universität
wien

# DIPLOMARBEIT

"The Density Matrix Renormalization Group on Tensor Networks"

Claude Klöckl

angestrebter akademischer Grad
Magister der Naturwissenschaften (Mag.rer.nat)

Wien, im April 2013
Studienkennzahl lt. Studienblatt: A 405
Studienrichtung lt. Studienblatt: Mathematik
Betreuer: Prof. Arnold Neumaier

**Acknowledgements**

To Tamara, Karim, Gabi, Chris and Marlen.

I want to thank my supervisor Prof. Arnold Neumaier for his support and Dr. Marcus Huber for further corrections and discussions.

# Introduction

The aim of this work is to provide an overview over recent advances in the simulation algorithms for quantum many body problems.

We focus on the class of algorithms that are based on the density matrix renormalization group or short DMRG [32] [20] [21]. It is a procedure originally designed to simulate one dimensional systems so called spin chains. The classical density matrix renormalization group is an extremely fast and precise algorithm, but is surprisingly only suited to simulate one dimensional systems while breaking down even for the two dimensional case. Obviously one would like to be able to have access to similar powerful methods for the much more interesting two or three dimensional case. In this work we will outline the density matrix renormalization group based approach to simulating quantum many body problems and the proposals to generalize the density matrix renormalization group to two or higher dimensional problems.

We do not cover Monte Carlo methods for quantum many body problems. These can be seen as the second big class of algorithms for simulating quantum many body problems. The main drawback of these techniques is that there are problems simulating interesting physical systems. Monte Carlo simulations exhibit the so-called fermionic sign problem when applied to fermions, leading to effort exponential in particle number of Monte Carlo simulations. The problem has been shown to be NP-hard [24]. This makes the density matrix renormalization group approach especially relevant for fermionic systems.

In chapter 1 we start by introducing the reader to the general description of quantum mechanical systems. First we consider in 1.1 the description of an isolated system and then we proceed to describing several interacting systems in 1.2.

Chapter 2 introduces the classical formulation of the density matrix renormalization group ansatz. 2.3 covers the reformulation of the ansatz from a renormalization group to a variational ansatz. This is a more modern viewpoint, that is based on the realization that the density matrix renormalization group ansatz always creates a certain class of states the so-called matrix product states. Finally we will explore in 2.3.7 the issue why the density matrix renormalization group breaks down in two and higher dimensions. We highlight the connection between the spectrum of a state, it's entanglement and the problems we encounter simulating the state in higher dimensions.

Computations with quantum many body problems involve many differently labelled coefficients. A lot of bookkeeping is required to keep track of all them. 3 introduces the tensor network formalism, that allows to graphically represent the coefficient tensors arising in the description of quantum many body problems. The special case of tree tensor networks is especially useful for calculating ground states, reduced density matrices and expectation values of spin systems. We will illustrate these possibilities in 3.5 - 3.6. In 3.4 we outline a possibility to reduce general tensor networks to tree tensor networks and make effort considerations about this process.

We proceed by introducing in detail the multiscale entanglement renormal-

ization ansatz in 4.1 and a short notice on projected entangled pair states 4.2.1 which are the two contemporary simulation algorithms for two or higher dimensional systems. Both are successful in the two dimensional case and are constantly being improved upon.

We close the work with a detailed literature overview in 5, where we discuss by topic articles, that could be of interest.

# Chapter 1

# Physical Basics

## 1.1 Description of a Quantum Mechanical System

Quantum mechanics usually employs the bra-ket notation, where elements of a vector space are written with brackets even if they are not part of a scalar product. We define the notation:

**Convention. 1.1.1.** *(Bra-Ket Notation) In this work $\mathcal{H}$ always denotes a complex separable Hilbert space. We also refer to a complex separable Hilbert space as a* ***state space****.*

*Let $\psi \in \mathbb{C}^n$ be a column vector. It is called a* ***ket*** *and written as $|\psi\rangle$. The conjugate transposed row vector $\psi^*$ is called a* ***bra*** *and written $\langle\psi|$,*

*the scalar product of $\psi$ and $\phi$*

$$\psi^*\phi = \langle\psi|\phi\rangle \in \mathbb{C},$$

*the outer product is*

$$A = \psi\phi^* = |\psi\rangle\langle\phi| \in \mathbb{C}^{n\times n},$$

*the norm of $\phi$:*

$$\|\phi\| := \sqrt{\langle\phi|\phi\rangle}.$$

While in mathematics it is common to denote vectors with lower case Latin letters (i.e.: $x, y, z$), in quantum mechanics states (i.e.: vectors) are usually denoted by Greek letters (i.e.: $\psi, \phi$). Since this convention may read quite unfamiliar for a mathematician, we compare the definition of positive definiteness in standard notation

$$x^*Ax > 0 \text{ for all } \psi \in \mathbb{C} \setminus 0,$$

with Bra-Ket notation

$$\langle\psi|A|\psi\rangle > 0 \text{ for all } \psi \in \mathbb{C} \setminus 0.$$

An analogous notation is used for general Hilbert spaces.

**Definition. 1.1.2.** *A **quantum mechanical system** is described by a pair $(\mathcal{H}, H)$, where $\mathcal{H}$ is a Hilbert space and $H$ is a Hermitian, linear, selfadjoint operator defined on a dense subspace of $\mathcal{H}$. $H$ is called the **Hamiltonian** of the system. The elements of $\mathcal{H}$ are referred to as **pure states** of the system.*

There is a very common convention in physics regarding the notation of basis elements from $\mathcal{H}$, that should be pointed out because it is rather uncommon in mathematics. All possible orthonormal bases of $\mathcal{H}$ are of course related simply by a unitary transformation. We can obviously choose any basis we like to describe $\mathcal{H}$. Physicist often do not state explicitly which basis is chosen, but simply label the basis elements. This allows for arguments using orthonormality of the basis without further specifying the basis.

**Convention. 1.1.3.** *Let $\mathcal{H}$ is a Hilbert space and $0 \leq n \leq dimH - 1 \in \mathbb{N}$. Then $|n\rangle$ denotes the nth basis element of $\mathcal{H}$.*

In the following we will cover the central objects of study in quantum mechanics the states and the observables. We define a state using the so-called density matrix:

**Definition. 1.1.4.** *$\rho \in Lin(\mathcal{H})$ is a **density matrix**, if the following conditions hold:*

- *$\rho = \rho^\dagger$ (Hermitian)*
- *$tr\rho = 1$ (normalization)*
- *$\langle\psi|\rho|\psi\rangle \geq 0$ for all $\psi \in \mathcal{H}$ (positive semidefiniteness)*

*A state $\rho$ is a **pure state** if $\rho$ has rank 1 and can be written as $\rho = \psi\psi^*$ with $\|\psi\| = 1$. A state $\rho$ is a **mixed state** if it is not pure.*

Defining a state via the density matrix is the most general approach to defining the state. It is however also common to first define only a part of all possibly interesting states, the so called pure states $\psi$. In a second step we also allow convex mixtures of pure state density matrices so called mixed states $\sum_i p_i|\psi_i\rangle\langle\psi_{i|}$, requiring $0 \leq p_i \leq$ and $\sum_i p_i = 1$ turning them into probabilities.

Observables describe interesting experimentally measurable properties of a system. The Hamiltonian is our first example of an observable. In general we define an observable as follows.

**Definition. 1.1.5.** *Let $\mathcal{H}$ be a Hilbert space. An observable $O$ is a Hermitian, linear, selfadjoint operator defined on a dense subspace of $\mathcal{H}$.*

*The **expectation value** of an observable $O$ with respect to $\rho$ is defined by*

$$\langle O\rangle_\rho := tr\rho O.$$

For pure states we express the expectation value $\langle O \rangle$ in terms of the state vector:

**Lemma. 1.1.6.** *Let $O$ be an observable, $|\psi\rangle$ a vector and $\rho$ its density matrix then*

$$\langle O \rangle_\rho = \langle \psi | O | \psi \rangle.$$

***Proof:*** *Recall the property of the trace:* $\operatorname{tr} xy^T = \sum_i x_i y_i = \langle x|y \rangle$. *This yields:* $\operatorname{tr} \rho O = \operatorname{tr} |\psi\rangle\langle\psi| O = \langle \psi | O | \psi \rangle$.

There are many possible observables. The following examples would be typical observables for a system at rest.

- Let $\mathcal{H} = \mathbb{C}^2$. The **spin operator** $\hat{S} = (S_x, S_y, S_z)$, where $S_i = \frac{\hbar}{2}\sigma_i$ and $\sigma_i$ denotes the **Pauli matrices** $\sigma_x = \left(\begin{smallmatrix} 0 & 1 \\ 1 & 0 \end{smallmatrix}\right)$,$\sigma_y = \left(\begin{smallmatrix} 0 & -i \\ i & 0 \end{smallmatrix}\right)$,$\sigma_z = \left(\begin{smallmatrix} 1 & 0 \\ 0 & -1 \end{smallmatrix}\right)$.
- Let $\mathcal{H} = \mathbb{C}^2 \otimes ... \otimes \mathbb{C}^2$ and $s_j \in \mathbb{C}^2$. A sum of spin operator $\sum_j S_{z_j}$, where $S_{z_j}$ is the spin operator measuring the direction $z$ in the $j$th subsystem. This evaluates the spin at several sites $s_j$ and sums over the results. This is relevant for studying quantities such as magnetization, which is essentially the sum over all spins in a system.
- The Hamiltonian $H$.
- The number of particles of type $j$ $N_j$.

Since the expectation value of the Hamiltonian is of special interest we will give it a name.

**Definition. 1.1.7.** *The **energy** of a state $\rho$ is*

$$E(\rho) := \langle \rho H \rangle.$$

We introduce in the following quantities which are of interest in physics.

**Definition. 1.1.8.** *A **ground state** is defined as one eigenvector of the minimal eigenvalue of the Hamiltonian $H$.*

There are cases where Hamiltonians exhibit a degenerate spectrum. These cases are sometimes studied as well, but often unwanted. If we want to exclude the possibility of non unique ground states, we have to make an extra assumption.

**Definition. 1.1.9.** *A Hamiltonian is **non-degenerate** if the smallest eigenvalue $\lambda_0$ has multiplicity 1.*

In quantum mechanics the energy of a system at rest can take only discrete values, so called **energy levels**. These are the eigenvalues of $H$. The eigenvector eigenvalue pairs describe the states attaining these energy levels and the associated energy. The state of lowest energy is usually of special interest.

A central concept from statistical mechanics which often needs to be calculated in physics, is the canonical partition function.

**Definition. 1.1.10.** *Let $E_j$ be the energy levels or eigenvalues of a systems $(\mathcal{H}, H)$ Hamiltonian, $T > 0$ and $k$ the fixed Boltzmann constant. We define the* ***canonical partition function at temperature*** *$T$ by*

$$Z = Z(T) := \sum_j e^{-\frac{E_j}{kT}}.$$

The reason this is interesting is because the probability of finding a system in a state of given energy level $E_j$ is defined in statistical mechanics as

$$P_j = \frac{1}{Z} e^{-E_j/kT}.$$

The division by $Z$ ensures that this is a probability distribution. This basically states that high energy states are far more unlikely than low energy states. All interesting quantities in equilibrium statistical mechanics can be derived from the partition function.

In statistical mechanics a system is assumed to be in contact with a much larger system the environment or heat bath. Several assumption are made regarding which parameters of system are to be seen as fixed and which can be exchanged with the heat bath. A system which can exchange energy, but has fixed temperature, volume and particle number is described by the canonical partition function. Assuming other combinations of parameters to be fixed or exchangeable results in a different partition function.

## 1.2 Composite Systems and Entanglement

We have already given the definition of a quantum mechanical system. If we have descriptions of different systems we might be interested in the interaction of these systems. The description of both systems should again be a system, although a larger one. The operation describing the process of finding a composite system is the tensor product.

**Definition. 1.2.1.**

- *Let us be given one quantum mechanical system $A$ of dimension $r$ and another system $B$ of dimension $s$. They are described by the Hilbert spaces $\mathcal{H}_A$ and $\mathcal{H}_B$. Then the* ***composite system*** *$AB$ is described by the Hilbert space*

  $$\mathcal{H}_{AB} := \mathcal{H}_A \otimes \mathcal{H}_B.$$

  *of dimension $rs$.*

- *As in the case of a single system the* ***basis of the composite system*** *is often not chosen explicitly. Often we write $|i_A j_B\rangle$ to denote $ij$-th basis element of $\mathcal{H}_{\mathcal{AB}}$, which is exactly the tensor product of the $i$-th element of $\mathcal{H}_{\mathcal{A}}$ and the $j$-th element of $\mathcal{H}_{\mathcal{B}}$.*

- *If $A, B$ are in the state $\rho_A$ and $\rho_B$ the* ***composite system's state*** *is $\rho_{AB}$ acting upon $\mathcal{H}_A \otimes \mathcal{H}_B$.*

- *The general case with $n$ systems $s_1, ..., s_n$ and their respective Hilbert spaces $\mathcal{H}_{s_1} \otimes ... \otimes \mathcal{H}_{s_n}$ of dimensions $d_{s_1,...,d_{s_n}}$ is analogously described by their* ***composite system's Hilbert space***

$$\mathcal{H}_{s_1...s_n} := \mathcal{H}_{s_1} \otimes .... \otimes \mathcal{H}_{s_n},$$

*of dimension $d_{s_1...d_{s_n}}$ and the* ***composite system's state*** *$\rho_{s_1...s_n}$ is acting on $\mathcal{H}_{s_1...s_n}$.*

On the other hand we might have a too complex description of a system, while we are only interested in a subsystem of the given system. We need an operation describing this simplification. This will be a generalization of the trace, the partial trace. Reducing a larger system to a subsystem is sometimes called tracing out the environment.

**Definition. 1.2.2.** *Let $AB$ be a composite system and a state $\rho_{AB} \in \mathcal{H}_{AB}$. $|k\rangle_B$ is the $k$th basis element of the Hilbert space describing $B$.*

- *We define the* ***partial trace*** *over subsystem $B$ by*

$$\begin{aligned} &\mathrm{tr}_B : Lin(\mathcal{H}_\mathcal{A} \otimes \mathcal{H}_\mathcal{B}) \to Lin(\mathcal{H}_\mathcal{A}) \\ &\mathrm{tr}_B(\rho_{AB}) := \sum_{k=1}^{dim(B)} \langle k_B|\rho_{AB}|k_B\rangle. \end{aligned}$$

- *The state of $A$ can be found by applying the partial trace over $B$ to $\rho_{AB}$,*

$$\rho_A = \mathrm{tr}_B(\rho_{AB}).$$

**Definition. 1.2.3.** *Let $A, B$ be quantum mechanical systems with associated Hilbert spaces $\mathcal{H}_A, \mathcal{H}_B$.*

- *We call $\rho_{AB} \in \mathcal{H}_A \otimes \mathcal{H}_B$ a* ***separable state*** *if there exist pure states $(\rho_A)_k \in \mathcal{H}_A, (\rho_B)_k \in \mathcal{H}_B$,$\sum_i p_i = 1$ with $\rho_{AB} = \sum_k p_k (\rho_A)_k \otimes (\rho_B)_k$*

- *We call $\rho_{AB}$* ***entangled*** *if it is not separable.*

*The case for $n$ systems is again analogous.*

In the above definition, a system is either entangled or not entangled. It is however useful to quantify entanglement. In order to compare two entangled systems, we define so-called entanglement measures. They allow us to say that a system is more or less entangled then another one.

**Definition. 1.2.4.**

- *Let $\rho$ be a state of a quantum mechanical system, $\lambda_i$ the eigenvalues of $\rho$. We define the entropy $S(\rho)$ of a system in state $\rho$ by*

$$S(\rho) := \sum_i \lambda_i \log_2(\lambda_i).$$

- *Let $\rho_{AB}$ be the state of a pure composite bipartite quantum system $AB$. We define the entanglement entropy $\mathcal{E}(.)$ of the state $\rho_{AB}$ by*

$$\mathcal{E}(\rho_{AB}) := S(\rho_A) = S(\rho_B),$$

*where $\rho_A = \mathrm{tr}_B(\rho_{AB})$ and $\rho_B = \mathrm{tr}_A(\rho_{AB})$.*

Applying the partial trace to the Schmidt decomposition of $\rho_{AB}$ yields that the spectra of the reduced density matrices are identical, hence the equality $S(\rho_A) = S(\rho_B)$ holds.

The entanglement of entropy $\mathcal{E}$ is only defined for bipartitions of a pure composite system. It is interpreted as fixing a subsystem of $AB$ and describing the entanglement between the fixed subsystem and the remainder of the system. It is a property of the whole system $AB$! This definition only considers the two parts of a fixed bipartition. Choosing a different bipartition of the same system may result in a different entanglement entropy.

We make a few comments to show that the above definition is plausibly quantyfing entanglement. Let us assume $\rho_{AB}$ is separable and pure thus $\rho_{AB} = \rho_A \otimes (\rho_B$. Recall that $(\rho_.)_k$ are as reductions of pure state again pure and that pure states have only a single eigenvector, which has to have eigenvalue 0. It follows from the concavity of $S$ that the entropy of an entangled state is 0.

We will see at the end of the next chapter that the amount of entanglement, as defined by the entanglement entropy, is indeed connected to the performance of a simulation algorithm for spin chains.

# Chapter 2

# Density Matrix Renormalization Group

We will discuss in this chapter the density matrix renormalization, often abbreviated as DMRG, algorithm. It was developed to study various models from statistical physics such as the Ising model or the Heisenberg model. These models are often employed to model the magnetic behaviour of a wire, surface or volume. The objects of study are discretized approximations of the models. When interaction rules between neighbouring sites are implemented, one can study the resulting behaviour of the system. One of the central features of the approximation value is that it depends strongly on the size of the discretized lattice. Larger lattices yield a better approximation value. Often one is interested in properties at the so-called **thermodynamic limit**, which is achieved if the lattice size $L$ goes to infinity. Many of the models from statistical physics pose a serious challenge for numerical computation if the lattice size is chosen big enough.

The predecessor of DMRG is the renormalization group method introduced by Wilson [33] trying to improve the tradeoff between lattice size and effort. The basic idea is to divide an initially big lattice into small sectors and approximating each sector through a single site, which is set to the mean value of the sector. Thereby a more coarse and more smaller grid is obtained, which reflects the properties of the bigger one.

The density matrix renormalization group was a refinement of earlier renormalization methods proposed by White [32]. His method was very successful in the simulation of one dimensional systems, so called spin chains, being quick as well as accurate. Unfortunately the generalization to higher dimensions encounters certain fundamental problems which will be covered in the Section 2.3 on matrix product states. One possible generalization attempt is to consider a tree structure instead of a line. We will focus in this work mostly on giving an overview over this line of thought. Meanwhile there have been certain other attempts which have been successful to a certain degree, where we will give a

short overview in Section 4.2.

Initially DMRG has been described as a renormalization group method, but meanwhile other descriptions with slightly different viewpoints have come up. Such as regarding DMRG as a nonlinear block-Gauss-Seidel method see Espig et al. [5] It has also been reformulated as a variational problem, see Section 2.3.6.

We now introduce the classical one dimensional density matrix renormalization group.

One can distinguish two algorithms in DMRG. First we have infinite-DMRG which creates iteratively longer systems. Second there is finite-DMRG which is run after a desired length of the chain is attained. It improves our approximation quality iteratively until some sort of convergence is reached.

## 2.1 Description of Quantum Mechanical System by Spin Chains

We set the stage by introducing spin models on lattices, these are the objects we want to simulate. Recall that a general quantum mechanical system is described by a pair $(\mathcal{H},H)$, where $\mathcal{H}$ is a Hilbert space and $H$ a Hamiltonian. In this section we give the explicit form these two objects take, when describing systems by spin chains and introduce some notation which is needed for the DMRG algorithms such as defining chains, blocks and reduced blocks.

### 2.1.1 The Hilbert space $\mathcal{H}$ of a Spin Model

We start by defining a microscopic system. It's behaviour should be well understood and their interaction will be described later on by the Hamiltonian. We refer to these in future as sites. These systems can be thought of as the building blocks of a macroscopic system.

**Definition. 2.1.1.**

- *A **site** $s$ is described by a $d$-dimensional Hilbert space $\mathcal{H}$. The state of a site is an element $\psi_s \in \mathcal{H}$. Sites are denoted by lowercase letters.*
- *$d$ is the **dimension of the local basis** of a site.*

Recall that by $\mathcal{H}$ we always refer to a complex separable Hilbert space. In the case of a spin system one usually chooses $\mathcal{H} = \mathbb{C}^2$ at each site.

Now we introduce a macroscopic one dimensional object. This is what we want to study. All macroscopic objects in this text are lattices. In this section we consider the one dimensional special case. We reserve the name chain for that special case.

**Definition. 2.1.2.**

- *A **spin model's Hilbert space** is described by the tensor product of several sites. In the context of DMRG we regard one dimensional spin models or also called **spin chains**. Chains are denoted by uppercase letters i.e. $SE$. The state of a chain is an element $\psi_{SE} \in \underbrace{\mathcal{H} \otimes ... \otimes \mathcal{H}}_{Ltimes}$.*
- *We call the total **chain length** $L$. Frequently we will indicate the chain length in superscript brackets by $SE^{(L)}$. We label the sites of the chain from left to right as $s_1, ..., s_L$. Without loss of generality we set $s_1$ as the leftmost site of the chain. For simplicity we assume* $\dim s_i = \dim s_j$.

### 2.1.2 The Hamiltonian $H$ of a Spin Model

**Definition. 2.1.3.** *Let us be given a composite system $S$ consisting of $d$ dimensional subsystems $s_i$. The **Hamiltonian of a spin model** $S$ is*

$$H := \sum_i H_i + \sum_{ik} H_{ik},$$

*where $H_i$ is the Hamiltonian of site $s_i$ and $H_{i,k}$ is the **two body interaction between** $i, k$. The amount of $H_{ik} \neq 0$ is an important modelling choice! Usually all adjacent $H_{ik}$ should be non zero. In a $d$ dimensional spin model all sites have $2d$ neighbours. We refer to one dimensional spin models as **spin chains**. The difference between a chain of nine sites and a $3 \times 3$ grid is defined solely by these interaction terms! Often all other $H_{ik}$ are set to zero which would describe **local or short range interactions**. If there are non zero, non next neighbour interaction terms we talk of **long range interactions**.*

Here we have made the common assumption that we can decompose the Hamiltonians into one site terms and two body interactions. This assumption is not only common in the literature it will also prove to be beneficial later on, because it allows for significant effort reductions, when employing tree tensor networks we will encounter in 3.

### 2.1.3 DMRG Jargon: Blocks, reduced Blocks and Truncation Error

DMRG uses a chain divided into two blocks. One part should describe the state of our system. This part of the chain is usually called the **system block** and denoted as $S$.

Not the whole part of the chain will be devoted to describe this state though. All unneglected outside influences are summed up in the remainder of the chain and referred to as the **environment block** $E$. Usually the construction of the environment depends highly on the problem setting. Most papers specificly state how they choose to build the environment, there is no general construction process.

**Definition. 2.1.4.**

- *A **block** is described by the tensor product of a subset of the sites creating a chain of length $L$. Blocks are denoted by capitals as well i.e. $S$,$E$. The state of a block $\phi$ is acting on $\underbrace{\mathcal{H} \otimes ... \otimes \mathcal{H}}_{ltimes}$ with $l < L$.*

- *We call the **block length** $l$. Analogous to whole chains we will sometimes write the block length in superscript by $S^{(l)}$.*

An exact representation of a chain would require $d^L$ basis elements, which is clearly too large for big $L$. Instead DMRG determines only the $D$ most significant basis elements and ignores the remaining ones of a block. We call that truncating the basis.

**Definition. 2.1.5.**
*Let $S$ be a chain with chain length $L$ and local basis dimensions $d$.*

- *$D$ is the **reduced dimension** of a block after truncation, we need $D << d^L$ and have to choose $D$ small enough to be computationally feasible.*

- *The **reduced chain** $\tilde{S}$ of a chain $S$ is a chain, with Hilbert space $\tilde{\mathcal{H}}$ satisfying $\dim(\tilde{\mathcal{H}})=D$ and $\tilde{\mathcal{H}} \subset \underbrace{\mathcal{H} \otimes ... \otimes \mathcal{H}}_{Ltimes}$ and the same Hamiltonian as the original chain $S$.*

Computationally feasible in this context means that we should choose our $D$ as large as possible while still being able to diagonalize a $D \times D$ matrix. This is done by means of the Lanczos algorithm or any other diagonalization method of choice. The diagonalization should be possible quickly enough to be iterated many times, since we will perform a new diagonalization in every step of DMRG!

Note that in the literature there are various notations for the reduced dimension, other common choices are $M$ or $\chi$. In general the notation varies significantly and should always be checked.

DMRG makes a compromise between accuracy and effort by neglecting all but the $D$ lowest eigenvalues. Therefore one can not expect exact accuracy.

**Definition. 2.1.6.** *Let $\rho$ the density matrix of a state generated by DMRG. We order the eigenvalues $\lambda_i$ of $\rho$ ascending as $\lambda_1 \leq .... \leq \lambda_n$. We define*

$$\epsilon_{trunc} := 1 - \sum_{i=D+1}^{n} \lambda_i.$$

*We call $\epsilon_{trunc}$ the **truncation error**.*

This is the theoretical limit of accuracy we can hope for.

## 2.2 Classical DMRG

The aim of the density matrix renormalization group is to simulate a spin chain in the thermodynamic limit. This is of course not possible directly, because we can only simulate finitely many sites. Instead we want to generate a spin chain with enough sites, so that we deem it as a good approximation to the thermodynamic limit. We fix some chain length $L$, that is sufficiently large. We will now introduce two algorithms **infinite DMRG** and **finite DMRG**.

A simulation begins by running the infinite DMRG algorithm. It starts with a small exactly diagonalizable spin chain and creates in every iteration a by two sites longer chain. We run this algorithm until we have reached a chain of length $L$, that should be able to approximate the thermodynamic limit well. However infinite DMRG tends to produce approximations that are not optimal. Infinite DMRG creates in every iteration two new sites. The newly created sites have relatively little iterations to interact with the remaining system and it can happen that they do not adapt fast enough to approximate the system well. For this reason finite DMRG is run afterwards, which usually decreases the approximation error of infinite DMRG down to nearly the truncation error.

Finite DMRG keeps the chain length constant but increases in every iteration the approximation value of the chain. We iterate until no changes in the system are observed anymore. We initialize finite DMRG with the spin chain obtained by infinite DMRG. Keep that the shorter chains calculated by infinite DMRG are also reused by finite DMRG and need to be stored!

The following two algorithms use the definitions for Hilbert spaces and Hamiltonians of spin models and DMRG jargon given in 2.1.1,2.1.2 and 2.1.3.

### 2.2.1 infinite DMRG

Let $D$ be the maximal dimension we can diagonalize exactly, we call it the **truncated dimension**.

- We start with a so-called **system block** or system of size $l$, labelled $S^{(l)}$. The initial choice is quite arbitrary! For example a single random spin is a valid initial choice or any system gained by exact diaoganalization. We add a new site $s_S$ with full degrees of freedom $d$ resulting in a new block $S^{(l+1)} = S^{(l)} \otimes s_S$.

- Analogously we start with a second block $E$ of length $l$ called the **environment**. There is no single way to choose the environment, that depends on the application. It should capture the interaction of the system with the rest of the chain. A simple way for reflection symmetric Hamiltonians is simply copying $S$. We again add a site $s_E$ of full degree forming $E^{(l+1)} = E^{(l)} \otimes s_E$.

- We now form a so-called superblock $SE^{(2l+2)} := S^{(l)} \otimes E^{(l)}$ of length $2l+2$ and determine and store its lowest eigenvector and eigenvalue pairs. This

is done using Lanczos algorithm. This step accounts for the main effort in the algorithm.

- By now $\dim(S^{(l+1)}) = \dim(S^l) * d$. In the first iteration we have $\dim(S^{(l)}) \leq d^l$ in later iterations $\dim(S^{(l)}) \leq d^l$. This is too much since the dimension of the block $S^{(l+1)}$ should at most be be $D$!

  In order to restrict the effort of the algorithm, we project $S^{(l+1)} \otimes E^{(l+1)}$ on the basis spanned by the $D$ lowest eigenvectors of the superblock. The projection can be achieved by a truncated singular value decomposition. This yields the reduced chain $\tilde{S}^{(l+1)} \otimes \tilde{E}^{(l+1)}$! This step keeps the size of the basis constant, while we continue adding new sites to the chain.

- We separate the reduced chain into blocks $\tilde{S}^{(l+1)}, \tilde{E}^{(l+1)}$ have a new length of $l + 1$. They can be used as blocks for the next iteration.

- Repeat until $L$ large enough

### 2.2.2 finite DMRG

It should be noted that this phase can be left out, but then our simulation will be a worse approximation. Therefore it might be worthwhile to consider using symmetries or other structural information and take care in calculating the environment of infinite DMRG.

- We start with a given 1d lattice of length $L$. It is the result of infinite-DMRG. Initially we set the two block sizes equally as $l_S = l_E = L/2$. One block will grow by one site with each iteration, while the other shrinks until it becomes diagonalizable. We chose a growing direction $n = \pm 1$

- We separate the lattice into two parts, a system block $S^{(l_S)}$ of size $l_S$ and an environment block $E^{(l_E)}$ and size $l_E$

- We replace one lattice site at the border of each block by a site with full degree of freedom $d$. This usually increases the dimension of the block, since DMRG truncates it's basis and not all basis elements of a site are kept! We label the sites we insert $s_S, s_E$. Note that in this step the border of a block depends on whetever we use open or periodic boundary conditions. Typically one would choose two adjacent sites $s_S = s_{l_s}$,$s_E = s_{l_s+1}$ in the middle of the chain for open boundary conditions. For periodic boundary conditions one would rather choose $s_S = s_1$ and $s_E = s_{l_s+1}$ to separate the blocks from both sides.

- We call the block to which $s_{l_s-n}$ belongs the **growing block** and the other one the **shrinking block**. We run the Lanczos algorithm determining the $D$ lowest eigenvalues and eigenvectors of the growing block. Then we project the superblock $S^{(l_S+1)} \otimes E^{(l_E+1)}$ onto a basis consisting of the newly determined $D$ eigenvectors of the growing block. We obtain a

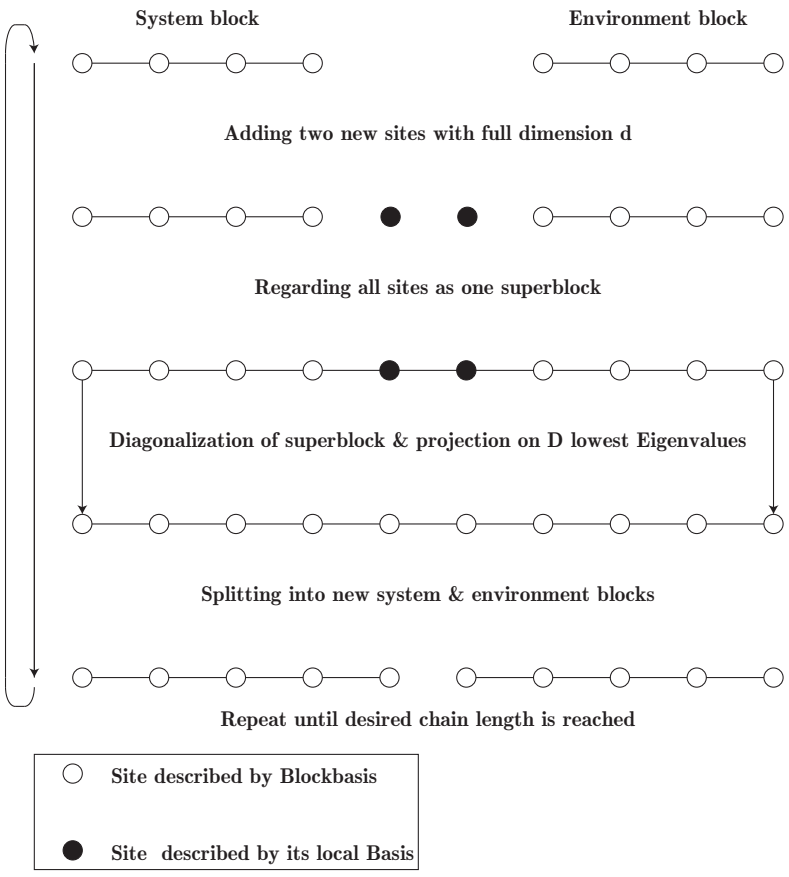


Figure 2.1: Graphical representation of one step of the infinite DMRG algorithm

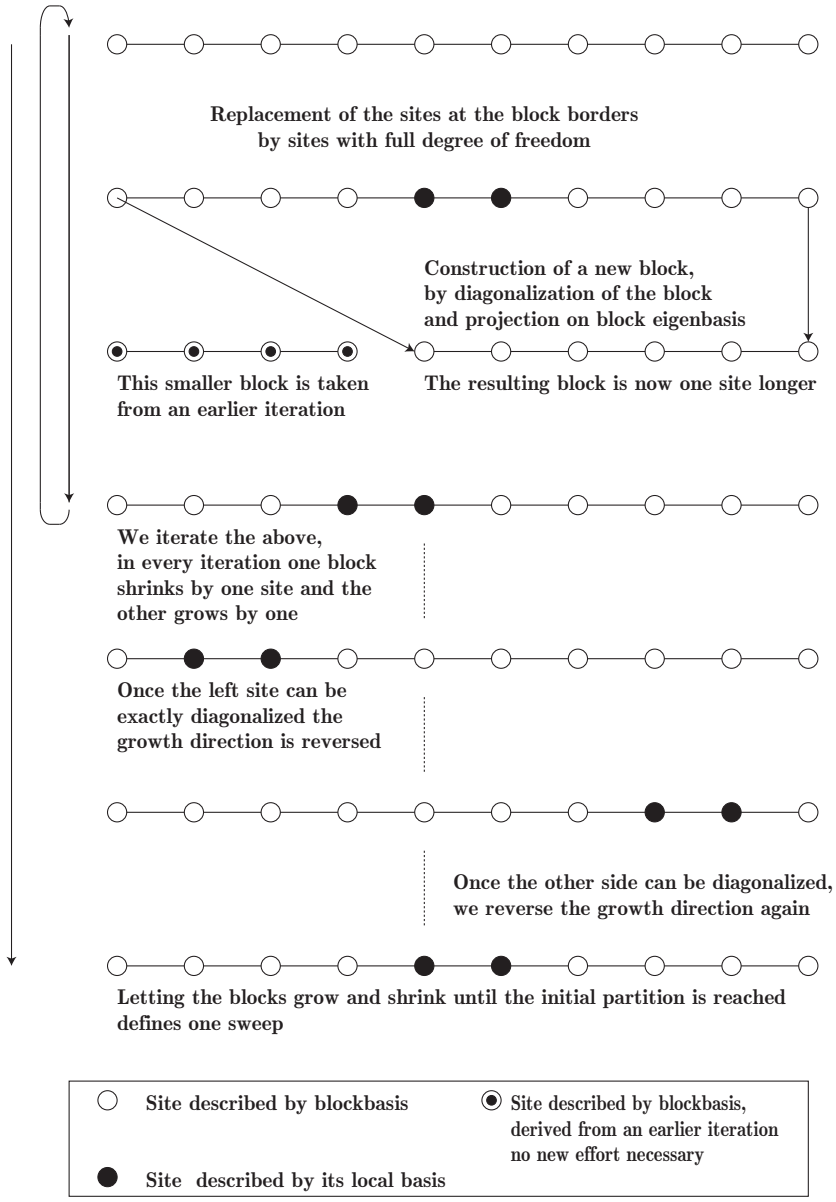


Figure 2.2: Graphical representation of one sweep of the finite DMRG algorithm

reduced growing block $\tilde{S}^{(l+1)}$ or $\tilde{E}^{(l+1)}$. Now we proceed slightly different for finding new blocks $S^{(l_S+1)}$ and $E^{(l_E+1)}$.

In contrast we do not make any calculations regarding the shrinking block! We simply reuse one of the previously stored blocks of appropriate length. Initially the blocks calculated by infinite DMRG are reused. Later sweeps simply use old blocks calculated in prior finite DMRG sweeps. Therefore we will keep the growing block before projecting it onto its eigenbasis. We can overwrite one of the old stored blocks, they will not be used anymore and save some storage.

- If the smaller block is small enough to be exactly diagonalizable multiply growing direction $n$ with $-1$.
- set $l_S = l_S + n$, $l_E = l_E - n$
- repeat from the second step until some kind of convergence is reached

The key difference to the earlier algorithm is that the total lattice length $L$ remains constant. Furthermore in each step one of the blocks grows, while the other block shrinks by one site. This continues until the smaller black becomes exactly diagonalizable, then the process is reversed. One sweep is determined by reversing the direction of shrinking two times and reaching equal block length again.

## 2.3 Matrix Product States

### 2.3.1 Motivation

The DMRG method was very well suited to simulate one dimensional systems, while on the other hand higher dimensional systems are problematic. The aim of this section is to explain this phenomenon.

This section follows *Schollwöck* [21], who provides an extensive review article entirely devoted to the connection between DMRG and matrix product states .

### 2.3.2 Construction of Matrix Product States

**Definition. 2.3.1.** *We call a pure state $\phi$ defined on a lattice with $L$ sites a* ***matrix product state (MPS)*** *if we can express it as*

$$|M\rangle := \sum_{s_1=1\ldots s_L=1}^{d_1,\ldots,d_L} M_{s_L}^{(L)} M_{s_{L-1}}^{(L-1)} \ldots M_{s_2}^{(2)} M_{s_1}^{(1)} |s_1 \ldots s_L\rangle,$$

*with $s_1, \ldots, s_L$ denoting the sites from 1 to $L$, $(|s_i\rangle)_{1\leq i\leq d_i}$ the basis of $s_i$ and $d_i$ the local dimension of the basis at a site $s_i$. $M$ being $d_i \times d_{i+1}$ matrices of coefficients.*

### 2.3.3 General States expressed as Matrix Product States

A general pure state $\phi$ is described by its coefficient tensor. The following argument is mainly about reshaping this coeffcient tensor in suitable ways to derive a matrix product state. We have to introduce some notation distinguishing the various forms in which we will express the coefficient tensor.

**Definition. 2.3.2.**

- $C \in \mathcal{H}^d \otimes ... \otimes \mathcal{H}^d$ *is a* ***coefficient tensor*** *and* $C_{i_1,...,i_L} \in \mathbb{C}$ *with* $1 \leq i_l \leq d$ *are coeffcients. Here we have made the assumption that all Hilbert spaces* $\mathcal{H}^d$ *have the same local dimension* $d$ *to simplify notation.*

- $C^{(1)} \in \mathbb{C}^{d \times d^{L-1}}$ *is the tensor* $C$ *expressed in* ***coefficient matrix*** *form. We construct* $C^{(1)}$ *by putting all the coefficients corresponding to the* $i$*th basis element of the first site in the* $i$*th column. The resulting matrix looks as following*

$$C^{(1)} = \begin{pmatrix} c_{11...1} & \cdots & c_{1d...d} \\ c_{21...1} & \cdots & c_{2d...d} \\ \vdots & \ddots & \vdots \\ c_{d1...1} & \cdots & c_{nd...d} \end{pmatrix}.$$

  *This reshaping is sometimes referred to as a* ***tensor unfolding*** *or* ***matricization****.*

- *We introduce the* ***reshaping*** *operation* $(.)' : \mathbb{C}^{d^l \times d^{L-l}} \to \mathbb{C}^{d^{l+1} \times d^{L-l-1}}$*. It maps a coefficent matrix which orders its first* $l$ *coefficients in rows and the remaining* $L-l$ *in columns into a coefficent matrix which orders its first* $l+1$ *coefficients in rows and the remaining* $L-l-1$ *in columns.*

We show that a general pure state $\phi = \sum_{s_1,..,s_L}^{d} C_{s_1,...,s_L} |s_1...s_L\rangle$ can be described by a matrix product state.

It is necessary to arrange the coefficients in matrix form, because the singular value decomposition is only valid for matrices. Generalizations for higher order tensors exist, but they are not unique and do not possess all properties of the standard singular value decomposition. An overview over these generalizations can be found in the work by *De Lathauwer* et al [10].

We aim to express a single coefficient as a matrix product. We use the singular value decomposition and the tensor unfolding to transform a general state into a matrix product state. We perform a matricization of the coefficient tensor by moving the coefficients of the first site into the first row. We call the resulting matrix $C^{(1)}$

$$C^{(1)} = U^{(1)} \Sigma^{(1)} V^{(1)}, \quad U^{(1)} \in \mathbb{C}^{d \times d}, \Sigma^{(1)} \in \mathbb{C}^{d \times d^{L-1}}, V^{(1)} \in \mathbb{C}^{d^{L-1} \times d^{L-1}}.$$

Since we are interested in expressing the single coefficients we consider the matrix elements in more detail

$$C_{ij_2...j_L} = C^{(1)}_{ij} = \sum_{k=1}^{d} U^{(1)}_{ik} \Sigma^{(1)}_{kk} (V^{\dagger})^{(1)}_{kj}.$$

Right now $U^{(1)}$ is a $d \times d$ matrix. We want to express coefficients, that are of course scalars, as some form of matrix product. Regardless how the product continues we should start with a row vector to obtain again a scalar in the end. That is why we decompose $U^{(1)}$ into its row vectors $U^{(1)}_{:,k}$ for $1 \leq k \leq d$. To describe the particular coefficient $C^{(1)}_{ij}$ we keep only the for that coefficient relevant $i$th row of $U^{(1)}$. We define

$$C^{(n+1)} := \Sigma^{(n)}(V^{(n)})^\dagger.$$

We apply the reshaping operator and another singular value decomposition yielding

$$(C^{(2)})' = U^{(2)}\Sigma^{(2)}(V^\dagger)^{(2)}(U^2)' \in \mathbb{C}^{d^2 \times d^{L-2}}.$$

Now again $U^{(2)}$ has to be reshaped because it is a $d^2 \times d^2$ matrix, which can not be multiplied with $U^{(1)}_{:,k} \in \mathbb{C}^d$. We split it into $d$ matrices $U^{(2)}_{j_r}$ $1 \leq j_r \leq d$ of dimension $d \times d^2$ each. We keep the $j_2$th of these matrices $U^{(2)}_{j_2}$ for expressing $C_{ij_2\ldots j_L}$. Note that $U^{(2)}_{j_r}$ and $U^{(1)}_{:,k}$ can be multiplied! We repeat this procedure iteratively generating in this way matrices $U^{(l)}_{j_r}$ $1 \leq l \leq L$ with $U^{(l)}_{j_r} \in \mathbb{C}^{d^{l-1} \times d^l}$. Finally we obtain

$$C_{ij_2\ldots j_L} = U^{(1)}_i U^{(2)}_{j_2} \ldots U^{(L)}_{j_L},$$

where the last $C^{(L)}$ is already a column vector and can be simply renamed into $U^{(L)}$. All of this matrix products can be formed by construction. The product starts with a row vector and ends with a column vector so it is simply a number again! This is consistent with the left side and what we wanted from beginning. We have shown

$$|\phi\rangle = \sum_{s_1,\ldots,s_L=1}^{d} C_{s_1\ldots s_L}|s_1\ldots s_L\rangle = \sum_{s_1\ldots s_L=1}^{d} U^{(1)}_{s_1}U^{(2)}_{s_2}\ldots U^{(L)}_{s_L}|s_1,..,s_L\rangle = |M\rangle.$$

We call this procedure **left canonisation**.

However there is a second way to choose $C^{n+1}$. Let us assume we have done $l$ left canonisation iterations, we obtain after another SVD with a different labelling

$$\sum_{s_1,\ldots,s_l=1}^{d} U^{(1)}_{s_1}\ldots U^{(l)}_{s_l}C^{(l+1)} = \sum_{s_1,\ldots,s_l=1}^{d} U^{(1)}_{s_1}\ldots U^{(l)}_{s_l}\underbrace{U^{(l+1)}\Sigma^{(l+1)}}_{:=C^{(l+2)}}(V^{(l+1)}_{s_L})^\dagger.$$

This is a **right canonisation**. We call the site $l$, the **site of canonisation**. It is the site where we switch from left to right canonisation, typically this will be the border between the system and environment block. Special cases are pure right or left canonisation. This is iterated $L - l$ times leaving us with

$$c_{s_1,\ldots,s_L} = \sum_{s_1,\ldots,s_L} U^{(1)}_{s_1}\ldots U^{(l)}_{s_l}U^{(L)}_{s_{l+1}}\Sigma^{(L)}(V^{(L)}_{s_{l+2}})^\dagger\ldots(V^{(l+2)}_{s_L})^\dagger,$$

in case of a general site of canonisation at $s_{l+1}$. For the special case of left canonisation we obtain

$$c_{s_1,\ldots,s_L} = U^{(1)}_{s_1}...U^{(L)}_{s_L}. \tag{2.1}$$

### 2.3.4 Properties of Matrix Product States

We constructed the matrix product state entirely by using the singular value decomposition. The constructed state inherits useful properties of the singular value decomposition.

First the singular value decomposition gives us a natural way to find lower rank approximations $\tilde{\phi}$ of any state $\phi$. If $\mathrm{rk}(\Sigma) = r$ and we want a $t < r$ rank state

$$\tilde{\rho} = U_{1:t,:}\Sigma_{1:t,1:t}(V_{:,1:t})^\dagger,$$

is the approximation of $\rho$ with minimal Frobenius norm. This is an easy way to project $\rho$ onto a basis of it's $t$ largest eigenvalues. A projection onto this basis is a key ingredient of DMRG.

Furthermore certain normality properties are guaranteed by the singular value decomposition. The singular value decomposition guarantees

$$\sum_i U^{(j)}_{:,i}(U^{(j)}_{:,i})^\dagger = \mathbb{1}\forall j,$$

$$\sum_i V^{(j)}_{i,:}(V^{(j)}_{i,:})^\dagger = \mathbb{1}\forall j.$$

This simple consequence justifies choosing the border between system and environment block as the site of canonisation. We can ensure by the above the orthonormality within the different blocks.

### 2.3.5 DMRG produces Matrix Product States

We will now show that any DMRG calculation produces in fact always matrix product states. The result has been proven by *Rommer* & *Östlund* [19]. To see that let us regard DMRG as a process in essential determining a certain projector $P$. The exact way how DMRG chooses this projector is unimportant, it suffices that it does.

DMRG is in this view simply a way of projecting the tensor product of a reduced $D$-dimensional block $S^{(L)}$ and a site $s$ back into a one site longer block $S^{(L+1)}$, that should again be described by a basis of only $D$ dimensions.

We assume for simplicity that all sites are describable by a $d$-dimensional Hilbert space $\mathcal{H}$. Let $|k\rangle_{S^L} \in \tilde{\mathcal{H}}_L \subset \mathcal{H}^{\otimes L}$ and $\dim(\tilde{\mathcal{H}})=D$ with $D \ll d^L$ be the $k$th basis element of a reduced $D$-dimensional block of length $L$.

Let $|l\rangle_s$ be the the $l$th basis element of a $d$ dimensional site $s$.

We define $P_{L+1} : \tilde{\mathcal{H}}_L \otimes \mathcal{H} \to \tilde{\mathcal{H}}_{L+1}$ as exactly the projector describing the truncation process in DMRG. A projector is a linear mapping and linear

mapping can be written as a matrix. We express the basis of the composite system as

$$|j\rangle_{S^{(L+1)}} = \sum_k \sum_l P_{L+1} |k\rangle_{S^{(L)}} |l\rangle_{s_{L+1}}.$$

This is an inductive relation which we can exploit by replacing $|k\rangle_{S^{(L)}}$, obtaining

$$|j\rangle_{S^{(L+1)}} = \sum_{l_{L+1},\ldots,l_1}^{d} P_{L+1}...P_1 |l_1...l_{L+1}\rangle_{s_1...s_{L+1}}.$$

Any state $|\psi\rangle_{DMRG}$ produced by the DMRG algorithm can always be written as a tensor product of the system block basis and its environment block basis. We utilise the relation above for the two different block bases utilising $A$ for the projectors of the system block and $B$ for the projectors of the environment block

$$|\psi_{DMRG}\rangle = \sum_{s,e} C_{i,j} |s\rangle_{S^{L/2}} |e\rangle_{L/2} = \\ = \sum A_{L+1}....A_{L/2+1} B_{L/2}...B_1 |l_1...l_{L+1}\rangle_{s_1...s_{L+1}}. \quad (2.2)$$

This is a matrix product state. Therefore DMRG naturally creates matrix product state.

The argument above hardly uses any properties of DMRG, therefore it holds more generally. All procedures which control growing dimensions by projecting back onto some fixed dimensions can be expressed as matrix product states.

### 2.3.6 Variational Matrix Product States

Traditionally DMRG has been considered as a renormalization group method. Surprisingly this viewpoint is not the only one anymore. The discovery that DMRG produces matrix product states has led to some authors considering DMRG as a variational method minimizing the energy over all matrix product states. This has led to a reformulation of the DMRG algorithm sometimes denoted shortly as (VMPS) or **variational matrix product states** [31]. This approach allows to approach DMRG without utilising the classic algorithms given in Section 2.2.1 and 2.2.2! Instead we can address several problems by means of quadratic optimization techniques.

For example we reformulate the calculation of a ground state algorithm into a multiquadratic problem. We find the ground state of a Hamiltonian $H$ by solving the following problem

$$\begin{aligned} \min_{\psi} \quad & \langle\psi|H|\psi\rangle \\ \text{subject to} \quad & \langle\psi|\psi\rangle = 1 \\ & \psi = \sum_{s_1=1\ldots s_L=1}^{d} M^{s_1} M^{s_2} ... M^{s_{L-1}} M^{s_L} |s_1...s_L\rangle. \end{aligned}$$

It has been demonstrated by *Eisert* [3] that solving this problem is in general NP-hard! This problem is solved by employing the **alternating least squares method** (ALS). ALS means starting with some guess for $\psi$ and then iteratively fixing all but one component $j$ of $\psi$. This simplifies our multiquadratic problem into a quadratic problem. This problem is now solvable. We will get some solution of the quadratic problem $\hat{\psi}^0$. Choose another component to vary over and repeat. This yields a decreasing sequence of approximate solutions $\hat{\psi}^l \leq \hat{\psi}^{l+1} \leq ... \leq \hat{\psi}^{l+n} \forall n$. There is no guarantee of convergence, if we use this simplification procedure! Still it is claimed to converge in practice.

### 2.3.7 Performance of DMRG in 1D and 2D+

DMRG divides a system into two blocks $S, E$. In each block we only allow a certain maximal dimension $D$. Higher dimensional bases of a block's state $\rho_S$ or $\rho_E$ are projected onto the $D$-dimensional truncated eigenbasis of the $D$-largest eigenvalues $\lambda_1 \leq ... \leq \lambda_D$. When will the obtained $D$-dimensional approximation $\tilde{\rho}$ be able to represent $\rho$ well? DMRG can inherently only represent systems with decaying eigenvalues well! The more significant the contributions of the smaller eigenvalues $\lambda_{D+1} \leq ... \leq \lambda_n$ are the less reliable the method gets. This relates DMRG performance to the spectrum via

$$\|\rho - \tilde{\rho}\| \to 0 \Rightarrow \left\| \sum_{i=D+1}^{\dim(\rho)} \lambda_i \right\| \to 0.$$

Let us recall the definition of the entanglement entropy of a bipartition of the composite system $\rho_{AB}$ into $\rho_A$ and $\rho_B$

$$\mathcal{E}(\rho_{AB}) = S(\rho_A) = -\sum_{i=1}^{n} \lambda_i log_2(\lambda_i).$$

Here $\lambda_i$ denotes the eigenvalues of $\rho_A$. This allows us to connect the notion of entanglement entropy to the spectrum. A spectrum with only one large eigenvalue will have minimal entropy. This extreme case corresponds in fact to a pure state. A spectrum of many equally sized eigenvalues favours large entropy. The other extreme case will be the maximally mixed state. Since DMRG performs better if only few eigenvalues are large, the equation tells us that DMRG performs well if we have small entanglement entropy.

This may be a nice connection to physics, but we have not yet tackled our initial question why DMRG performance is related to its dimension. Surprisingly this is answered in the literature by the equation above. It has been found that certain states, including ground states and especially matrix product states[21], which are exactly the states created by DMRG exhibit the so-called entropic area law

$$S(\rho_A) \sim |\delta A|,$$

where $\delta A$ denotes the border of $A$. Basically the law states that a subsystems entanglement entropy scales proportional to the size of the border of the subsystem and the remainder. Intuitively this behaviour should be expected of systems with short range or next neighbour interactions, since there interior sites wouldnot interact outside of theis block. On the other hand systems with long range interaction should scale proportional to the volume.

The entropic area law is a subject currently undergoing research. It is not entirely clear which states satisfy it. However there are many single results, where the area laws have been proven for particular states or some classes of states. A clear classification is not yet published. Maybe the most important result has been proven by Hastings [9] showing that ground states of one dimensional, non-degenerate, one dimensional systems with local interactions satisfy the entropic area law. Furthermore Brandao & Horodecki [2] have shown that one dimensional systems with exponentially decaying spectrum satisfy the are law. Similar results are expected for higher dimensional systems, but to the best of our knowledge not yet proven. Even though many states do not satisfy the area law, it seems that interesting class of groundstates tend to satisfy it.

It is seen as the relation between the entropy and the dimension of the spin model, because borders are related to dimension. Eisert et al. [4] provide a review article discussing the entropic area law for many physical systems and its connection to DMRG.

A border is dependent on the dimension of the system. Especially in the case of a one dimensional system the border of a block consists always of two sites. This is totally independent of the block length $l$ leading to

$$|\delta A| = O(1),$$

for one dimensional systems! This means that the entropy is bounded regardless of system size and with that we get somewhat heuristically a bound on the equal distribution of the spectrum. In contrast in two or more dimensional spin models, if we have large systems and large blocks we also will have large borders. In this case it can and in practice does happen that DMRG is not suited to describe these systems anymore. Higher dimensional systems borders scale as

$$|\delta A| = O(l^{d-1}),$$

making entropy between a block and the remainder grow with block size. This is considered the reason why DMRG performs different in one and higher dimensional systems.

We can take the physical interpretation a little bit further. Since an increase in entropy is bad for DMRG performance, one idea to improve upon DMRG was to try to avoid entanglement in the systems we study. This is the key ingredient in Vidal's multi scale entanglement renormalization ansatz we describe in Section 4.1.

# Chapter 3

# Tree Tensor Network and Contractibility

In this chapter we will introduce tensor networks. We employ them to describe the geometry underlying the interactions of a composite system. Traditionally DMRG assumes the geometry of a line. A tree can be regarded as a generalization of a line. The idea is to utilise the method for composite systems describable by tree geometries. This allows us to treat systems that are somewhat in between the one dimensional and two dimensional case.

In order to define tensor networks we need the notion of a contraction. Consider the following simple case of a product of several matrices $A, B, C, D \in \mathbb{C}^{n\times n}$ and a vector $v \in \mathbb{C}^n$

$$x = ABCDv.$$

We get of course by the standard definition of matrix multiplication, the component wise formula

$$x_i = \sum_j \sum_k \sum_l \sum_m a_{ij} b_{jk} c_{kl} d_{lm} v_m. \tag{3.1}$$

Note that only the terms $j, k, l, m$ which appear in two different coefficents are summed over. In this case only $i$ appears once and it is not summed over. Vectors and matrices could also be seen as tensors of respectively order 2 and 1. We write the above as tensors , pay attention that $x^i$ is not a component but a (1,0)-tensor i.e. a vector.

$$x^i = A^i_j B^j_k C^k_l D^l_m v^m \tag{3.2}$$

We employ Einstein's summation convention, where we sum over an index if it appears once as superscript and once as subscript. We also see that in the resulting $x^i$ all indices we summed up vanished and only $i$ remains.

If we are interested in studying tensors of order 3 or higher, we might want to generalize this approach. The resulting operation, in fact simply summation

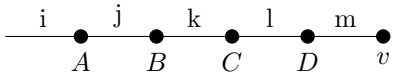


Figure 3.1: A simple tensor network representing the equations 3.1 & 3.2 from above

over matching superscript and subscript indices, is called contraction. The argument above should illustrate that the contraction is simply a generalization of the matrix multiplication to higher order tensors.

In order to avoid keeping track of lots of indices, which inevitable result from employing higher order tensors and their contractions, we introduce the so-called tensor networks in this chapter. In addition tensor networks allow us to connect multilinear algebra with graph theory.

## 3.1 Tensor Networks

Tensor networks are a way of visualizing complicated tensors by a graphical representation. We start by visualizing the above tensor network.

**Definition. 3.1.1.** *A **tensor network** is a pair (G,$\mathcal{T}$). $G(V,E)$ is a graph with an edge labelling $\mathcal{J} = j_1, .., j_n n = |E|$ and a vertex labelling $\mathcal{I} = i_1, ..., i_m m = |V|$, where edges may connect with either one or two vertices! $\mathcal{T}$ is a family of tensors where the rank of $T_{i_k}$ equals the degree of $v_{i_k}$ and the indices of $T_{i_k}$ share the labelling of the edges connected to $v_{i_k}$.*

*The tensor associated to the tensor network graph is the tensor obtained by taking the product of the tensors $T_{i_1}, ..., T_{i_m}$ and summing over all edge indices $j_l$ shared by any $T_{i_a}, T_{i_b} \in \mathcal{T}$.*

Take care not to confuse the summation indices with the vertex labelling!

**Definition. 3.1.2.** *A edge connected to only one vertex is called an **open** index.*

If we want to describe covariant and contravariant tensors, we should require $G$ to be a directed graph. This allows us to define:

**Definition. 3.1.3.** *Let $v_{i_k} \in V$ with $fan-in(v_{i_k}) = r$ and $fan-out(v_{i_k}) = s$. The $r$ edges ending in $v_{i_k}$ are called **arms**. The $s$ edges starting in are called **legs**. The arms are usually depicted pointing upwards and the legs are pointing downwards from the vertex.*

**Definition. 3.1.4.** *A **directed tensor network** is a pair (G,$\mathcal{T}$). $G(V,E)$ is a digraph with an edge labelling $\mathcal{J} = j_1, .., j_n n = |E|$ and a vertex labelling $\mathcal{I} = i_1, ..., i_m m = |V|$, where edges may connect with either one or two vertices! $\mathcal{T}$ is a family of tensors where $T_{i_k} \in \mathcal{T}^r_s$ if $v_{i_k}$ has $r$ arms and $s$ legs. The $r$ covariant indices are labelled like the arms and the $s$ contra like the legs of $v_{i_k}$.*

*The tensor associated to the tensor network graph is the tensor obtained by taking the product of the tensors $T_{i_1}, ..., T_{i_m}$ and summing over all edge inidices $j_l$ shared by any $T_{i_a}, T_{i_b} \in \mathcal{T}$.*

Connecting two edges simply means summing over the summation index denoted by their shared label. This can be seen as a graphical representation of Einsteins summation convention.

**Definition. 3.1.5.** *A **tree tensor network** is an tensor network that has tree form.*

For a lenghty but mathematical concise definition of a tensor network and its associated graph refer to Espig et al. [5]. The graphical representation of tensors is also known as Penrose graphical notation. It might be worthwhile to read Penrose [16] work for an introduction to the notation.

**Definition. 3.1.6.** *We call a edge representing a site a **physical index**, it has to be designated whetever a edge is physical or not. Connecting two vertices by a physical index results in summation over the index and the respective basis of the site!*

*A edge not representing a physical index is a **virtual index**. Connecting two vertices by a virtual index result only in summation over the index!*

There is no mathematical way to see what a physical index is or not. This is modelling choice. Connecting a tensor to physical index means that it acts locally on that site. Connecting a tensor to virtual index means that the tensor acts directly on another tensor and indirectly on all the sites connected to it.

**Definition. 3.1.7.** *Another graph $G'(V, E)$ is **associated to a tensor network** $G$, if the graph can be gained by a sequence of allowed manipulations of the tensor network. Allowed manipulations are contracting edges of the graph $G$ or inserting two unitary tensors in an edge.*

Note that a graph associated to a tensor network is by no means unique! We will introduce in the following the allowed manipulations and argue why it is sensible to consider them as allowed. We will see that under the right manipulations the graphs associated to the network are invariant. Later on we will introduce a second class of manipulations, which change the network, but compute something interesting in the process.

## 3.2 Invariant Operations on Tensor Networks

We call contracting an edge, if we replace an edge and its adjacent vertices, by a vertex representing the two multiplied tensors of the removed vertices summed up over all values of the removed edges index.

**Definition. 3.2.1.** *If $A$, $B$ are tensors their respective **contraction** Con over the indices $j_1, ..j_n$ is again a tensor $C$*

$$\mathrm{Con}(A^{i_1,..i_m}_{j_1,...,jn} B^{j_1,..j_n}_{k_1,...,kl}) = \sum_{j_1,...,jn} A^{i_1,..i_m}_{j_1,...,jn} B^{j_1,..j_n}_{k_1,...,kl} = C^{i_1,..i_m}_{k_1,...,kl}.$$

*We call the* ***value*** *of a tensor network the number which results after contracting all tensors. If we do not want to contract all indices we write the indices to be contracted as subscript of* Con *like this*

$$\underset{j_r j_s}{\mathrm{Con}}(A^{i_1,..i_m}_{j_1,...,jn} B^{j_1,..j_n}_{k_1,...,kl}) := \sum_{j_r,js} A^{i_1,..i_m}_{j_1,...,jn} B^{j_1,..j_n}_{k_1,...,kl}.$$

The second important manipulation is **inserting suitable tensors** into the tensor network. Note that a single vector or a matrix are of course only rank-1 or rank-2 tensors and can be just treated as any other matrices. Therefore in tensor networks there is always a certain degree of freedom because between two neighbouring sites $s_i, s_{i+1}$ one can always insert matrices $BB^{-1} = \mathbb{1}$ and contract $s_i B, B^{-1} s_{i+1}$ leaving the tensor network invariant. This is referred to as the **canonical freedom** or inherent degeneracy of tensor networks.

If for example $B = U\Sigma V^*$ is a singular value decomposition we have

$$\sum_{i,j,k} A^i B^k_i C_i = \sum_{i,j,k} A^i U U^* B^k_i V V^* C_i = \sum_{i,j,k} (A^i)' \Sigma C'_i.$$

Therefore the canonical freedom opens up a number of interesting derived manipulations, such as the singular value decomposition or the QR decomposition. Especially the singular value decomposition is interesting since it splits a matrix into three matrices. Due to this the singular value decomposition can be considered as in some sense inverse to the contraction of a tensor, which merges several tensors into a single one.

We also note that the singular value decomposition is only defined for matrices or tensors of rank 2. There are generalizations of the singular value decomposition for higher order tensors. Unfortunately this requires us frequently to use the sometimes inconvenient matricization of higher order tensors, if we want to make them accessible to the singular value decomposition. The singular value decomposition has several desirable properties like the orthogonality of the rows or columns for $V$ and $U$ or uniqueness of the singular values.

There is no single generalization which fullfills all desired properties, so which one is to be utilised depends on the context. One rather successful approach to generalize the singular value decomposition as well as an overview and a discussion over alternative attempts is given by *De Lathauwer* et al. [10]. To the best of our knowledge there is no current literature employing higher order singular value decompositions in the context of tensor networks, this may be due to the difficulties of generalizing these.

We have introduced the contraction and the insertion of the unity as allowed manipulations. In other words the allowed manipulations are exactly those that leave the value of a tensor network invariant.

## 3.3 Noninvariant Operations on Tensor Networks

We will outline in the following manipulations of tensor networks, which calculate interesting properties of the tensor network. The possibility to represent

operations such as taking reduced density matrices and expectations by tensor networks have been pointed out by Shi et al. [22].

### 3.3.1 Reduced Density Matrices

Summing over a physical index $s_i$ is interpreted as summing over the respective basis of subsystem $s_i$. A contraction is nothing else than summation over the edge's index

$$\operatorname*{Con}_{s_i}(\mathcal{T}^{s_1\dots s_n}(\mathcal{T}^{s_1\dots s_n})^\dagger) = \sum_{i=1}^{d}\langle i_{s_i}|\psi\rangle\langle\psi|i_{s_i}\rangle =^! \operatorname{tr}_{s_i}(|\psi\rangle\langle\psi|).$$

This means that connecting the physical indices $s_i$ of a state's $|\psi\rangle$ tensor network representation $\mathcal{T}$ to the corresponding physical index $s_i'$ of it's adjoint is nothing else then the reduced density matrix traced over subsystem $s_i$. We can represent calculating the reduced density matrix of subsystems $s_{i_1}...s_{i_k}$ as a tensor network by connecting all physical indices $s_j \neq s_{i_1}...s_{i_k}$ to the same index of the adjoint tensor network.

Connecting all physical indices to their adjoint counter part results in simply taking the trace. By definition of the density matrix we have

$$\operatorname{Con}\mathcal{T}\mathcal{T}^\dagger = \operatorname{tr}(|\psi\rangle\langle\psi|) = \operatorname{tr}(\rho) = 1.$$

### 3.3.2 Expectation Values

Among the most common tasks is the calculation of an observable's expectation value. We have argued how to represent traces by tensor networks, so we can also represent expectation values, since they are defined via the trace. We now specialise to the case of tree tensor networks, because they allow a simple calculation of expectation values for pure states. Calculating a pure state's expectation value is equivalent to calculating the value of a certain tensor network.

Let $\mathcal{T}$ be the tensor network representing $|\psi\rangle$ with open indices $s_1, ..., s_n$ representing physical indices. If an observable $O$ acts on a site $s_i$ it is connected to the open physical index $s_i$ of $\mathcal{T}$ representing that site. An observable $O$ acting on n sites has to be of rank $2n$. $\operatorname{Con}(\mathcal{T}^{s_1\dots s_n}O^{s_i}_{s_i})$ has the same rank as $\mathcal{T}$ and can therefore be connected to $\mathcal{T}^\dagger$. $\operatorname{Con}(\mathcal{T}^{s_1\dots s_n}O^{s_i}_{s_i}(\mathcal{T}^{s_1\dots s_n})^\dagger)$ is a rank 0 tensor. This number is exactly the expectation value of $O$.

Tree tensor networks are especially suited for calculating observables which have support on very few leaves, then it is possible to make big effort reductions through the tree structure. Due to the orthogonality, that can always be enforced in tree tensor networks, the matrices connected directly to its conjugate transposed can be immediately contracted to the identity. This can make whole branches of the tree disappear.

## 3.4 Contractibility

Contracting a tensor network is a useful operation, because it enables us to manipulate the tensor network and easily calculate a number of physically interesting properties such as expectation values, reduced density matrices and correlation function. For example calculating expectation values is equivalent to contracting certain tensor networks. Furthermore should a tensor network be given in non tree form, it can be contracted into tree form.

The effort of calculating all the above properties is dominated by the cost of the singular value decomposition and the contraction of the tensor network, which in turn depends on the so called contraction complexity.

### 3.4.1 Effort of a single Contraction

We first ask how much effort contracting only two tensors takes. Let $u^{i_1 \dots i_l}_{j_1 \dots j_m}$ be a $(l,m)$-tensor and $w^{j_1 \dots j_m}_{k_1 \dots k_n}$ a $(m,n)$-tensor both defined on some vector space $\mathbb{V}$ of dimension $n$, then contracting $u$ and $w$ has effort of $O(n^{\mathrm{rank}(u)+\mathrm{rank}(w)-m})$.

We check that representing the matrix multiplications of two $n \times n$ matrices $u, v$ as a tensor network leads as expected to an effort of $O(n^3)$. One advantage of the tensor network representation is that one can easily read of the effort of contracting two tensors by counting the number of open indices of the two connected tensors, this is the power of the contraction effort.

It should be cautioned that when contracting more then two tensors, the complexity is determined by the maximal tensor rank appearing in the whole contraction process. This maximal rank depends on the contraction ordering!

### 3.4.2 Contraction Complexity of a Tensor Network

We define contraction complexity and note its relation to tree width as in Markov & Shi [11].

**Definition. 3.4.1.** *A **contraction ordering** of a graph $G$ $\pi$ is an ordering of all the edges of $G$ in which they are contracted.*

*The **complexity** of $\pi$ a contraction ordering, is the maximum degree of the merged vertices.*

*The **contraction complexity** $cc(G)$ of $G$ is the minimal complexity of all contraction orderings.*

*The **line graph** of $G$ is denoted as $G^*$.*

*A **tree decomposition** of a graph $G = (V, E)$ is a pair $(\mathcal{C}, \mathcal{T})$. $\mathcal{C}$ is a family of subsets of $C_\nu$ of $V$ and $\mathcal{T}$ is a tree. In addition we require the following properties:*

- $\bigcup_{\nu \in \mathcal{C}} C_\nu = V$
- *for all edges $(v, w) \in E$, there exists a $\nu \in \mathcal{C}$ with $v \in C_\nu$ an $w \in C_\nu$*
- *for all $i, j, k \in V$ with $j$ on a path from $i$ to $k$ in $\mathcal{T}$: $C_i \cap C_k \in C_j$*

*The **tree width** $tw(G)$ is the minimal value of the width of all tree decompositions of the graph $G$.*

It is not easy to see which contraction order is optimal but it is shown by Markov & Shi [11] that $cc(G) = tw(G^*)$.

Furthermore it is noted that though finding the tree width is NP-hard, Robertson & Seymour give a way of finding a nonoptimal tree decomposition of width $O(tw(G))$ [18]. Readers further interested in finding a tree decomposition should refer to Neumaier, who gives an review on how to efficiently find tree decompositions. [15].

If a problem has bounded tree width, it is said to lie in a class of so-called parametrized complexity. This means basically that an initially NP-hard problem, becomes tractable if we are able to bound a certain parameter, where in our case the tree width is the parameter. If the tree width is reasonably bounded we can efficiently contract a tensor network containing circles to a tree tensor network! The following work might be interesting, but technical. It concerns finding a best approximation, with given rank, to a tensor network by means of optimization see Espig et al. [5], which should allow to find easier contractible approximations of given tensor networks.

Furthermore it is notable that while the contraction complexity is not independent of the contraction order, the value resulting from contracting a tensor network is!

We close this section by summing up, that it is in general hard to find the optimal contraction order for a tensor network. It is however easy to check the effort of a particular contraction ordering. This leads to most authors simply trying to find a clever contraction ordering and then computing the effort for that ordering and bound the effort by that. Explicit optimal effort estimates are usually not given. This also means that it is very much possible that later papers give lower effort estimates if a better contraction ordering is found.

## 3.5 Causal Cones and Orthogonality

The canonical freedom of tensor networks can be exploited to it's full extent only in the case of tree tensor networks. Choosing a canonical representation enforcing the orthogonality of the basis at any local site $s_i$ becomes possible there. The canonical freedom allows us to insert suitable unitaries, which in turn makes singular value decompositions possible. In quantum mechanical context often the reformulation of the singular value decomposition as the Schmidt decomposition is used. They are mathematically equivalent. It allows us decompose the state of a composite system $|\psi\rangle_{AB}$ into a linear combination of the tensor product of the two orthogonal bases of the composite systems bipartition $|k\rangle_A \otimes |s\rangle_B$. If we have for example several $d$ dimensional sites $s_i$ forming one system we can proceed with

$$|\psi\rangle_{s_1\dots s_n} = \sum_{k_1=1}^{d} \lambda_{k_1} |k_1\rangle_{s_1} \otimes |l_{s_2\dots s_n}\rangle = \dots = \sum_{k_1,\dots,k_n=1}^{d} \lambda_{k_1} |k_1\rangle_{s_1} \otimes \dots \otimes |k_n\rangle_{s_n},$$

where all $(|k_i\rangle)_{1\leq i\leq d}$ are orthogonal bases.

The difference between a general tensor network and a tree network is that the latter one has leaves and no circles. The Schmidt decomposition is used to obtain a basis, that is the tensor product of an orthogonal basis at a leave and another basis of the rest of the system. Iteratively applying the Schmidt decomposition to basis of the remainder, we can obtain leave per leave an orthogonal basis until all local bases are orthogonalized as shown in Shi et al. [22] or QR decomposition in Murg et al. [12].

In a general tensor network it can happen that we have a circle and we cannot not apply the above because removing one edge does not result in a bipartition any more. If we start with one site $s$ of the circle and orthogonalize, we will have to orthogonalize a neighbour $s'$ later on. It is possible that recontracting the unitaries needed to orthogonalize this neighbour $s'$, destroys the orthogonality of the site $s$ we started at!

The reason for requiring the orthogonality condition is that many applications are interested in expectation values of operators, which always leads to some kind of quadratic problem.

If an operator only acts on a few leaves of a tensor tree network, whole branches of the tensor tree network disappear due to the orthogonality. This simplifies the calculation of the operators expectation value tremendously. To formalize the idea we introduce the causal cone of a site $s_i$.

**Definition. 3.5.1.** *Let $\mathcal{T}$ be a tree tensor network. The leaves of $\mathcal{T}$ are labelled $s_1, \dots, s_n$. We consider $\mathcal{T}'$ a directed graph, with identical edges and vertices as $\mathcal{T}$ ,where the edges always lead from the site with smaller tree order to the site with larger tree order. We call the subtree $\mathcal{C}_{s_i}$ of $\mathcal{T}$ the* ***causal cone of*** *$s_i$*

$$\mathcal{C}_{s_i} := \{v | v \sim s_i v \in \mathcal{T}'\}.$$

*More generally we have*

$$\mathcal{C}_{s_i,s_j} = \mathcal{C}_{s_i} \cup \mathcal{C}_{s_j}.$$

The causal cone $\mathcal{C}_{s_i}$ is the set of all sites in the tree, which can be influenced by operators acting only locally at a leave $s_i$. We already argued above in 3.3 that calculating the expectation of an operator $o$ acting on i.e. $s_l, s_k$ is the same as calculating $\mathrm{Con}(\mathcal{T}^{i_1,\dots,i_n} o^{i'_k,i'_l}_{i_k,i_l} \mathcal{T}^{\dagger}_{i'_1,\dots,i'_n})$.

Let for example $o$ be acting on sites $s_{i_k}, s_{i_l}$ then all tensors $w \notin \mathcal{C}_{s_{i_l},s_{i_k}}$ cancel leading to

$$E(o^{i'_k,i'_l}_{i_k,i_l}) = \mathrm{Con}(\mathcal{T}^{i_1,\dots,i_n} o^{i'_k,i'_l}_{i_k,i_l} \mathcal{T}^{\dagger}_{i'_1,\dots,i'_n}) = \mathrm{Con}(\mathcal{C}_{s_{i_1},\dots,s_{i_k}} o^{i'_k,i'_l}_{i_k,i_l} (\mathcal{C}_{s_{i_1},\dots,s_{i_k}})^{\dagger}).$$

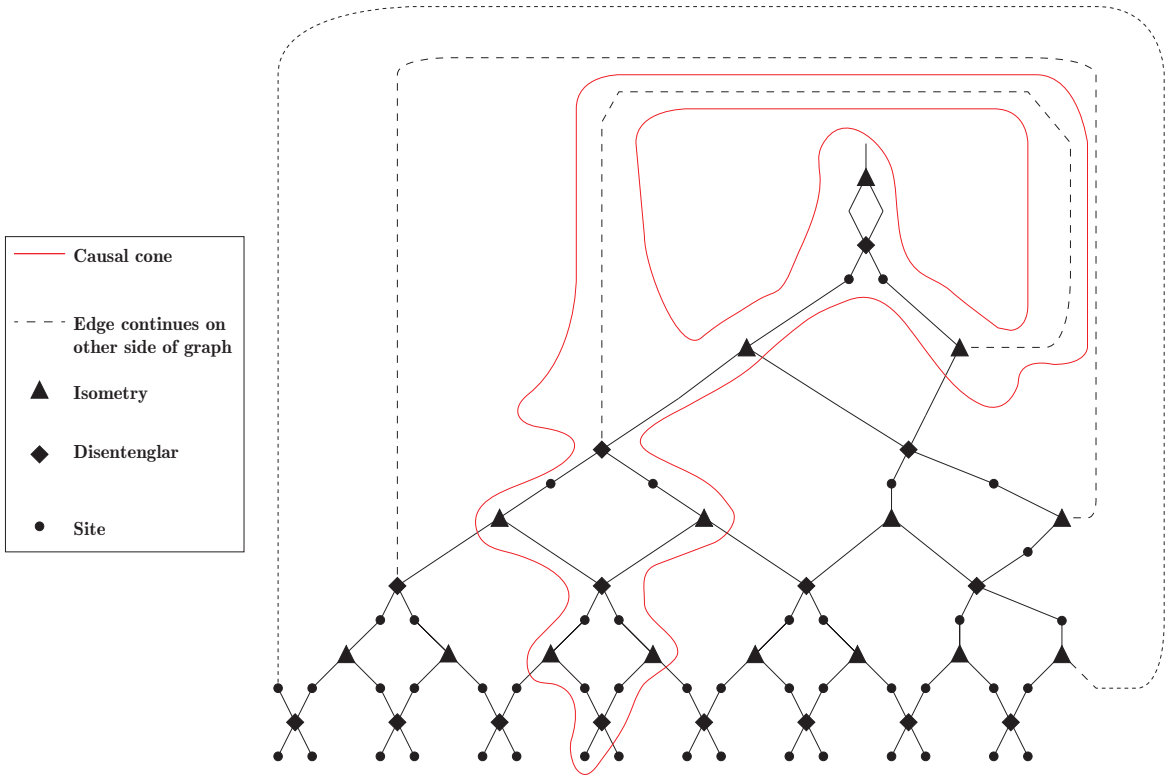


Figure 3.2: Graphical representation of the Causal Cone $\mathcal{C}_{s_7}$ on one dimensional 16 site lattice

For operators acting on few sites this results in big effort reductions. It can be argued that the effort reduction made by considering only the causal cone is the idea which motivated the research into tree tensor methods.

Traditionally DMRG has been used to find the ground states (i.e. Hamiltonians of minimal energy) of spin chain systems. Finding a ground state leads to a quadratic nonconvex minimization problem. Several authors propose schemes to compute the ground state of a tree tensor network by employing the causal cone.

## 3.6 Application: Ground State Calculation on a Tree Tensor Network I

In order to study the entropy of ground states of quantum mechanical systems representable by 2 dimensional lattices Tagliacozzo et al. [23] propose an iterative optimization procedure. To find the ground state of a tree tensor network. The lattice length is denoted $L$ and the number of sites is denoted

$N = L * L$. The tensors $s_1, ..., s_N$ at the leaves represent physical indices on the lattice, while the tensors $w$ connecting them are interpreted as several coarse graining steps applied to a lattice. A gauge condition is assumed requiring every tensor $w$ or $s_i$ to be an isometry in order to allow an easy computation of expectation values. This can always be achieved by exploiting the canonical freedom of tensor networks.

**Definition. 3.6.1.** *The **isometries** $w$, are rank $p+1$ tensors satisfying*

$$\sum_{\nu_1,...,\nu_p} (w^*)^{\alpha}_{\nu_1,...,\nu_p} (w)^{\alpha'}_{\nu_1,...,\nu_p} = \delta_{\alpha,\alpha'}.$$

The rank of $w$ depends on how many lattice sites are renormalized into a site. Renormalizing $p$ sites is described by a rank $p+1$ tensor.

To make use of the orthogonality properties of a tree tensor network we have to make the assumption that the Hamiltonian $H$ of our system can be decomposed into two body interactions $h_r$ yielding

$$H = \sum_r h_r.$$

It is a common assumption in the literature and therefore seems to be deemed justified in the literature. We will see later that it is a basic requirement for all tensor network based algorithms and is utilised in the multiscale entanglement renormalization ansatz and the projected entangled pair state algorithm as well. This assumption is essential because the $h_r$ only act on two sites $s_l, s_k$. Therefore they have small causal cones $\mathcal{C}_{s_l,s_k}$ and can be efficiently evaluated by a tree tensor network method.

To find the ground state $|\psi_{GS}\rangle$ of our lattice we would like to solve the quadratic problem

$$\min_{\psi} E(\psi) = \langle\psi|H|\psi\rangle.$$

In order to simplify this quadratic problem we regard a series of easier problems which depend only on a single tensor $w_i$

$$\min_{w_i} E(h_r) = \langle\psi(w_i)|h_r|\psi(w_i)\rangle.$$

We sweep through all the tensors $w_i$, optimising them site wise while fixing the rest. Let us call the two interacting sites described by $h_r$ $s_k, s_l$. We rewrite the general problem in terms of a tensor network by expanding $\psi$ into a tree tensor network $\mathcal{T}$

$$\langle\psi|h_r|\psi\rangle = \mathrm{Con}(\mathcal{T}^{s_1,...,s_n}(h_r)^{s_k,s_l}_{s_k,s_l}(\mathcal{T}^{\dagger})_{s_1,...,s_n}) = \mathrm{Con}(\mathcal{C}_{s_k,s_l}(h_r)^{s_k,s_l}_{s_k,s_l}(\mathcal{C}_{s_k,s_l})^{\dagger}).$$

This is already a much simpler tensor network then before, since $\mathcal{C}$ is a much smaller tree tensor network than $\mathcal{T}$. We split the tensor $w_i$ we want to vary in this step from the causal cone to be able to rewrite the simplified problems in tensor networks $\mathcal{C}_{s_k,s_l} = (\mathcal{C}_{s_k,s_l} \setminus w_i)^{\nu_{p+1}}_{\nu_1...\nu_p}(w_i)^{\nu_1...\nu_p}_{\nu_{p+1}}$.

This allows us to rewrite the above tensor network depending on $w_i$

$$\begin{aligned}&\langle\psi(w_i)|h_r|\psi(w_i)\rangle = \\ &= \mathrm{Con}[((\mathcal{C}_{s_k,s_l} \setminus w_i)^{\nu_{p+1}}_{\nu_1\dots\nu_p}(w_i)^{\nu_1\dots\nu_p}_{\nu_{p+1}})(h_r)^{s_k,s_l}_{s_k,s_l}((\mathcal{C}_{s_k,s_l} \setminus w_i)^{\nu_{p+1}}_{\nu_1\dots\nu_p}(w_i)^{\nu_1\dots\nu_p}_{\nu_{p+1}})^\dagger].\end{aligned} \tag{3.3}$$

Now remember that all the rewriting we have done has left the tensor network noninvariant. It is still the same problem we have started with and is still quadratic! To ease the quadratic optimization problem, it is linearized by considering $w^\dagger$ to be independent of $w$.

If we remove a single isometry $w$ and contract the remaining tensor network, including its now fixed conjugated counterpart $w^\dagger$, we gain the so called **environment** $\Upsilon$ of $w$. The effort is determined by the tree width of the tensor network

$$\underbrace{\mathrm{Con}[(\mathcal{C}_{s_k,s_l} \setminus w_i)^{\nu_{p+1}}_{\nu_1\dots\nu_p}(h_r)^{s_k,s_l}_{s_k,s_l}((\mathcal{C}_{s_k,s_l} \setminus w_i)^{\nu^{p+1}}_{\nu_1\dots\nu_p}(w_i)^{\nu_1\dots\nu_p}_{\nu_{p+1}})^\dagger]}_{:=\Upsilon^{p+1}_{\nu_1\dots\nu_p}}(w_i)^{\nu_1\dots\nu_p}_{\nu_{p+1}}).$$

It might be good to refer to figure 13 in Tagliacozzo et.al. [23] to visualize the environment. We have approximated our initial quadratic problem by a linear problem. This can now of course be solved. The problem

$$\min_w \mathrm{tr}(\Upsilon w)$$

is solvable by singular value decomposition of $\Upsilon = U\Sigma V^\dagger$ with solution $w' = -VU^\dagger$. $w'$ is of course by no means guarantueed to be a minimum of the quadratic problem but it yields a smaller or equal energy. We can iterate this processes with an updated value $w'$ by linearising again, recalculating the environment and solving a new linear problem. This could be done until convergence. This is a very popular optimization scheme for solving tensor network problems called **alternating least squares**, we will encounter again in all following applications!

The authors advise not to wait for convergence because it is unclear if quick convergence occurs. They rather accept a small decrease in energy and continue to the next tensor. Minimizing all tensors once defines one sweep. The sweeps are iterated until only small changes in energy occur delivering an tree tensor approximation to the ground state $|\psi_{GS}\rangle$. In principle every other way to minimize the energy could be chosen but this seems to be the most popular one in the literature.

## 3.7 Application: Ground State Calculation on a Tree Tensor Network II

Another way to calculated ground states by tree tensor networks is given Murg et al. [12]. We note that in February 2013 shortly before finishing this work a

new paper by NAKATANI & CHAN has been published continuing the work of MURG et al., that has not been included in our work, but should be noted since it directly continues this line of thought.

MURG et al. use a tree tensor network $\mathcal{T}$ with fixed coordination number z. All nodes represent a physical site. Comparably to DMRG a gauge condition is assumed, which enforces orthogonality of the basis on a local site. They propose an iterative two step optimization procedure to solve the problem

$$E = \min_{A_1,...,A_M,U} \langle \psi\left(A_1,...,A_M,U\right) |H| \psi\left(A_1,...,A_M,U\right)\rangle .$$

Here $A_i$ are the tensors of the tree tensor network $\mathcal{T}$, $U$ unitary rotations of the basis and $H$ the Hamiltonian and $|\psi\rangle = \sum_{k_1,..k_m} C_{k_1,..k_m}|k_1,..k_m\rangle$ resulting from contracting $A_i$. The algorithm divides itself into a network optimization step minimizing over the $A_i$ and an orbital optimization step minimizing over $U$. Both steps are made iteratively until convergence.

## Network Optimization

In the network optimization step we regard $U$ as fixed, resulting in a fixed basis. We want to minimize the energy subject to the constraint that $|\psi\rangle$ is normed over only the $A_i$ leading to

$$\min_{A_1,...,A_M} F = \langle\psi(A_1,...,A_M)|H|(A_1,...,A_M)\psi\rangle - E(\langle\psi(A_1,...,A_M)|(A_1,...,A_M)\psi\rangle - 1).$$

Similar to the alternating least squares method we have employed already, we fix all tensors but $A_m$ to simplify the problem. The resulting problem is a quadratic problem, solving quadratic problems is equivalent to solving generalized eigenvalue problems

$$H_m A_m = N_m A_m.$$

Here $A_m$ refers to the optimized tensor, $H_m$ is the effective Hamiltonian at site $m$. $N_m$ is introduced using $\vec{A}$ as the tensor written in vector form

$$\langle\psi|\psi\rangle = \vec{A}_m^\dagger N_m \vec{A}_m.$$

Since we are using a tree tensor network we can use the canonical freedom of tensor networks to get orthogonal local bases of the sites forcing $N_m = \mathbb{1}$. Now that only is an eigenvalue problem. What remains is calculating the effective Hamiltonian.

The Hamiltonian can be split in to sum of Hamiltonians $h_r$ yielding

$$H = \sum_r^R h_r,$$

where we will solve for each of the summands independently.

To find the effective Hamiltonian, we are forming the environment in this step. If we remove a tensor $A_m$ from $\mathcal{T}$, we are left with $z$ connected components

we label as $\mathcal{T}_s$ for $1 \leq s \leq z$, $z$ being the coordination number. We gain the effective Hamiltonian $h_{m,r}$ of the $r$th interaction at site $A_m$ if we contract along all edges except the $z$ edges of $A_m$

$$h_{m,r} = \mathrm{Con}(\mathcal{T}_1)^{i_1} \otimes ... \otimes \mathrm{Con}(\mathcal{T}_z)^{i_z} := h^1_{m,r} \otimes ... \otimes h^z_{m,r}.$$

Note that the contraction of the connected components $\mathrm{Con}(\mathcal{T}_s)^{i_s}$ is efficient due to the tree tensor network structure. This can be seen as a DMRG form with $z$ environment blocks instead of the usual one. Each $\mathrm{Con}(\mathcal{T}_s)^{i_s}$ describes the interaction of $A_m$, considered as the system, with the $s$th environment block.

$$\langle\psi|H|\psi\rangle = \sum_r \langle\psi|h_{m,r}|\psi\rangle$$

Each Hamiltonian splits into a sum of tensor product of $z$ matrices. Note that if $h_r$ describes only a few body action, many $h^i_{m,r}$ might be missing. In the case of a 2-body interaction we will have only 2 terms!

$$h_{m,r} = h^i_{m,r} \otimes h^j_{m,r}.$$

Calculating $\langle\psi|H|\psi\rangle$ all branches of the tree, where $h_r$ does not interact, disappear due to the orthogonality condition of the tree tensor network. Even vertices in a branch with non trivial interaction cancel, if they are not on the path between the two interacting sites. In the case of a 2-body interaction this simplifies the calculation to only the path connecting the interacting edges remaining to be calculated.

This can be regarded as a spin chain on which DMRG is possible, finding a state of minimal energy for that chain. Therefore one run of DMRG yields one 2-body interaction effective Hamiltonian. We repeat for all $h_r$ and the move on to the next site. Repeating this for all sites defines a sweep.

## Orbital Optimization

After several sweeps of network optimization, where the energy is minimized by varying the tensors in the tensor network, Murg et al. optimize the basis of the state $\psi$ in the so-called orbitial optimization step. They explain that the choice of basis can heavily influence how many basis elements $D$ are needed to describe the system. The authors use a basis in so-called second quantization, defined by

$$|k_1, ..., k_M\rangle = (a^\dagger_1)^{k_1} ... (a^\dagger_m)^{k_m}|0\rangle,$$

where $a_i$ denote so called creation operators, that arise in physics when descibing fock spaces. The explicit form of these operators depends on the particles that are studied, but they can be thought of as fixed matrices. We do not cover second quantization further, but simply remark that $|0\rangle$ is interpreted as the basis describing a state without any particle. Applying a creation $a_i$ operator changes this basis to another basis describing a system with on more particle.

The basis is optimized by unitary rotations $U$ in order to minimize the local virtual dimension $D$. Recall that the accuracy of a DMRG calculation depends on the truncated dimension $D_{DMRG}$. If we can make $D \leq D_{DMRG}$ all DMRG calculation in the orbital optimization step are exact, else we can at least improve DMRG's accuracy by minimizing $D$. To achieve this a gradient search is applied to

$$\min_U \langle \psi(U)|H|\psi(U)\rangle \,.$$

It is noted that the resulting problem is non convex, but finding a minimum is not necessary, simply minimizing the energy a bit suffices, to improve the algorithm.

## 3.8 Application: Circuits

We close the chapter on tree tensor networks with a small note on the topic of circuits, where tree tensor networks also have recently played a role. We are not going to treat them in depth, just note that these applications exist as well. In classical and quantum theoretical informatics central objects to be studied are circuits, who model calculation steps on a computer. One would assume some input string or state and a predetermined sequence of manipulations on that string or state. Each input bit or qbit corresponds to a leave and manipulations are represented vertices. The incoming edges represent input and the outgoing output.

Simulating these circuits under various inputs of qbits and calculating their respective expectation values on classical computer is often interesting in computer science. Circuits can be represented naturally by tree tensor networks. As long as we have circuits with reasonably bounded tree width computing an expectation value is possible efficiently see Markov & Shi. [11].

Apart from this practical question, there have been theoretical applications. Aharonov [1] et al. reduced the circuit of the quantum Fourier transform to tree tensor network with bounded tree width. This shows that it can be feasibly simulated on classical computers. This is a surprising fact since the quantum Fourier transform appeared to be a key ingredient in the prime number factorization algorithm on quantum computers, which can not be classical simulable.

# Chapter 4

# Higher Dimensional Algorithms

## 4.1 Multiscale Entanglement Renormalization Ansatz

Meanwhile several generalizations to DMRG have been proposed. One of the successful generalizations of the DMRG method is the multiscale entanglement renormalization ansatz. It has been proposed by VIDAL[29] in 2007 . In contrast to DMRG it is applicable for two dimensional or higher systems [7]. It also allows calculation of energy and expectation values of operators with support on few sites. It is a variation of the renormalization group idea.

Wilson's renormalization scheme only applied rescaling steps, approximating blocks of several sites in fine lattices by a single site in coarser lattices. The core idea of MERA is to add another renormalization step in between the rescaling steps. In this step the entanglement between a block and the adjacent blocks is minimized. This seems plausible since we have demonstrated in 2.3.7 that a large amount of entanglement entropy has limits the performance of DMRG in higher dimensions. The natural remedy in MERA is to eliminate that harmful entanglement.

MERA introduces two different kinds of tensors which are applied to a lattice in succession in every coarse graining step:

**Definition. 4.1.1.**

- *A **disentangler** $u$ is a rank 4 tensor satisfying*

$$\sum_{\nu,\mu}(u^*)^{\alpha\beta}_{\nu\mu}(u)^{\alpha'\beta'}_{\nu\mu} = \delta_{\alpha,\alpha'}\delta_{\beta\beta'}.$$

*We require furthermore that the disentangler decreases the entanglement of the sites it is applied to.*

- *An **isometry** $w$ is a rank 3 tensor satisfy*

$$\sum_{\nu,\mu}(w^*)^{\alpha}_{\nu\mu}(w)^{\alpha'}_{\nu\mu} = \delta_{\alpha,\alpha'}.$$

  *We require that the isometries map a block of neighbouring sites to a single site. We call applying an isometry **coarse graining** the lattice.*

- *We call a tensor in a tensor network, which is connected by $d$ disentanglers and $i$ isometries to the next leave, a tensor in **layer** $\tau = 1 + i + d$.*

MERA applies at first the disentantglers $u$ to all neighbouring sites, which are not in the same block. Second an isometry $w$ is applied to each block. It maps all sites of the block to a single site. This defines one iteration, each iteration yielding a renormalized lattice.

We have left the definition of decreasing entanglement and coarse graining intentionally inexplicit. This is because in the literature does not provide a constructive way of deriving the tensors. Instead an optimization algorithm is applied which determines the form of disentanglers and isometries best suited to the particular tensor network. We go on detailing the optimization procedure for an isometry $w$, in principle the same procedure is used for the disentanglers.

### 4.1.1 Determining Isometries and Disentanglers of the MERA

We point out that the optimization algorithm for single isometries $w$ or disentanglers is not given in the original paper[29] explaining MERA, but in a subsequent one [6].

In general every single isometry $w$ has to be optimized on its own, although symmetries (i.e. translation or scale invariance of a system) might reduce the effort considerably.

**Definition. 4.1.2.** *Let $(G, \mathcal{T})$ be a tensor network representing a MERA. We define the **environment** of $w \in \mathcal{T}$ by*

$$\Upsilon_w := \mathrm{Con}(\mathcal{T} \setminus w).$$

For each isometry $w$, we keep all other tensors (i.e. isometries and disentanglers) of $\psi$ fixed and only vary in $w$.

Let $\psi$ be a pure state, operating on a lattice of $L$ sites, we want to represent by means of a MERA. We assume $\psi$ can be represented by a tensor network $\mathcal{T}$ with $L$ open indices and index labelling $c$. Furthermore we denote by $\mathcal{T}^\dagger$ the family consisting of all conjugated transposed tensors from $\mathcal{T}$ with index labelling $i'_1, ..., i'_{|E|}$. All elements in $\mathcal{T}$ have either rank 4, rank 2 or rank 1.

We want to solve the following optimization problem

$$\begin{aligned}
\min_{w} \quad & E(\phi(w)) = \mathrm{Con}(\mathcal{T}^{i_1,\ldots,i_{|E|}} H^{i'_1,\ldots,i'_{|E|}}_{i_1,\ldots,i_{|E|}} (\mathcal{T}^\dagger)_{i'_1,\ldots,i'_{|E|}}) = \\
& = \mathrm{Con}((w\Upsilon_w)^{i_1,\ldots,i_{|E|}} H^{i'_1,\ldots,i'_{|E|}}_{i_1,\ldots,i_{|E|}} (w\Upsilon_w)^\dagger_{i'_1,\ldots,i'_{|E|}}) \\
\text{subject to} \quad & \sum_{\nu,\mu} (w^*)^\alpha_{\nu\mu} (w)^{\alpha'}_{\nu\mu} = \delta_{\alpha,\alpha'} \\
& \sum_{\nu,\mu} (w^*)^\alpha_{\nu\mu} (w)^{\alpha'}_{\nu\mu} = \delta_{\alpha,\alpha'}.
\end{aligned}$$

This is a quadratic optimization problem in $w$.

The following algorithm is used iteratively to calculate a solution for the problem:

- While optimizing over $w$, we keep the whole remaining tensor network constant including $w^\dagger$ pretending it to be independent of $w$! Thereby our quadratic problem is linearized. We then contract the whole part we are not optimizing into the environment $\Upsilon'_w$.

  $$\begin{aligned}
  \min_{w} \quad & E(w) = \mathrm{tr}(w\Upsilon'_w) \\
  \text{subject to} \quad & w^\dagger w = 1 \\
  & \sum_{\nu,\mu} (w^*)^\alpha_{\nu\mu} (w)^{\alpha'}_{\nu\mu} = \delta_{\alpha,\alpha'}
  \end{aligned}$$

  $\Upsilon'_w$ now contains the whole conjugated transposed part $\mathcal{T}^\dagger$!

- The linearized minimization problem is now easily solvable by means of a singular value decomposition of $\Upsilon_w = U\Sigma V$ yielding a solution of the simplified problem $w^0 = VU^\dagger$.

- We now return to the quadratic problem inserting $w^0$ and linearize again resulting in the new problem:

  $$\begin{aligned}
  \min_{w} \quad & E(w) = \mathrm{tr}(w\Upsilon'_{w^0}) \\
  \text{subject to} \quad & (w^0)^\dagger w = 1 \\
  & \sum_{\nu,\mu} (w^*)^\alpha_{\nu\mu} (w)^{\alpha'}_{\nu\mu} = \delta_{\alpha,\alpha'}
  \end{aligned}$$

  This updates the environment and we need to calculate the new $\Upsilon'_{w^0}$.

- Finally we solve the new linearized problem yielding a new solution $w^1$ as above and repeat

The authors do not give an analysis of convergence but claim that in practice very fast convergence is achieved. It is suggested to only use very few iterations instead of searching for true convergence before moving on. Since this algorithm will be run over and over again in several sweeps the authors seem to be confident that it is only required to improve the energy a bit in every step.

### 4.1.2 Optimization Sweeps over the Entire MERA

We now have a way to determine each tensor of the MERA. Running a small fixed number of algorithm steps for each of the involved tensors defines one sweep of the MERA. Usually several sweeps are applied. In practice effort can be saved by optimising the tensor in a sensible order. A sensible ordering is optimizing layer wise either from top-down or from-bottom up. The main effort of the MERA is determined by the contraction. If we start i.e. from bottom up we can reuse old contractions. The past causal cone of layer $\tau$ can be updated by contracting all tensors of the layer and then be reused as the past causal cone of the layer $\tau + 1$. Analogously the intersection of future causal cones needs to be only computed once.

To exactly quantify the effort of contracting the MERA is difficult since the optimal contraction order is related to the tree width see 3.4. Another problem is that there are many possible choices for the rank of the isometries and disentanglers and it is not clear what the optimal choice is. In fact finding the best combination of contraction order and tensor rank is still under research. This means that the effort anounced in papers can still go down later on.

In the case of the one dimensional MERA with type (2,2) disentanglers and type (1,3) isometries Evenbly & Vidal [6] initially assume an effort of $O(\chi^8)$, but in a subsequent paper a better contraction order resulting in $O(\chi^6)$[8] is found.

In the more interesting case of two dimensional MERA working with type (4,4) as well as type (4,2) disentanglers and type (5,1) isometries they derive an effort of rather big $O(\chi^{16})$ [29]. The authors are confident to improve the effort by finding a suitable scheme with a lower effort.

It is notable that MERA can be represented as a tensor network. The resulting networks are tree like structures. In fact a tree tensor networks could be seen as a special case of the MERA, where the disentanglers are set to the identity. A MERA network contains many circles but not very big ones. This makes them efficiently computable. Since the tree width is bounded, the mera can be contracted to a tree [11]. Therefore a MERA shares the benefits of a tree tensor network namely the easy calculation of 2 site operators.

## 4.2 Other Algorithms of Interest

### 4.2.1 Projected Entangled Pair States

We want to point out that there is at the moment another algorithm for simulating 2 dimensional systems besides the MERA. The so-called projected entangled pair states algorithm or shorter PEPS. It has been proposed in a paper by *Verstrate and Cirac* in 2004 [25]. PEPS and MERA are the two most discussed methods for higher dimensional systems in the literature at the moment. *Verstrate* et al. [26] provide a review article revising DMRG, matrix product states and projected pair states . This work has focused on the MERA but we want to emphasize that the PEPS approach is also completely viable. At the moment

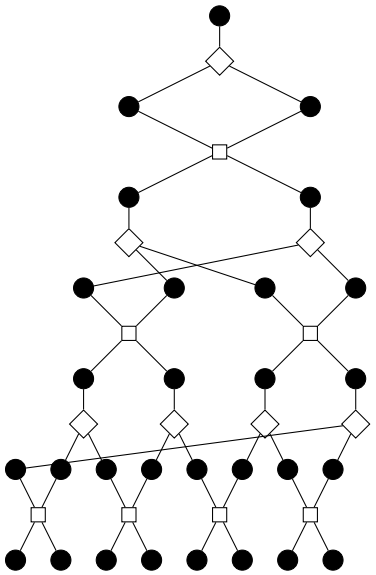

Figure 4.1: graphical representation of the multiscale entanglement renormalization ansatz

there seems to be no clear tendency in the literature to favour one approach over the other!

### 4.2.2 Time Evolution

Another topic left out up until now is how to calculate the time evolution of a system. It should be said that efficient time evolution algorithms exist. The most well-known may be Vidal's time evolving block decimation [28], it is a truncation procedure related to DMRG and we have added a small section of recommended reading introducing the various formulations of it in 5.5. Both two dimensional generalizations of DMRG PEPS [26] as well as MERA[17] allow for time evolution.

# Chapter 5

# Literature Overview

In the following we give a short summary of the utilised literature:

## 5.1 Density Matrix Renormalization Group

**The density-matrix renormalization group** by Schollwoeck [20] This is an extensive review article, which serves as the standard introduction into DMRG. It covers the basics of DMRG how it works and how it relates to matrix product states. Moreover it provides an overview over various fields and applications which have adapted the method with success such as quantum information theory or quantum chemistry. The only drawback is that it is not entirely new anymore, being written in 2004. Generalizations of DMRG on higher dimensions are only covered for the 2D case, applications with tree structure are not covered.

**The density-matrix renormalization group in the age of matrix product states** by Schollwoeck [21] Another long review article by Schollwoeck, which is also widely cited. It takes a different focus than the previous article. Published in 2011 it is quite recent it uses less time to talk about the basics of DMRG, provide citations to applications or give examples of dmrg variations such as momentum space DMRG or non equilibrium systems. On the other hand it focuses on the meanwhile discovered connection of matrix product states and the density matrix renormalization group. Important applications such as ground state calculations and time evolution are still covered. The two articles are somewhat complementary and can both be considered as the standard introduction into DMRG at the moment.

## 5.2 Tree Tensor Network Approaches

**Simulating Strongly Correlated Quantum Systems with Tree Tensor Networks** by Murg et al. [12].

This work is a good example for a generalization of DMRG on a tree tensor network. The theoretical part treats calculation of expectation values and ground state energy quite detailed. It might be notable that the author give some references to the use of DMRG in quantum chemistry. It is pointed out that fermionic systems are treatable by the tree tensor network methods in contrast to quantum Monte Carlo methods. The second part gives the results the authors obtained in a series of numerical simulations of models from statistical physics and quantum chemistry.

**Simulation of two-dimensional quantum systems using a tree tensor network that exploits the entropic area law** by Tagliacozzo et al. [23].

Together with the above work by Murg another of the more cited approaches to calculating ground states. The work is similar and also explains calculation of expectations or correlations. Vidal the inventor of MERA regards the tree tensor network as a special case of his MERA network and applies similar optimization techniques as in his work covering optimization of MERA. There is a short introduction to the entanglement are law, which is deemed to be the reason for the different achievement between 1d and 2d+ DMRG.

## 5.3 Multiscale Entanglement Renormalization Ansatz

**Class of quantum many-body states that can be efficiently simulated** by G. Vidal [30]

Here the multiscale entanglement renormalization ansatz is introduced and explained. It is interesting historically because it is the first paper on MERA. The article gives an introduction but is quite short and does not provide everything which is needed to implement a MERA, a review article like the following, which examines technical details should be read as follow up or even be the one to start with. It is along with projected entangled pair states one of the two successful generalizations of DMRG to higher dimensional systems. It is similar to tree tensor networks in some ways. It is describable also as a tree like tensor network, which simplifies quite similar to tensor tree networks when calculating expectation values of observable between few sites. The article should be read together with the earlier Entanglement Renormalization [29] by Vidal where he introduces the tensors which are needed to renormalize the entanglement of blocks in the MERA.

**Algorithms for entanglement renormalization** by G. Evenbly and G. Vidal [6]

This review article covers calculations with MERA. It gives an intro on the definition and motivation, evaluating two site operators and correlations, incorporation of translation and scale invariance and most importantly delivers optimization procedures to find the exact form of the tensors involved in MERA. It is recommended as an introduction to MERA. We also warn that the article exists in 4 versions, where some of the interesting content of earlier versions

have been dropped. Version two for example contains remarks on the explicit optimization which are lost in the newest version, it might be worthwhile to compare different versions.

Vidal has published several follow up articles on the topic of the MERA. In Entanglement Renormalization in Two Spatial Dimensions [7] might be of interest because it introduces for the first time a generalization of MERA onto 2 dimensional lattices.

## 5.4 Projected Entangled Pair States

**Matrix Product States, Projected Entangled Pair States, and variational renormalization group methods for quantum spin systems** by F. Verstrate et al. [6]

We recommend this review article as an introduction into the projected entangled pair states ansatz. The authors regard their work as another a two dimensional generalization of the variational matrix product states ansatz utilising another class of states to vary over. The projected entangled pair sates are better suited to approximating two dimensional lattices then matrix product states. They start with recalling the matrix product state formalism and then introducing their approach. Apart from the calculation of ground states, the calculation of time evolution is also covered.

## 5.5 Time Evolving Block Decimation

**Efficient simulation of one-dimensional quantum many-body systems** by G. Vidal [28]

In this paper Vidal introduces his time evolving block decimation method or (TEBD). It is closely related to DMRG. It shares the requirement on the simulation to be a one dimensional system and to restrict entanglement in the system. While DMRG simulates the ground state or other static properties of a system, TEBD allows to simulate the time development of a given ground state. Often one would first get the ground state via DMRG and then use TEBD on it, to simulate both at once.

The method has gained quite some attention leading to several generalizations especially suited to simulate infinite translational invariant one dimensional systems [27], tree structured systems [22] and finally infinite translational invariant tree structure systems [14].

**Local Hamiltonians in Quantum Computation** by D. Nagaj [13]

Unfortunately i do not know of a review article devoted specificly to TEBD, but for an overview i would recommend this thesis by Nagaj. The thesis covers a diverse set of topic so one should directly commence the chapter 3 "'Matrix Product States on Infinite Trees"', where he outlines the developments above.

## 5.6 Tensor Networks & Tensor Decompositions

**Applications of Negative Dimensional Tensors** by R. Penrose [16]

Introduces the concept of representing tensors by tensor networks for the first time. Gives very nice step by step introduction including pictures along with formulas.

**A Multilinear Singular Value Decomposition** by De Lathauwer et al. [10]

Proposes a multilinear generalization of the singular value decomposition. Among many other proposed generalizations this paper seem well received with over 1000 citations. Gives an overview over the several desirable properties of a generalized singular value decomposition and argues why some of them are attainable and others not with the proposed decomposition. De Lathauwer et al. are successful in generalizing concepts such as row and column orthogonality and singular value uniqueness. They state themselves that the generalization of rank is problematic in this approach. There is also a well written section with examples that illustrate i.e. the problem of generalized rank notions. Therefore there is no unified concept of a generalized SVD. De Lathauwers generalization can be seen as one example of the class of Tucker decomposition generalizations. Other attempts to generalize are also pointed out and compared, the most prominent other decompositions are the class of CP decompositions, which are better suited to generalizing the rank.

# Chapter 6

# Curiculum Vitae & Deutsche Zusammenfassung

## 6.1 Curiculum Vitae

**Personal**

| | |
|---|---|
| Name: | Claude Klöckl |
| Date of birth: | 11.06.1985 |

**Curiculum Vitae**

| | |
|---|---|
| 10.05 - 04.13 | Student of mathematics (University of Vienna)<br>Branch: "Applied Mathematics and Scientific Computing" |
| 10.04 - 10.05 | Civil service |
| 06.2004 | High school certificate |

**Further Skills**

| | |
|---|---|
| Languages: | German (fluent), English (fluent), French (moderate) |
| Experience with: | Matlab,Mathematica |

## 6.2 Deutsche Zusammenfassung

Das Ziel dieser Arbeit ist einen berblick ber die Entwicklung der Simulation von quanten Vielteilchenproblemen zu geben. Wir behandeln die Algorithmen die auf der Density Matrix Renormalization Methode basieren. Sie ist eine der schnellsten und przisesten Simulationsmethoden. Allerdings war ursprnglich die Methode ausschlielich zur Simulation eindimensionaler Probleme gedacht. Diese Arbeit behandelt den klassischen Ansatz sowie die Versuche diese eindimensional Methode auf hherdimensionale Probleme umzulegen. Im speziellen

wird auf die Mglichkeit der generalisierung auf Probleme mit baumartiger Geometrie eingegangen und auch auf andere Tensornetzwerk basierende Methoden eingegangen.

# Bibliography


[1] Aharonov, Landau, and Makowsky. The quantum FFT can be classically simulated. *arxiv preprint*, 2008.

[2] F. G. S. L. Brandao and M. Horodecki. Exponential decay of correlations implies area law. *arXiv:1206.2947*, 2012.

[3] J. Eisert. Computational difculty of global variations in the density matrix renormalization group. *Phys. Rev. Lett. 97, 260501-260505*, 2006.

[4] J. Eisert, M. Cramer, and M.B. Plenio. Area laws for the entanglement entropy - a review. *Rev. Mod. Phys. 82, 277-306*, 2008.

[5] M. Espig, W. Hackbusch, S. Handschuh, and R. Schneider. Optimization problems in contracted tensor networks, 2011. http://www.mis.mpg.de/publications/preprints/2011/2011-66.html.

[6] G. Evenbly and G. Vidal. Alghorithms for entanglement renormalization. *Phys. Rev. B 79-102, 144108*, 2009.

[7] G. Evenbly and G. Vidal. Entanglement renormalization in two spatial dimensions. *Phys. Rev. Lett. 102, 180406*, 2009.

[8] G. Evenbly and G. Vidal. Quantum criticality with the multi-scale entanglement renormalization ansatz. *arXiv:1109.5334 [quant-ph]*, 2011.

[9] M. B. Hastings. An area law for one dimensional quantum systems. *JSTAT, P08024*, 2007.

[10] L. De Lathauwer, B. De Moor, and J: Vandewalle. A multilinear singular value decomposition. *Rev. Mod. Phys. 77, 259315*, 2000.

[11] Markov and Shi. Simulating quantum computation by contracting tensor networks. *SIAM Journal on Computing*, 2008.

[12] V. Murg, F. Verstrate, . Legeza, and R.M. Noack. Simulating strongly correlated quantum systems with tree tensor networks. *Physical Review B*, 2010.

[13] D. Nagaj. Local hamiltonians in quantum computation. *Thesis published on Arxiv e-print*, 2008. http://arxiv.org/abs/0808.2117.

[14] D. Nagaj, E. Farhi, J. Goldstone, P. Shor, and I. Sylvester. Quantum transverse-field ising model on an infinite tree from matrix product states. *Phys. Rev. B 77, 214431-214446*, 2008.

[15] A. Neumaier. Tree decompositions and large-scale computation, 2012. http://www.mat.univie.ac.at/ neum/Tree/Tree.pdf.

[16] R. Penrose. Applications of negative dimensional tensors. *Society for Industrial and Applied Mathematics Journal Matrix Anal. Appl. Vol. 21, No. 4, pp. 12531278*, 1971.

[17] M. Rizzi, S. Montangero, and G. Vidall. Simulation of time evolution with multiscale entanglement renormalization ansatzn. *Phys. Rev. A 77, 052328-052332*, 2004.

[18] Robertson and Seymour. Graph minors x. obstructions to tree decompositions. *Journal of Combinatorial Theory, Series B*, 1991.

[19] S. Rommer and S. Östlund. A class of ansatz wave functions for 1d spin systems and their relation to dmrg. *Physical Review B 55, 21642181*, 1997.

[20] U. Schollwöck. The density-matrix renormalization group. *Rev. Mod. Phys. 77, 259315*, 2005.

[21] U. Schollwöck. The density-matrix renormalization group in the age of matrix product states. *Annals of Physics, Volume 326, Issue 1, p. 96-192*, 2011.

[22] Shi, Duan, and Vidal. Classical simulation of quantum many-body systems with a tree tensor network. *Physical Review A*, 2005.

[23] 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*, 2009.

[24] M. Troyer and U.J. Wiese. Computational complexity and fundamental limitations to fermionic quantum monte carlo simulations. *Phys. Rev. Lett. 94, 170201-170205*, 2005.

[25] F. Verstrate and J. I. Cirac. Renormalization algorithms for quantum-many body systems in two and higher dimensions. *arXiv:cond-mat/0407066*, 2004.

[26] F. Verstrate, V. Murg, and J. I. Cirac. Matrix product states, projected entangled pair states, and variational renormalization group methods for quantum spin systems. *Advances in Physics Volume 57, Issue 2,p. 143-224*, 2008.

[27] G. Vidal. Classical simulation of infinite-size quantum lattice systems in one spatial dimension. *Phys. Rev. Lett. 98, 070201-070205*, 2004.

[28] G. Vidal. Efficient simulation of one-dimensional quantum many-body systems. *Phys. Rev. Lett. 93, 040502-040506*, 2004.

[29] Guifre Vidal. Entanglement renormalization. *Phys. Rev. Lett. 99, 220405*, 2007.

[30] Guifre Vidal. Class of quantum many-body states that can be efficiently simulated. *Phys. Rev. Lett. 101, 110501*, 2008.

[31] A. Weichselbaum, F. Verstraete, U. Schollwöck, J. I. Cirac, and Jan von Delft. Variational matrix-product-state approach to quantum impurity models. *PHYSICAL REVIEW B 80, 165117-165124*, 2009.

[32] Steven White. Density matrix formulation for quantum renormalization groups. *Phys. Rev. Lett. 69, 2863-2866*, 1992.

[33] Kenneth Wilson. The renormalization group: Critical phenomena and the kondo problem. *Rev. Mod. Phys. 47, 773840*, 1975.