of the last values obtained with both methods the one proposed in this work was always superior.

Finally, we adjusted the modified TEBD^2 algorithm to the task of simulation of random quantum circuits. Thanks to the introduced alterations we were able to predict the multi-qubit fidelity of our calculations with high precision, which we were not able to achieve with the initial version of the isoTNS formalism, due to the error caused by the Moses Move. Using this method, we obtained very high average two-qubit fidelity with only a fraction of the parameters used, with respect to the number required by an exact simulation, which also depended on the amount of entanglement being present in the simulated state. We compared these results with those of the MPS-based technique. For the largest system studied, being a grid of size 7×7, the average two-qubit fidelity of the modified TEBD^2 algorithm was worse by $\sim$1% in the case of the *CNOT* gates used in the circuit, and $\sim$2% worse for the *CZ* and *ISWAP* gates, than the fidelity obtained with the MPS-based method. We found the contraction of two sites into a single tensor and merging of different branches to be the bottlenecks of the modified TEBD^2 algorithm, and proposed a DMRG-based approach potentially allowing to

bypass these issues and extend our algorithm to the case of general tree tensor networks. In the case of the basic isoTNS we also suggested a method involving the calculation of the square of the overlap between the two columns being the input of the Moses Move, and the ones being its output, potentially allowing an exact estimation of the multi-qubit fidelity when shifting the orthogonality surface. This technique might allow the use of standard isoTNS to its full extent in the context of simulating random quantum circuits. However, the last two conjectures should be verified by extensive numerical studies.

Altogether, the results presented in this thesis show the great power of tensor networks as a tool to investigate the properties of quantum systems. The two-dimensional isoTNS seem to hold great potential in the context of studying physical models, although there is still no consensus in the community as to which of the methods used to canonize PEPS is best, and further research in this area is needed. New discoveries in this field could be also used in the context of noisy simulation of quantum circuits, resulting in calculations of even higher fidelity.

BIBLIOGRAPHY

[1] J. Quintanilla and C. Hooley, "The strong-correlations puzzle", *Physics World*, vol. 22, no. 06, p. 32, Jun. 2009. doi: 10.1088/2058-7058/22/06/38. [Online]. Available: https://dx.doi.org/10.1088/2058-7058/22/06/38

[2] J. G. Bednorz and K. A. Müller, "Possible high Tc superconductivity in the Ba-La-Cu-O system", *Zeitschrift für Physik B Condensed Matter*, vol. 64, no. 2, pp. 189–193, Jun. 1986. doi: 10.1007/BF01303701. [Online]. Available: https://doi.org/10.1007/BF01303701

[3] A. Aharoni, *Introduction to the theory of ferromagnetism*, 2nd ed. Oxford. :: Oxford University Press, 2000. ISBN 978-0-19-850808-3 OCLC: 1123630092.

[4] R. P. Feynman, "Simulating physics with computers", *International Journal of Theoretical Physics*, vol. 21, no. 6-7, pp. 467–488, Jun. 1982. doi: 10.1007/BF02650179. [Online]. Available: http://link.springer.com/10.1007/BF02650179

[5] E. Manousakis, "A Quantum-Dot Array as Model for Copper-Oxide Superconductors: A Dedicated Quantum Simulator for the Many-Fermion Problem", Jan. 2002, arXiv:cond-mat/0201142. [Online]. Available: http://arxiv.org/abs/cond-mat/0201142

[6] D. Porras and J. I. Cirac, "Effective Quantum Spin Systems with Trapped Ions", *Physical Review Letters*, vol. 92, no. 20, p. 207901, May 2004. doi: 10.1103/PhysRevLett.92.207901. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevLett.92.207901

[7] A. Y. Smirnov, S. Savel'ev, L. G. Mourokh, and F. Nori, "Modelling chemical reactions using semiconductor quantum dots", *Europhysics Letters (EPL)*, vol. 80, no. 6, p. 67008, Dec. 2007. doi: 10.1209/0295-5075/80/67008. [Online]. Available: https://iopscience.iop.org/article/10.1209/0295-5075/80/67008

[8] I. Georgescu, S. Ashhab, and F. Nori, "Quantum simulation", *Reviews of Modern Physics*, vol. 86, no. 1, pp. 153–185, Mar. 2014. doi: 10.1103/RevModPhys.86.153. [Online]. Available: https://link.aps.org/doi/10.1103/RevModPhys.86.153

[9] S. Lloyd, "Universal Quantum Simulators", *Science*, vol. 273, no. 5278, pp. 1073–1078, Aug. 1996. doi: 10.1126/science.273.5278.1073. [Online]. Available: https://www.science.org/doi/10.1126/science.273.5278.1073

[10] D. S. Abrams and S. Lloyd, "Simulation of Many-Body Fermi Systems on a Universal Quantum Computer", *Physical Review Letters*, vol. 79, no. 13, pp. 2586–2589, Sep. 1997. doi: 10.1103/PhysRevLett.79.2586. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevLett.79.2586

[11] D. A. Lidar and O. Biham, "Simulating Ising Spin Glasses on a Quantum Computer", *Physical Review E*, vol. 56, no. 3, pp. 3661–3681, Sep. 1997. doi: 10.1103/PhysRevE.56.3661 ArXiv:quant-ph/9611038. [Online]. Available: http://arxiv.org/abs/quant-ph/9611038

[12] G. Ortiz, J. E. Gubernatis, E. Knill, and R. Laflamme, "Quantum algorithms for fermionic simulations", *Physical Review A*, vol. 64, no. 2, p. 022319, Jul. 2001. doi: 10.1103/PhysRevA.64.022319. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevA.64.022319

[13] F. Verstraete, J. I. Cirac, and J. I. Latorre, "Quantum circuits for strongly correlated quantum systems", *Physical Review A*, vol. 79, no. 3, p. 032316, Mar. 2009. doi: 10.1103/PhysRevA.79.032316. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevA.79.032316

[14] R. Somma, G. Ortiz, J. E. Gubernatis, E. Knill, and R. Laflamme, "Simulating physical phenomena by quantum networks", *Physical Review A*, vol. 65, no. 4, p. 042323, Apr. 2002. doi: 10.1103/PhysRevA.65.042323. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevA.65.042323

[15] M. A. Nielsen and I. L. Chuang, *Quantum computation and quantum information*, 10th ed. Cambridge ; New York: Cambridge University Press, 2010. ISBN 978-1-107-00217-3

[16] S. Raeisi, N. Wiebe, and B. C. Sanders, "Quantum-circuit design for efficient simulations of many-body quantum dynamics", *New Journal of Physics*, vol. 14, no. 10, p. 103017, Oct. 2012. doi: 10.1088/1367-2630/14/10/103017. [Online]. Available: https://iopscience.iop.org/article/10.1088/1367-2630/14/10/103017

[17] S. J. Devitt, W. J. Munro, and K. Nemoto, "Quantum error correction for beginners", *Reports on Progress in Physics*, vol. 76, no. 7, p. 076001, Jul. 2013. doi: 10.1088/0034-4885/76/7/076001. [Online]. Available: https://iopscience.iop.org/article/10.1088/0034-4885/76/7/076001

[18] P. W. Shor, "Scheme for reducing decoherence in quantum computer memory", *Physical Review A*, vol. 52, no. 4, pp. R2493–R2496, Oct. 1995. doi: 10.1103/PhysRevA.52.R2493. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevA.52.R2493

[19] A. M. Steane, "Error Correcting Codes in Quantum Theory", *Physical Review Letters*, vol. 77, no. 5, pp. 793–797, Jul. 1996. doi: 10.1103/PhysRevLett.77.793. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevLett.77.793

[20] P. W. Shor, "Fault-tolerant quantum computation", Mar. 1997, arXiv:quant-ph/9605011. [Online]. Available: http://arxiv.org/abs/quant-ph/9605011

[21] D. Gottesman, "An Introduction to Quantum Error Correction and Fault-Tolerant Quantum Computation", Apr. 2009, arXiv:0904.2557 [quant-ph]. [Online]. Available: http://arxiv.org/abs/0904.2557

[22] E. T. Campbell, B. M. Terhal, and C. Vuillot, "Roads towards fault-tolerant universal quantum computation", *Nature*, vol. 549, no. 7671, pp. 172–179, Sep. 2017. doi: 10.1038/nature23460 ArXiv:1612.07330 [quant-ph]. [Online]. Available: http://arxiv.org/abs/1612.07330

[23] J. Preskill, "Quantum Computing in the NISQ era and beyond", *Quantum*, vol. 2, p. 79, Aug. 2018. doi: 10.22331/q-2018-08-06-79 ArXiv:1801.00862 [cond-mat, physics:quant-ph]. [Online]. Available: http://arxiv.org/abs/1801.00862

[24] M. Fannes, B. Nachtergaele, and R. F. Werner, "Finitely correlated states on quantum spin chains", *Communications in Mathematical Physics*, vol. 144, no. 3, pp. 443–490, Mar. 1992. doi: 10.1007/BF02099178. [Online]. Available: https://doi.org/10.1007/BF02099178

[25] G. Vidal, "Efficient Classical Simulation of Slightly Entangled Quantum Computations", *Physical Review Letters*, vol. 91, no. 14, p. 147902, Oct. 2003. doi: 10.1103/PhysRevLett.91.147902. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevLett.91.147902

[26] G. Vidal, "Efficient Simulation of One-Dimensional Quantum Many-Body Systems", *Physical Review Letters*, vol. 93, no. 4, p. 040502, Jul. 2004. doi: 10.1103/PhysRevLett.93.040502. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevLett.93.040502

[27] D. Perez-Garcia, F. Verstraete, M. M. Wolf, and J. I. Cirac, "Matrix Product State Representations", May 2007, arXiv:quant-ph/0608197. [Online]. Available: http://arxiv.org/abs/quant-ph/0608197

[28] U. Schollwöck, "The density-matrix renormalization group in the age of matrix product states", *Annals of Physics*, vol. 326, no. 1, pp. 96–192, Jan. 2011. doi: 10.1016/j.aop.2010.09.012. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0003491610001752

[29] F. Verstraete and J. I. Cirac, "Renormalization algorithms for Quantum-Many Body Systems in two and higher dimensions", Jul. 2004, arXiv:cond-mat/0407066. [Online]. Available: http://arxiv.org/abs/cond-mat/0407066

[30] F. Verstraete, M. M. Wolf, D. Perez-Garcia, and J. I. Cirac, "Criticality, the area law, and the computational power of PEPS", *Physical Review Letters*, vol. 96, no. 22, p. 220601, Jun. 2006. doi: 10.1103/PhysRevLett.96.220601 ArXiv:quant-ph/0601075. [Online]. Available: http://arxiv.org/abs/quant-ph/0601075

[31] R. Haghshenas, M. J. O'Rourke, and G. K.-L. Chan, "Conversion of projected entangled pair states into a canonical form", *Physical Review B*, vol. 100, no. 5, p. 054404, Aug. 2019. doi: 10.1103/PhysRevB.100.054404. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevB.100.054404

[32] M. P. Zaletel and F. Pollmann, "Isometric Tensor Network States in Two Dimensions", *Physical Review Letters*, vol. 124, no. 3, p. 037201, Jan. 2020. doi: 10.1103/PhysRevLett.124.037201 ArXiv:1902.05100 [cond-mat, physics:quant-ph]. [Online]. Available: http://arxiv.org/abs/1902.05100

[33] K. Hyatt and E. M. Stoudenmire, "DMRG Approach to Optimizing Two-Dimensional Tensor Networks", Apr. 2020, arXiv:1908.08833 [cond-mat, physics:quant-ph]. [Online]. Available: http://arxiv.org/abs/1908.08833

[34] S.-H. Lin, M. Zaletel, and F. Pollmann, "Efficient Simulation of Dynamics in Two-Dimensional Quantum Spin Systems with Isometric Tensor Networks", *Physical Review B*, vol. 106, no. 24, p. 245102, Dec. 2022. doi: 10.1103/PhysRevB.106.245102 ArXiv:2112.08394 [cond-mat, physics:quant-ph]. [Online]. Available: http://arxiv.org/abs/2112.08394

[35] S. R. White, "Density matrix formulation for quantum renormalization groups", *Physical Review Letters*, vol. 69, no. 19, pp. 2863–2866, Nov. 1992. doi: 10.1103/PhysRevLett.69.2863. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevLett.69.2863

[36] S. R. White, "Density-matrix algorithms for quantum renormalization groups", *Physical Review B*, vol. 48, no. 14, pp. 10 345–10 356, Oct. 1993. doi: 10.1103/PhysRevB.48.10345. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevB.48.10345

[37] I. P. McCulloch, "From density-matrix renormalization group to matrix product states", *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2007, no. 10, pp. P10 014–P10 014, Oct. 2007. doi: 10.1088/1742-5468/2007/10/P10014. [Online]. Available: https://iopscience.iop.org/article/10.1088/1742-5468/2007/10/P10014

[38] J. Hauschild and F. Pollmann, "Efficient numerical simulations with Tensor Networks: Tensor Network Python (TeNPy)", *SciPost Physics Lecture Notes*, p. 005, Oct. 2018. doi: 10.21468/SciPostPhysLectNotes.5. [Online]. Available: https://scipost.org/10.21468/SciPostPhysLectNotes.5

[39] M. S. L. d. C. de Jongh and J. M. J. van Leeuwen, "The critical behaviour of the 2D Ising model in Transverse Field; a Density Matrix Renormalization calculation", *Physical Review B*, vol. 57, no. 14, pp. 8494–8500, Apr. 1998. doi: 10.1103/PhysRevB.57.8494 ArXiv:cond-mat/9709103. [Online]. Available: http://arxiv.org/abs/cond-mat/9709103

[40] H. W. J. Blöte and Y. Deng, "Cluster Monte Carlo simulation of the transverse Ising model", *Physical Review E*, vol. 66, no. 6, p. 066110, Dec. 2002. doi: 10.1103/PhysRevE.66.066110. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevE.66.066110

[41] I. Frérot and T. Roscilde, "Reconstructing the quantum critical fan of strongly correlated systems using quantum correlations", *Nature Communications*, vol. 10, no. 1, p. 577, Feb. 2019. doi: 10.1038/s41467-019-08324-9 Number: 1 Publisher: Nature Publishing Group. [Online]. Available: https://www.nature.com/articles/s41467-019-08324-9

[42] M. Schmitt, M. M. Rams, J. Dziarmaga, M. Heyl, and W. H. Zurek, "Quantum phase transition dynamics in the two-dimensional transverse-field Ising model", *Science Advances*, vol. 8, no. 37, p. eabl6850, Sep. 2022. doi: 10.1126/sciadv.abl6850 Publisher: American Association for the Advancement of Science. [Online]. Available: https://www.science.org/doi/10.1126/sciadv.abl6850

[43] J. L. Lado and J. Fernández-Rossier, "On the origin of magnetic anisotropy in two dimensional CrI3", *2D Materials*, vol. 4, no. 3, p. 035002, jun 2017. doi: 10.1088/2053-1583/aa75ed. [Online]. Available: https://doi.org/10.1088%2F2053-1583%2Faa75ed

[44] F. Arute, K. Arya, R. Babbush, D. Bacon, J. C. Bardin, R. Barends, R. Biswas, S. Boixo, F. G. S. L. Brandao, D. A. Buell, B. Burkett, Y. Chen, Z. Chen, B. Chiaro, R. Collins,

W. Courtney, A. Dunsworth, E. Farhi, B. Foxen, A. Fowler, C. Gidney, M. Giustina, R. Graff, K. Guerin, S. Habegger, M. P. Harrigan, M. J. Hartmann, A. Ho, M. Hoffmann, T. Huang, T. S. Humble, S. V. Isakov, E. Jeffrey, Z. Jiang, D. Kafri, K. Kechedzhi, J. Kelly, P. V. Klimov, S. Knysh, A. Korotkov, F. Kostritsa, D. Landhuis, M. Lindmark, E. Lucero, D. Lyakh, S. Mandrà, J. R. McClean, M. McEwen, A. Megrant, X. Mi, K. Michielsen, M. Mohseni, J. Mutus, O. Naaman, M. Neeley, C. Neill, M. Y. Niu, E. Ostby, A. Petukhov, J. C. Platt, C. Quintana, E. G. Rieffel, P. Roushan, N. C. Rubin, D. Sank, K. J. Satzinger, V. Smelyanskiy, K. J. Sung, M. D. Trevithick, A. Vainsencher, B. Villalonga, T. White, Z. J. Yao, P. Yeh, A. Zalcman, H. Neven, and J. M. Martinis, "Quantum supremacy using a programmable superconducting processor", *Nature*, vol. 574, no. 7779, pp. 505–510, Oct. 2019. doi: 10.1038/s41586-019-1666-5. [Online]. Available: http://www.nature.com/articles/s41586-019-1666-5

[45] Y. Zhou, E. M. Stoudenmire, and X. Waintal, "What limits the simulation of quantum computers?" *Physical Review X*, vol. 10, no. 4, p. 041038, Nov. 2020. doi: 10.1103/PhysRevX.10.041038 ArXiv:2002.07730 [cond-mat, physics:quant-ph]. [Online]. Available: http://arxiv.org/abs/2002.07730

[46] J. Biamonte and V. Bergholm, "Tensor Networks in a Nutshell", Jul. 2017, arXiv:1708.00006 [cond-mat, physics:gr-qc, physics:hep-th, physics:math-ph, physics:quant-ph]. [Online]. Available: http://arxiv.org/abs/1708.00006

[47] R. Penrose, "Applications of negative dimensional tensors", *Combinatorial Mathematics and Its Applications*, vol. 1, p. 221–244, 1971.

[48] M. B. Hastings, "An area law for one-dimensional quantum systems", *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2007, no. 08, p. P08024, aug 2007. doi: 10.1088/1742-5468/2007/08/P08024. [Online]. Available: https://dx.doi.org/10.1088/1742-5468/2007/08/P08024

[49] J. Eisert, M. Cramer, and M. B. Plenio, "Colloquium: Area laws for the entanglement entropy", *Rev. Mod. Phys.*, vol. 82, pp. 277–306, Feb 2010. doi: 10.1103/RevModPhys.82.277. [Online]. Available: https://link.aps.org/doi/10.1103/RevModPhys.82.277

[50] F. Verstraete and J. I. Cirac, "Matrix product states represent ground states faithfully", *Phys. Rev. B*, vol. 73, p. 094423, Mar 2006. doi: 10.1103/PhysRevB.73.094423. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevB.73.094423

[51] F. Verstraete, J. J. García-Ripoll, and J. I. Cirac, "Matrix Product Density Operators: Simulation of Finite-Temperature and Dissipative Systems", *Physical Review Letters*, vol. 93, no. 20, p. 207204, Nov. 2004. doi: 10.1103/PhysRevLett.93.207204. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevLett.93.207204

[52] B. Pirvu, V. Murg, J. I. Cirac, and F. Verstraete, "Matrix product operator representations", *New Journal of Physics*, vol. 12, no. 2, p. 025012, Feb. 2010. doi: 10.1088/1367-2630/12/2/025012. [Online]. Available: https://iopscience.iop.org/article/10.1088/1367-2630/12/2/025012

[53] F. Fröwis, V. Nebendahl, and W. Dür, "Tensor operators: Constructions and applications for long-range interaction systems", *Physical Review A*, vol. 81, no. 6,

p. 062337, Jun. 2010. doi: 10.1103/PhysRevA.81.062337. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevA.81.062337

[54] K. Batselier, W. Yu, L. Daniel, and N. Wong, "Computing low-rank approximations of large-scale matrices with the tensor network randomized svd", 2017. doi: 10.48550/ARXIV.1707.07803. [Online]. Available: https://arxiv.org/abs/1707.07803

[55] G. M. Crosswhite and D. Bacon, "Finite automata for caching in matrix product algorithms", *Phys. Rev. A*, vol. 78, p. 012356, Jul 2008. doi: 10.1103/PhysRevA.78.012356. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevA.78.012356

[56] M. Suzuki, "Generalized Trotter's formula and systematic approximants of exponential operators and inner derivations with applications to many-body problems", *Communications in Mathematical Physics*, vol. 51, no. 2, pp. 183–190, Jun. 1976. doi: 10.1007/BF01609348. [Online]. Available: http://link.springer.com/10.1007/BF01609348

[57] M. Suzuki, "On the convergence of exponential operators—the Zassenhaus formula, BCH formula and systematic approximants", *Communications in Mathematical Physics*, vol. 57, no. 3, pp. 193–200, Oct. 1977. doi: 10.1007/BF01614161. [Online]. Available: http://link.springer.com/10.1007/BF01614161

[58] M. Suzuki, "Decomposition formulas of exponential operators and Lie exponentials with some applications to quantum mechanics and statistical physics", *Journal of Mathematical Physics*, vol. 26, no. 4, pp. 601–612, Apr. 1985. doi: 10.1063/1.526596. [Online]. Available: http://aip.scitation.org/doi/10.1063/1.526596

[59] B. Rzepkowski, M. Kupczyński, P. Potasz, and A. Wójs, "DMRG and Monte Carlo studies of CrI3 magnetic phases and the phase transition", *Physica E: Low-dimensional Systems and Nanostructures*, vol. 146, p. 115520, 2023. doi: https://doi.org/10.1016/j.physe.2022.115520. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1386947722003435

[60] A. Geim and I. Grigorieva, "Van der waals heterostructures", *Nature*, vol. 499, p. 419–425, 2013. doi: 10.1038/nature12385. [Online]. Available: https://doi.org/10.1038/nature12385

[61] K. F. Mak, C. Lee, J. Hone, J. Shan, and T. F. Heinz, "Atomically thin mos_2: A new direct-gap semiconductor", *Phys. Rev. Lett.*, vol. 105, p. 136805, Sep 2010. doi: 10.1103/PhysRevLett.105.136805. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevLett.105.136805

[62] Q. H. Wang, K. Kalantar-Zadeh, A. Kis, J. N. Coleman, and M. S. Strano, "Electronics and optoelectronics of two-dimensional transition metal dichalcogenides", *Nature Nanotechnology*, vol. 7, p. 699–712, 2012. doi: 10.1038/nnano.2012.193. [Online]. Available: https://doi.org/10.1038/nnano.2012.193

[63] M. Chhowalla, H. S. Shin, G. Eda, L. J. Li, K. P. Loh, and H. Zhang, "The chemistry of two-dimensional layered transition metal dichalcogenide nanosheets", *Nature Chemistry*, vol. 5, p. 263–275, 2013. doi: 10.1038/nchem.1589. [Online]. Available: https://doi.org/10.1038/nchem.1589

[64] J. M. Lu, O. Zheliuk, I. Leermakers, N. F. Q. Yuan, U. Zeitler, K. T. Law, and J. T. Ye, "Evidence for two-dimensional ising superconductivity in gated mos2", *Science*, vol. 350, no. 6266, pp. 1353–1357, 2015. doi: 10.1126/science.aab2277. [Online]. Available: https://science.sciencemag.org/content/350/6266/1353

[65] X. Xi, L. Zhao, Z. Wang, H. Berger, L. Forró, and K. F. Shan, J.and Mak, "Strongly enhanced charge-density-wave order in monolayer nbse2", *Nature Nanotechnology*, vol. 10, pp. 765–769, 2015. [Online]. Available: https://doi.org/10.1038/nnano.2015.143

[66] M. M. Ugeda, A. J. Bradley, Y. Zhang, S. Onishi, Y. Chen, W. Ruan, C. Ojeda-Aristizabal, H. Ryu, M. T. Edmonds, H.-Z. Tsai, A. Riss, S.-K. Mo, D. Lee, A. Zettl, Z. Hussain, Z.-X. Shen, and M. F. Crommie, "Characterization of collective ground states in single-layer nbse2", *Nature Physics*, vol. 12, pp. 92–97, 2016. [Online]. Available: https://doi.org/10.1038/nphys3527

[67] K. S. Novoselov, A. Mishchenko, A. Carvalho, and A. H. Castro Neto, "2d materials and van der waals heterostructures", *Science*, vol. 353, no. 6298, 2016. doi: 10.1126/science.aac9439. [Online]. Available: https://science.sciencemag.org/content/353/6298/aac9439

[68] Y. Ren, Z. Qiao, and Q. Niu, "Topological phases in two-dimensional materials: a review", *Reports on Progress in Physics*, vol. 79, no. 6, p. 066501, may 2016. doi: 10.1088/0034-4885/79/6/066501. [Online]. Available: https://doi.org/10.1088%2F0034-4885%2F79%2F6%2F066501

[69] M. Bieniek, T. Woźniak, and P. Potasz, "Stability of topological properties of bismuth (1 1 1) bilayer", *Journal of Physics: Condensed Matter*, vol. 29, no. 15, p. 155501, mar 2017. doi: 10.1088/1361-648x/aa5e79. [Online]. Available: https://doi.org/10.1088%2F1361-648x%2Faa5e79

[70] G. Wang, A. Chernikov, M. M. Glazov, T. F. Heinz, X. Marie, T. Amand, and B. Urbaszek, "Colloquium: Excitons in atomically thin transition metal dichalcogenides", *Rev. Mod. Phys.*, vol. 90, p. 021001, Apr 2018. doi: 10.1103/RevModPhys.90.021001. [Online]. Available: https://link.aps.org/doi/10.1103/RevModPhys.90.021001

[71] M. Brzezińska, M. Bieniek, T. Woźniak, P. Potasz, and A. Wójs, "Entanglement entropy and entanglement spectrum of bi1-xSbx(1 1 1) bilayers", *Journal of Physics: Condensed Matter*, vol. 30, no. 12, p. 125501, feb 2018. doi: 10.1088/1361-648x/aaaf54. [Online]. Available: https://doi.org/10.1088%2F1361-648x%2Faaaf54

[72] A. Splendiani, L. Sun, Y. Zhang, T. Li, J. Kim, C.-Y. Chim, G. Galli, and F. Wang, "Emerging photoluminescence in monolayer mos2", *Nano Letters*, vol. 10, no. 4, pp. 1271–1275, 2010. doi: 10.1021/nl903868w PMID: 20229981. [Online]. Available: https://doi.org/10.1021/nl903868w

[73] K. F. Mak and J. Shan, "Photonics and optoelectronics of 2d semiconductor transition metal dichalcogenides", *Nature Photonics*, vol. 10, p. 216–226, mar 2016. doi: 10.1038/nphoton.2015.282. [Online]. Available: https://doi.org/10.1038/nphoton.2015.282

[74] J. R. Schaibley, H. Yu, G. Clark, P. Rivera, J. S. Ross, K. L. Seyler, W. Yao, and X. Xu, "Valleytronics in 2d materials", *Nature Reviews Materials volume*, vol. 1,

no. 16055, August 2016. doi: 10.1038/natrevmats.2016.55. [Online]. Available: https://doi.org/10.1038/natrevmats.2016.55

[75] K. F. Mak, X. D., and J. Shan, "Light–valley interactions in 2d semiconductors", *Nature Photonics*, vol. 12, p. 451–460, June 2018. doi: 10.1038/s41566-018-0204-6. [Online]. Available: https://doi.org/10.1038/s41566-018-0204-6

[76] L. Mennel, J. Symonowicz, S. Wachter, D. K. Polyushkin, A. J. Molina-Mendoza, and T. Mueller, "Ultrafast machine vision with 2d material neural network image sensors", *Nature*, vol. 579, p. 62–66, mar 2020. doi: 10.1038/s41586-020-2038-x. [Online]. Available: https://doi.org/10.1038/s41586-020-2038-x

[77] C. Gong, L. Li, Z. Li, H. Ji, A. Stern, Y. Xia, T. Cao, W. Bao, C. Wang, Y. Wang, Z. Q. Qiu, R. J. Cava, S. G. Louie, J. Xia, and X. Zhang, "Discovery of intrinsic ferromagnetism in two-dimensional van der waals crystals", *Nature*, vol. 546, pp. 265–169, April 2017. doi: 10.1038/nature22060. [Online]. Available: https://doi.org/10.1038/nature22060

[78] B. Huang, G. Clark, E. Navarro-Moratalla, D. Klein, R. Cheng, K. L. Seyler, D. Zhong, E. Schmidgall, M. A. McGuire, D. H. Cobden, W. Yao, D. Xiao, P. Jarillo-Herrero, and X. Xu, "Layer-dependent ferromagnetism in a van der waals crystal down to the monolayer limit", *Nature*, vol. 546, p. 270–273, June 2017. doi: 10.1038/nature22391. [Online]. Available: https://doi.org/10.1038/nature22391

[79] X. Wang, K. Du, Y. Y. F. Liu, P. Hu, J. Zhang, Q. Zhang, M. H. S. Owen, L. X., C. K. Gan, P. Sengupta, C. Kloc, and Q. Xiong, "Raman spectroscopy of atomically thin two-dimensional magnetic iron phosphorus trisulfide (FePS 3) crystals", *2D Materials*, vol. 3, no. 3, p. 031009, aug 2016. doi: 10.1088/2053-1583/3/3/031009. [Online]. Available: https://doi.org/10.1088%2F2053-1583%2F3%2F3%2F031009

[80] J.-U. Lee, S. Lee, J. H. Ryoo, S. Kang, T. Y. Kim, P. Kim, C.-H. Park, J.-G. Park, and H. Cheong, "Ising-type magnetic ordering in atomically thin feps3", *Nano Letters*, vol. 16, no. 12, pp. 7433–7438, 2016. doi: 10.1021/acs.nanolett.6b03052 PMID: 27960508. [Online]. Available: https://doi.org/10.1021/acs.nanolett.6b03052

[81] Y. Ma, Y. Dai, M. Guo, C. Niu, Y. Zhu, and B. Huang, "Evidence of the existence of magnetism in pristine vx2 monolayers (x = s, se) and their strain-induced tunable magnetic properties", *ACS Nano*, vol. 6, no. 2, pp. 1695–1701, 2012. doi: 10.1021/nn204667z PMID: 22264067. [Online]. Available: https://doi.org/10.1021/nn204667z

[82] B. Sachs, T. O. Wehling, K. S. Novoselov, A. I. Lichtenstein, and M. I. Katsnelson, "Ferromagnetic two-dimensional crystals: Single layers of k_2cuf_4", *Phys. Rev. B*, vol. 88, p. 201402, Nov 2013. doi: 10.1103/PhysRevB.88.201402. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevB.88.201402

[83] N. Sivadas, M. W. Daniels, R. H. Swendsen, S. Okamoto, and D. Xiao, "Magnetic ground state of semiconducting transition-metal trichalcogenide monolayers", *Phys. Rev. B*, vol. 91, p. 235425, Jun 2015. doi: 10.1103/PhysRevB.91.235425. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevB.91.235425

[84] B. L. Chittari, Y. Park, D. Lee, M. Han, A. H. MacDonald, E. Hwang, and J. Jung, "Electronic and magnetic properties of single-layer $m\mathrm{P}X_3$ metal phosphorous trichalcogenides", *Phys. Rev. B*, vol. 94, p. 184428, Nov 2016. doi: 10.1103/PhysRevB.94.184428. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevB.94.184428

[85] M. Bonilla, S. Kolekar, Y. Ma, H. C. Diaz, V. Kalappattil, R. Das, T. Eggers, H. R. Gutierrez, M.-H. Phan, and M. Batzill, "Strong room-temperature ferromagnetism in vse2 monolayers on van der waals substrates", *Nature Nanotechnology*, vol. 546, p. 265–269, April 2017. doi: 10.1038/nature22060. [Online]. Available: https://doi.org/10.1038/nature22060

[86] C. Tan, J. Lee, S.-G. Jung, T. Park, S. Albarakati, J. Partridge, M. R. Field, D. G. McCulloch, L. Wang, and C. Lee, "Magnetic ground state of semiconducting transition-metal trichalcogenide monolayers", *Nat. Commun.*, vol. 9, p. 1554, Apr 2018. doi: 10.1038/s41467-018-04018-w. [Online]. Available: https://doi.org/10.1038/s41467-018-04018-w

[87] Z. Lin, J.-H. Choi, Q. Zhang, W. Qin, S. Yi, P. Wang, L. Li, Y. Wang, H. Zhang, Z. Sun, L. Wei, S. Zhang, T. Guo, Q. Lu, J.-H. Cho, C. Zeng, and Z. Zhang, "Flatbands and emergent ferromagnetic ordering in fe_3sn_2 kagome lattices", *Phys. Rev. Lett.*, vol. 121, p. 096401, Aug 2018. doi: 10.1103/PhysRevLett.121.096401. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevLett.121.096401

[88] M. M. Otrokov, I. P. Rusinov, M. Blanco-Rey, M. Hoffmann, A. Y. Vyazovskaya, S. V. Eremeev, A. Ernst, P. M. Echenique, A. Arnau, and E. V. Chulkov, "Unique thickness-dependent properties of the van der waals interlayer antiferromagnet $mnbi_2te_4$ films", *Phys. Rev. Lett.*, vol. 122, p. 107202, Mar 2019. doi: 10.1103/PhysRevLett.122.107202. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevLett.122.107202

[89] X. Tang, W. Sun, Y. Gu, C. Lu, L. Kou, and C. Chen, "cob_6 monolayer: A robust two-dimensional ferromagnet", *Phys. Rev. B*, vol. 99, p. 045445, Jan 2019. doi: 10.1103/PhysRevB.99.045445. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevB.99.045445

[90] B. L. Chittari, D. Lee, N. Banerjee, A. H. MacDonald, E. Hwang, and J. Jung, "Carrier- and strain-tunable intrinsic magnetism in two-dimensional MAX_3 transition metal chalcogenides", *Phys. Rev. B*, vol. 101, p. 085415, Feb 2020. doi: 10.1103/PhysRevB.101.085415. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevB.101.085415

[91] F. Kargar, E. A. Coleman, S. Ghosh, J. Lee, M. J. Gomez, Y. Liu, A. S. Magana, Z. Barani, A. Mohammadzadeh, B. Debnath, R. B. Wilson, R. K. Lake, and A. A. Balandin, "Phonon and thermal properties of quasi-two-dimensional feps3 and mnps3 antiferromagnetic semiconductors", *ACS Nano*, vol. 14, no. 2, pp. 2424–2435, 2020. doi: 10.1021/acsnano.9b09839 PMID: 31951116. [Online]. Available: https://doi.org/10.1021/acsnano.9b09839

[92] K. Yang, F. Fan, H. Wang, D. I. Khomskii, and H. Wu, "vi_3: A two-dimensional ising ferromagnet", *Phys. Rev. B*, vol. 101, p. 100402, Mar 2020. doi: 10.1103/PhysRevB.101.100402. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevB.101.100402

[93] F. Subhan and J. Hong, "Magnetic anisotropy and curie temperature of two-dimensional VI3 monolayer", *Journal of Physics: Condensed Matter*, vol. 32, no. 24, p. 245803, mar 2020. doi: 10.1088/1361-648x/ab7c14. [Online]. Available: https://doi.org/10.1088%2F1361-648x%2Fab7c14

[94] M. Pizzochero, R. Yadav, and O. V. Yazyev, "Magnetic exchange interactions in monolayer CrI3 from many-body wavefunction calculations", *2D Materials*, vol. 7, no. 3, p. 035005, apr 2020. doi: 10.1088/2053-1583/ab7cab. [Online]. Available: https://doi.org/10.1088%2F2053-1583%2Fab7cab

[95] N. D. Mermin and H. Wagner, "Absence of ferromagnetism or antiferromagnetism in one- or two-dimensional isotropic heisenberg models", *Phys. Rev. Lett.*, vol. 17, pp. 1133–1136, Nov 1966. doi: 10.1103/PhysRevLett.17.1133. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevLett.17.1133

[96] M. Pizzochero, "Atomic-scale defects in the two-dimensional ferromagnet CrI3 from first principles", *Journal of Physics D: Applied Physics*, vol. 53, no. 24, p. 244003, apr 2020. doi: 10.1088/1361-6463/ab7ca3. [Online]. Available: https://doi.org/10.1088%2F1361-6463%2Fab7ca3

[97] L. Webster and J.-A. Yan, "Strain-tunable magnetic anisotropy in monolayer $crcl_3$, $crbr_3$, and cri_3", *Phys. Rev. B*, vol. 98, p. 144411, Oct 2018. doi: 10.1103/PhysRevB.98.144411. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevB.98.144411

[98] F. Zheng, J. Zhao, Z. Liu, M. Li, M. Zhou, S. Zhang, and P. Zhang, "Tunable spin states in the two-dimensional magnet cri3", *Nanoscale*, vol. 10, pp. 14 298–14 303, 2018. doi: 10.1039/C8NR03230K. [Online]. Available: http://dx.doi.org/10.1039/C8NR03230K

[99] S. Jiang, L. Li, Z. Wang, K. F. Mak, and J. Shan, "Controlling magnetism in 2d cri3 by electrostatic doping." *Nature Nanotech*, vol. 13, p. 549–553, 2018. doi: 10.1038/s41565-018-0135-x. [Online]. Available: https://doi.org/10.1038/s41565-018-0135-x

[100] F. Pan and P. Zhang, "Simulating the sycamore quantum supremacy circuits", 2021. [Online]. Available: https://arxiv.org/abs/2103.03074

[101] J. Gray and S. Kourtis, "Hyper-optimized tensor network contraction", *Quantum*, vol. 5, p. 410, Mar. 2021. doi: 10.22331/q-2021-03-15-410. [Online]. Available: https://doi.org/10.22331/q-2021-03-15-410

[102] F. Pan, K. Chen, and P. Zhang, "Solving the sampling problem of the sycamore quantum circuits", *Phys. Rev. Lett.*, vol. 129, p. 090502, Aug 2022. doi: 10.1103/PhysRevLett.129.090502. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevLett.129.090502

[103] T. Ayral, T. Louvet, Y. Zhou, C. Lambert, E. M. Stoudenmire, and X. Waintal, "A density-matrix renormalization group algorithm for simulating quantum circuits with a finite fidelity", Aug. 2022, arXiv:2207.05612 [cond-mat, physics:quant-ph]. [Online]. Available: http://arxiv.org/abs/2207.05612

[104] I. L. Markov, A. Fatima, S. V. Isakov, and S. Boixo, "Quantum Supremacy Is Both Closer and Farther than It Appears", Sep. 2018, arXiv:1807.10749 [quant-ph]. [Online]. Available: http://arxiv.org/abs/1807.10749

[105] B. Villalonga, S. Boixo, B. Nelson, C. Henze, E. Rieffel, R. Biswas, and S. Mandrà, "A flexible high-performance simulator for verifying and benchmarking quantum circuits implemented on real hardware", *npj Quantum Information*, vol. 5, no. 1, p. 86, Oct. 2019. doi: 10.1038/s41534-019-0196-1 ArXiv:1811.09599 [quant-ph]. [Online]. Available: http://arxiv.org/abs/1811.09599

[106] Y. Shi, L. Duan, and G. Vidal, "Classical simulation of quantum many-body systems with a tree tensor network", *Physical Review A*, vol. 74, no. 2, p. 022320, Aug. 2006. doi: 10.1103/PhysRevA.74.022320 ArXiv:quant-ph/0511070. [Online]. Available: http://arxiv.org/abs/quant-ph/0511070

[107] L. Tagliacozzo, G. Evenbly, and G. Vidal, "Simulation of two-dimensional quantum systems using a tree tensor network that exploits the entropic area law", *Physical Review B*, vol. 80, no. 23, p. 235127, Dec. 2009. doi: 10.1103/PhysRevB.80.235127 ArXiv:0903.5017 [cond-mat]. [Online]. Available: http://arxiv.org/abs/0903.5017

[108] V. Murg, Legeza, R. M. Noack, and F. Verstraete, "Simulating Strongly Correlated Quantum Systems with Tree Tensor Networks", *Physical Review B*, vol. 82, no. 20, p. 205105, Nov. 2010. doi: 10.1103/PhysRevB.82.205105 ArXiv:1006.3095 [cond-mat, physics:quant-ph]. [Online]. Available: http://arxiv.org/abs/1006.3095

[109] N. Nakatani and G. K.-L. Chan, "Efficient tree tensor network states (TTNS) for quantum chemistry: Generalizations of the density matrix renormalization group algorithm", *The Journal of Chemical Physics*, vol. 138, no. 13, p. 134113, Apr. 2013. doi: 10.1063/1.4798639. [Online]. Available: http://aip.scitation.org/doi/10.1063/1.4798639

[110] M. Gerster, P. Silvi, M. Rizzi, R. Fazio, T. Calarco, and S. Montangero, "Unconstrained Tree Tensor Network: An adaptive gauge picture for enhanced performance", *Physical Review B*, vol. 90, no. 12, p. 125154, Sep. 2014. doi: 10.1103/PhysRevB.90.125154 ArXiv:1406.2666 [cond-mat, physics:quant-ph]. [Online]. Available: http://arxiv.org/abs/1406.2666

[111] V. Murg, F. Verstraete, R. Schneider, P. R. Nagy, and Legeza, "Tree Tensor Network State with Variable Tensor Order: An Efficient Multireference Method for Strongly Correlated Systems", *Journal of Chemical Theory and Computation*, vol. 11, no. 3, pp. 1027–1036, Mar. 2015. doi: 10.1021/ct501187j. [Online]. Available: https://pubs.acs.org/doi/10.1021/ct501187j

[112] B. Kloss, D. Reichman, and Y. Bar Lev, "Studying dynamics in two-dimensional quantum lattices using tree tensor network states", *SciPost Physics*, vol. 9, no. 5, p. 070, Nov. 2020. doi: 10.21468/SciPostPhys.9.5.070. [Online]. Available: https://scipost.org/10.21468/SciPostPhys.9.5.070

[113] X. Qian and M. Qin, "From Tree Tensor Network to Multiscale Entanglement Renormalization Ansatz", *Physical Review B*, vol. 105, no. 20, p. 205102, May 2022. doi: 10.1103/PhysRevB.105.205102 ArXiv:2110.08794 [cond-mat, physics:quant-ph]. [Online]. Available: http://arxiv.org/abs/2110.08794

[114] G. Vidal, "Entanglement Renormalization", *Physical Review Letters*, vol. 99, no. 22, p. 220405, Nov. 2007. doi: 10.1103/PhysRevLett.99.220405. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevLett.99.220405

[115] G. Evenbly and G. Vidal, "Entanglement Renormalization in Two Spatial Dimensions", *Physical Review Letters*, vol. 102, no. 18, p. 180406, May 2009. doi: 10.1103/PhysRevLett.102.180406. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevLett.102.180406

[116] G. Evenbly, "Foundations and Applications of Entanglement Renormalization", Sep. 2011, arXiv:1109.5424 [cond-mat, physics:quant-ph]. [Online]. Available: http://arxiv.org/abs/1109.5424

[117] J. Hauschild, E. Leviatan, J. H. Bardarson, E. Altman, M. P. Zaletel, and F. Pollmann, "Finding purifications with minimal entanglement", *Physical Review B*, vol. 98, no. 23, p. 235163, Dec. 2018. doi: 10.1103/PhysRevB.98.235163 ArXiv:1711.01288 [cond-mat]. [Online]. Available: http://arxiv.org/abs/1711.01288

[118] G. Evenbly and G. Vidal, "Algorithms for entanglement renormalization", *Physical Review B*, vol. 79, no. 14, p. 144108, Apr. 2009. doi: 10.1103/PhysRevB.79.144108 Publisher: American Physical Society. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevB.79.144108

[119] G. Evenbly and G. Vidal, "Algorithms for entanglement renormalization: boundaries, impurities and interfaces", *Journal of Statistical Physics*, vol. 157, no. 4-5, pp. 931–978, Dec. 2014. doi: 10.1007/s10955-014-0983-1 ArXiv:1312.0303 [cond-mat, physics:quant-ph]. [Online]. Available: http://arxiv.org/abs/1312.0303

[120] A. V. Goldberg, É. Tardos, and R. E. Tarjan, "Network flow algorithms", 1989.

[121] A. Milsted, M. Ganahl, S. Leichenauer, J. Hidary, and G. Vidal, "TensorNetwork on TensorFlow: A Spin Chain Application Using Tree Tensor Networks", May 2019, arXiv:1905.01331 [cond-mat, physics:hep-th, physics:physics, stat]. [Online]. Available: http://arxiv.org/abs/1905.01331

[122] M. Lubasch, J. I. Cirac, and M.-C. Bañuls, "Algorithms for finite Projected Entangled Pair States", *Physical Review B*, vol. 90, no. 6, p. 064425, Aug. 2014. doi: 10.1103/PhysRevB.90.064425 ArXiv:1405.3259 [cond-mat, physics:quant-ph]. [Online]. Available: http://arxiv.org/abs/1405.3259