Time evolution of matrix product operators in the Heisenberg picture

Jarno van der Kolk
Supervisor: Prof. U. Schollwöck

March 2, 2011

Abstract

Using the Density Matrix Renormalization Group (DMRG) one can simulate spin systems without the computation time growing as 2^N, making it feasible to simulate large systems. In this paper, the impact of using the DMRG in the Schrödinger picture and in the Heisenberg picture is investigated for three different models, a spin-$\frac{1}{2}$ XX chain, a spin-$\frac{1}{2}$ XXZ chain and a Bose-Hubbard model. Simulations in the Heisenberg model will turn out to perform better only in the XX chain for certain operators. For all the other tested models, the Schrödinger picture is to be preferred. We will also try to shed some light on why the simulations in the Heisenberg picture fail so quickly.

Contents

Chapter 1

Introduction

Simulating complex spin systems is challenging because the number of calculations grows exponentially with the size of the system. Systems of more than 40 spin-$\frac{1}{2}$ particles cannot be simulated exactly even by the current supercomputers. Directly diagonalizing the Hamiltonian would mean finding the eigenvalues and eigenvectors of an $s^N \times s^N$ matrix, where N is the number of particles and s is the number of possible spins a particle can have, i.e. 2 for a spin-$\frac{1}{2}$ particle. A spin-$\frac{1}{2}$ system consisting of 40 particles would already make a matrix with $2^{40} \times 2^{40} = 1.2 \times 10^{24}$ elements when disregarding any symmetries. However, there are some approaches that give insight into larger systems. In our case, we will use the Density Matrix Renormalization Group (DMRG), in which states and operators are written as Matrix Product States (MPS) and Matrix Product Operators (MPO) respectively. The idea is to write all of the coefficients in some superposition as a product of matrices. One matrix will be used for every site and possible spin value per site. Operators can be written in a similar fashion. Applying an operator to a state will amount to multiplying the matrices at every site. This multiplication will however increase the dimensions of the resulting state by a factor d, which is the dimension of the MPO. This will cause the state to grow exponentially when repeatedly applying operators, since after applying and MPO T times, one will have a increase in dimensions of d^T. The approximation consists of performing a singular value decomposition (SVD) of the matrices and dropping the lowest singular values. One is left with a state consisting of smaller matrices that nonetheless closely resembles the original state. The same holds for applying operators on other operators.

This method has been used by the group of Ulrich Schollwöck for studying the time evolution of various initial states. Up until now this was done in the Schrödinger picture. This thesis will explore the Heisenberg picture, which generally gives better results for single site operators, albeit at the price of a longer computation time. To that end, three models will be studied: a 1-dimensional spin-$\frac{1}{2}$ XX chain, a 1-dimensional spin-$\frac{1}{2}$ XXZ chain and a 1-dimensional Bose-Hubbard model. The first model is chosen because an analytical solution exists for the XX chain, which allows us to examine how accurate the simulations are in both the Schrödinger and Heisenberg picture. This first model will turn out to have the 'disadvantage' that the time evolving MPO will be an exact solution in the Heisenberg picture, which means that the accuracy will not degrade due

to the aforementioned approximation. This is however a rare coincidence and to study the behaviour in the Heisenberg picture properly, the second model was chosen. Finally, the third model is chosen because the Heisenberg picture could potentially make a big difference there, because the current simulations in the Schrödinger picture fail quickly in Bose-Hubbard model. If the simulation in the Heisenberg picture turns out to outperform the simulations in the Schrödinger picture significantly, then that would give us a better tool to study such systems.

Why would one expect there to be a difference between calculating an expectation value in the Schrödinger picture and calculating the same value in the Heisenberg picture? At first glance, simulating in the Heisenberg picture has the disadvantage that more calculations have to be done and that one can only study one operator at a time. The answer to that comes from entanglement. When the initial state is an entangled state, then the algorithm that compresses the MPS will throw away information about that entanglement. A single site operator in the Heisenberg picture, however, starts out without knowledge of entanglement in the state, since it is only a local operator and works on one site only. Of course, during the time evolution 'entanglement' will be generated, but the hope is that the compression algorithm for the MPO does not throw away much information, potentially resulting in better results in the Heisenberg picture. If the operator in question is a two-site operator, such as correlation or current density, then the operator is entangled and the results in the Heisenberg picture could degrade significantly. It is therefore worth examining both pictures quantitatively.

Chapter 2

Overview of the DMRG and MPSs

The DMRG, albeit still without MPSs, was first presented by White [1, 2] in 1992 as an improvement to the numerical renormalization group procedure by Wilson[3, 4]. Instead or renormalizing the Hamiltonian, he focused on renormalizing the reduced density matrices (hence the name) of subsystems of the state, by keeping only the states that were the most probable to be used in the final state. The DMRG has helped to deal with the complexities of many-body problems. He demonstrated the use of the DMRG with simulations of Heisenberg antiferromagnetic spin chains with $S = \frac{1}{2}$ and $S = 1$, while being able to use relatively few computational resources compared to methods like Monte Carlo calculations and achieving a much greater accuracy on top of that. This led to a massive adoption of the new method with the original article by White currently being cited well over a thousand times. The DMRG was quickly adopted and already the next year his method was successfully used to show evidence of Bose condensation in the $S = 1$ antiferromagnetic chain in a strong field[5]. Followed by insights in Mott insulators and superfluidity[6] and even for the spin quantum Hall effect in unconventional superconductors[7], the DMRG method has been very successful. Starting from 1995 the link between the DMRG and MPSs was demonstrated[8, 9, 10], allowing for a more elegant description. Since 2004 time evolution came under investigation using Trotter steps[11, 12, 13]. The application of the DMRG in the Heisenberg picture only came along in 2009, showing that it can sometimes be more efficient to do the numerical simulations in the Heisenberg picture[14, 15, 16]. In fact, in some models, certain local operators can even be simulated exactly in the Heisenberg picture[17]. A detailed description about the current state of affairs in the DMRG can be found in the recent review article by Schollwöck[18].

Chapter 3

Mathematical framework

Here, we will outline the calculations and approximations that are required for working with matrix product operators and matrix product states. Most of the calculations are just matrix multiplications and reordering of indices, but even so, they are required to translate physical states and operators into matrices. For simplicity we will only discuss the case of the spin-$\frac{1}{2}$ system, where the σ_i stand for either spin up or down. In the case of the Bose-Hubbard model, one simply uses occupation number instead of spin.

3.1 Matrix product notation

Normally one could write any state as an interpolation of all basis states. For example if one were to pick $|\sigma_1, \sigma_2, \dots \sigma_N\rangle$ as your basis, then any arbitrary state could be written as:

$$\sum_{\boldsymbol{\sigma}} c_{\boldsymbol{\sigma}} |\boldsymbol{\sigma}\rangle \tag{3.1}$$

This sum would run over all the possible combinations for every σ_i, i.e. 2^N indices. This is not feasible for many particles, so we need an alternative way of representing states. A state of a system consisting of N sites can also be represented by matrices as a Matrix Product State (MPS).

$$|\psi\rangle = p \sum_{\boldsymbol{\sigma}} \sum_{a_1,\dots} A^{\sigma_1}_{a_0,a_1} A^{\sigma_2}_{a_1,a_2} A^{\sigma_3}_{a_2,a_3} \dots A^{\sigma_N}_{a_{N-1},a_N} |\boldsymbol{\sigma}\rangle \tag{3.2}$$

The a_0 and a_N are dummy index that can only be 1. The only reason it is not just written as 1, is that it is aesthetically more pleasing. The whole state is written as a sum of the basis states $|\boldsymbol{\sigma}\rangle$, which is a short hand notation for $|\sigma_1, \sigma_2, \sigma_3, \dots, \sigma_N\rangle$. σ_i is simply the spin at spin site i. Compare this to the usual way of writing an arbitrary state in its base components, $\sum c_{\boldsymbol{\sigma}} |\boldsymbol{\sigma}\rangle$. The coefficient, $c_{\boldsymbol{\sigma}}$, for each basis state can be obtained by multiplying the matrices A^{σ_i} and the prefactor p.

$$c_{\boldsymbol{\sigma}} = p A^{\sigma_1} A^{\sigma_2} A^{\sigma_3} \dots A^{\sigma_N} \tag{3.3}$$

Depending on the state in question, the MPS way of writing down the state is far more efficient compared to keeping all of the $c_{\boldsymbol{\sigma}}$-indices of all the basis

states in the superposition separately. As an example, let us look at a system of three spin-$\frac{1}{2}$ sites. Suppose we want to describe a state $\frac{1}{\sqrt{2}}(|\uparrow\uparrow\downarrow\rangle + |\downarrow\downarrow\uparrow\rangle)$. This state can then be described by the following six matrices with $p = \frac{1}{\sqrt{2}}$.

$$\begin{aligned} A^{\uparrow_1} &= \begin{pmatrix} 1 & 0 \end{pmatrix} \quad A^{\uparrow_2} = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \quad A^{\uparrow_3} = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \\ A^{\downarrow_1} &= \begin{pmatrix} 0 & 1 \end{pmatrix} \quad A^{\downarrow_2} = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix} \quad A^{\downarrow_3} = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \end{aligned} \tag{3.4}$$

There is no unique choice of matrices to describe a state. This is clear from the above example; one could easily swap $\uparrow$ with $\downarrow$ in the above matrices and still end up with the same state. Also the dimensions of the matrices are arbitrary (up to some minimum of course), although one obviously strives to use the smallest matrices possible to keep the number of calculations to a minimum.
When working with kets, one of course also needs bras. Simply taking the complex conjugate of (3.2) results in the bra.

$$\langle\psi| = p^\star \sum_{\boldsymbol{\sigma}} \sum_{a_1,\ldots} (A^{\sigma_N\dagger})_{a_N,a_{N-1}} \ldots (A^{\sigma_3\dagger})_{a_3,a_2} (A^{\sigma_2\dagger})_{a_2,a_1} (A^{\sigma_1\dagger})_{a_1,a_0} \langle\boldsymbol{\sigma}| \tag{3.5}$$

Completely analogous with the MPS, one can also construct a Matrix Product Operator (MPO)[19]. An MPO for a system containing N sites can be written as:

$$\hat{O} = q \sum_{\boldsymbol{\sigma},\boldsymbol{\sigma}'} \sum_{b_1,\ldots} B^{\sigma_1,\sigma_1'}_{b_0,b_1} B^{\sigma_2,\sigma_2'}_{b_1,b_2} B^{\sigma_3,\sigma_3'}_{b_2,b_3} \ldots B^{\sigma_N,\sigma_N'}_{b_{N-1},b_N} |\boldsymbol{\sigma}\rangle\langle\boldsymbol{\sigma}'| \tag{3.6}$$

Again the b_0 and b_N are just a dummy indices, equal to 1.

3.2 Overlap

To do anything meaningful with MPOs and MPSs one needs to be able to calculate overlaps $\langle\tilde{\psi}|\psi\rangle$. Calculating the overlap is nothing more than multiplying all the matrices from (3.2) and (3.5).

$$\begin{aligned} &\langle\tilde{\psi}|\psi\rangle \\ &= \tilde{p}^\star p \sum_{\boldsymbol{\sigma},\tilde{\boldsymbol{\sigma}}} \sum_{a_1,\ldots} \sum_{\tilde{a}_1,\ldots} \ldots (\tilde{A}^{\tilde{\sigma}_3\dagger})_{\tilde{a}_3,\tilde{a}_2} (\tilde{A}^{\tilde{\sigma}_2\dagger})_{\tilde{a}_2,\tilde{a}_1} (\tilde{A}^{\tilde{\sigma}_1\dagger})_{\tilde{a}_1,\tilde{a}_0} A^{\sigma_1}_{a_0,a_1} A^{\sigma_2}_{a_1,a_2} A^{\sigma_3}_{a_2,a_3} \ldots \langle\tilde{\boldsymbol{\sigma}}|\boldsymbol{\sigma}\rangle \\ &= \tilde{p}^\star p \sum_{\boldsymbol{\sigma}} \sum_{a_1,\ldots} \sum_{\tilde{a}_1,\ldots} \ldots (\tilde{A}^{\sigma_3\dagger})_{\tilde{a}_3,\tilde{a}_2} (\tilde{A}^{\sigma_2\dagger})_{\tilde{a}_2,\tilde{a}_1} (\tilde{A}^{\sigma_1\dagger})_{\tilde{a}_1,\tilde{a}_0} A^{\sigma_1}_{a_0,a_1} A^{\sigma_2}_{a_1,a_2} A^{\sigma_3}_{a_2,a_3} \ldots \\ &= \tilde{p}^\star p \sum_{\sigma_2,\ldots} \sum_{a_1,\ldots} \sum_{\tilde{a}_1,\ldots} \ldots (\tilde{A}^{\sigma_3\dagger})_{\tilde{a}_3,\tilde{a}_2} (\tilde{A}^{\sigma_2\dagger})_{\tilde{a}_2,\tilde{a}_1} \left(\sum_{\sigma_1} (\tilde{A}^{\sigma_1\dagger})_{\tilde{a}_1,\tilde{a}_0} A^{\sigma_1}_{a_0,a_1} \right) A^{\sigma_2}_{a_1,a_2} A^{\sigma_3}_{a_2,a_3} \ldots \end{aligned}$$

$$
\begin{aligned}
&= \tilde{p}^\star p \sum_{\sigma_2,\ldots} \sum_{a_1,\ldots} \sum_{\tilde{a}_1,\ldots} \ldots (\tilde{A}^{\sigma_3\dagger})_{\tilde{a}_3,\tilde{a}_2} (\tilde{A}^{\sigma_2\dagger})_{\tilde{a}_2,\tilde{a}_1} M^{(1)}_{\tilde{a}_1,a_1} A^{\sigma_2}_{a_1,a_2} A^{\sigma_3}_{a_2,a_3} \ldots \\
&= \tilde{p}^\star p \sum_{\sigma_3,\ldots} \sum_{a_2,\ldots} \sum_{\tilde{a}_2,\ldots} \ldots (\tilde{A}^{\sigma_3\dagger})_{\tilde{a}_3,\tilde{a}_2} \left(\sum_{\sigma_2,a_1,\tilde{a}_1} (\tilde{A}^{\sigma_2\dagger})_{\tilde{a}_2,\tilde{a}_1} M^{(1)}_{\tilde{a}_1,a_1} A^{\sigma_2}_{a_1,a_2} \right) A^{\sigma_3}_{a_2,a_3} \ldots \\
&= \tilde{p}^\star p \sum_{\sigma_3,\ldots} \sum_{a_2,\ldots} \sum_{\tilde{a}_2,\ldots} \ldots (\tilde{A}^{\sigma_3\dagger})_{\tilde{a}_3,\tilde{a}_2} M^{(2)}_{\tilde{a}_2,a_2} A^{\sigma_3}_{a_2,a_3} \ldots \\
&= \ldots \\
&= \tilde{p}^\star p \, M^N_{\tilde{a}_N,a_N}
\end{aligned} \tag{3.7}
$$

where we have introduced $M^{(i)}_{\tilde{a}_i,a_i} = \sum_{\sigma_i,a_1,\tilde{a}_i} (\tilde{A}^{\sigma_i\dagger})_{\tilde{a}_i,\tilde{a}_{i-1}} M^{(i-1)}_{\tilde{a}_{i-1},a_{i-1}} A^{\sigma_i}_{a_{i-1},a_i}$. Since $\tilde{a}_N$ and a_N are dummy variables, the result is simply a scalar, which stands for the overlap between $|\tilde{\psi}\rangle$ and $|\psi\rangle$.

3.3 Applying an MPO to an MPS

To calculate expectation values and matrix elements we have to be able to apply an MPO to an MPS. If we write out $\hat{O}|\psi\rangle$ using (3.2) and (3.6) we get:

$$
\begin{aligned}
\hat{O}|\psi\rangle &= qp \sum_{\boldsymbol{\sigma},\boldsymbol{\sigma}',\boldsymbol{\sigma}''} \sum_{b_1,\ldots} \sum_{a_1,\ldots} B^{\sigma_1,\sigma_1'}_{b_0,b_1} B^{\sigma_2,\sigma_2'}_{b_1,b_2} \ldots A^{\sigma_1''}_{a_0,a_1} A^{\sigma_2''}_{a_1,a_2} \ldots |\boldsymbol{\sigma}\rangle\langle\boldsymbol{\sigma}'|\boldsymbol{\sigma}''\rangle \\
&= qp \sum_{\boldsymbol{\sigma},\boldsymbol{\sigma}'} \sum_{b_1,\ldots} \sum_{a_1,\ldots} B^{\sigma_1,\sigma_1'}_{b_0,b_1} B^{\sigma_2,\sigma_2'}_{b_1,b_2} \ldots A^{\sigma_1'}_{a_0,a_1} A^{\sigma_2'}_{a_1,a_2} \ldots |\boldsymbol{\sigma}\rangle \\
&= qp \sum_{\boldsymbol{\sigma}} \sum_{(b_1,a_1),\ldots} \left(\sum_{\sigma_1'} B^{\sigma_1,\sigma_1'}_{b_0,b_1} A^{\sigma_1'}_{a_0,a_1} \right) \left(\sum_{\sigma_2'} B^{\sigma_2,\sigma_2'}_{b_1,b_2} A^{\sigma_2'}_{a_1,a_2} \right) \ldots |\boldsymbol{\sigma}\rangle \\
&= \tilde{p} \sum_{\boldsymbol{\sigma}} \sum_{\tilde{a}_1,\ldots} \tilde{A}^{\sigma_1}_{\tilde{a}_0,\tilde{a}_1} \tilde{A}^{\sigma_2}_{\tilde{a}_1,\tilde{a}_2} \tilde{A}^{\sigma_3}_{\tilde{a}_2,\tilde{a}_3} \ldots |\boldsymbol{\sigma}\rangle
\end{aligned} \tag{3.8}
$$

In the last step we set $\tilde{p} = qp$, $\tilde{a}_{i+jN} = (b_j, a_i)$ and $\tilde{A}^{\sigma_i}_{\tilde{a}_{i-1},\tilde{a}_i} = \sum_{\sigma_i'} B^{\sigma_i,\sigma_i'}_{b_{i-1},b_i} A^{\sigma_i'}_{a_{i-1},a_i}$. The resulting MPS is bigger than the original MPS, because the dimensions of the matrices of the new MPS are the dimensions of the matrices of the MPO multiplied by the dimensions of the matrices of the old MPS. Applying an MPO to an MPS repeatedly, like one would do with time evolution, would blow up the dimensions of matrices in the MPS. This exponential growth is the main problem in time evolution in the DMRG. From this it is clear that some form of approximation will be necessary. This will be discussed in section 3.7.

3.4 Multiplying MPOs

Multiplying two MPOs is basically the same as applying an MPO to an MPS. The only difference is that there is an extra spin index.

$$
\hat{O} = \hat{O}_1 \cdot \hat{O}_2 \tag{3.9}
$$

Using

$$\hat{O}_1 = q_1 \sum_{\boldsymbol{\sigma},\boldsymbol{\sigma}'} \sum_{c_1,\ldots} C^{\sigma_1,\sigma_1'}_{c_0,c_1} C^{\sigma_2,\sigma_2'}_{c_1,c_2} C^{\sigma_3,\sigma_3'}_{c_2,c_3} \ldots |\boldsymbol{\sigma}\rangle\langle\boldsymbol{\sigma}'| \tag{3.10}$$

$$\hat{O}_2 = q_2 \sum_{\boldsymbol{\sigma}'',\boldsymbol{\sigma}'''} \sum_{d_1,\ldots} D^{\sigma_1'',\sigma_1'''}_{d_0,d_1} D^{\sigma_2'',\sigma_2'''}_{d_1,d_2} D^{\sigma_3'',\sigma_3'''}_{d_2,d_3} \ldots |\boldsymbol{\sigma}''\rangle\langle\boldsymbol{\sigma}'''| \tag{3.11}$$

we will end up with

$$\begin{aligned}
\hat{O} &= q_1 q_2 \sum_{\boldsymbol{\sigma},\boldsymbol{\sigma}',\boldsymbol{\sigma}'',\boldsymbol{\sigma}'''} \sum_{c_1,\ldots} \sum_{d_1,\ldots} C^{\sigma_1,\sigma_1'}_{c_0,c_1} C^{\sigma_2,\sigma_2'}_{c_1,c_2} \ldots D^{\sigma_1'',\sigma_1'''}_{d_0,d_1} D^{\sigma_2'',\sigma_2'''}_{d_1,d_2} \ldots |\boldsymbol{\sigma}\rangle\langle\boldsymbol{\sigma}'|\boldsymbol{\sigma}''\rangle\langle\boldsymbol{\sigma}'''| \\
&= q_1 q_2 \sum_{\boldsymbol{\sigma},\boldsymbol{\sigma}',\boldsymbol{\sigma}'''} \sum_{c_1,\ldots} \sum_{d_1,\ldots} C^{\sigma_1,\sigma_1'}_{c_0,c_1} C^{\sigma_2,\sigma_2'}_{c_1,c_2} \ldots D^{\sigma_1',\sigma_1'''}_{d_0,d_1} D^{\sigma_2',\sigma_2'''}_{d_1,d_2} \ldots |\boldsymbol{\sigma}\rangle\langle\boldsymbol{\sigma}'''| \\
&= q_1 q_2 \sum_{\boldsymbol{\sigma},\boldsymbol{\sigma}'''} \sum_{(c_1,d_1),\ldots} \left(\sum_{\sigma_1'} C^{\sigma_1,\sigma_1'}_{c_0,c_1} D^{\sigma_1',\sigma_1'''}_{d_0,d_1} \right) \left(\sum_{\sigma_2'} C^{\sigma_2,\sigma_2'}_{c_1,c_2} D^{\sigma_2',\sigma_2'''}_{d_1,d_2} \right) \ldots |\boldsymbol{\sigma}\rangle\langle\boldsymbol{\sigma}'''| \\
&= q \sum_{\boldsymbol{\sigma},\boldsymbol{\sigma}'''} \sum_{b_1,\ldots} B^{\sigma_1,\sigma_1'''}_{b_0,b_1} B^{\sigma_2,\sigma_2'''}_{b_1,b_2} B^{\sigma_3,\sigma_3'''}_{b_2,b_3} \ldots |\boldsymbol{\sigma}\rangle\langle\boldsymbol{\sigma}'''|
\end{aligned} \tag{3.12}$$

Just as in the previous section, we again set $q = q_1 q_2$, $b_{i+jN} = (c_j, d_i)$ and $B^{\sigma_i,\sigma_i'''}_{b_{i-1},b_i} = \sum_{\sigma_i'} C^{\sigma_i,\sigma_i'}_{c_{i-1},c_i} D^{\sigma_i',\sigma_i'''}_{d_{i-1},d_i}$ in the last step. This increases the dimension of the B-matrices compared to the original C- and D-matrices, since c_i and d_i run only from 1 to N, but b_i runs from 1 to N^2. Therefore, we will also need to look at compressing MPOs in section 3.7.

3.5 Canonizing an MPS

For future uses, it would be convenient to normalize the constituent matrices of an MPS, since this can simplify calculations. There are two norms we can use for this.

- Left normalization:
$$\sum_{\sigma_i} A^{\sigma_i\dagger} A^{\sigma_i} = 1 \tag{3.13}$$
- Right normalization:
$$\sum_{\sigma_i} A^{\sigma_i} A^{\sigma_i\dagger} = 1 \tag{3.14}$$

Left-canonizing consists of left-normalizing all the matrices in the MPS. The procedure for this is to regroup the indices and apply a QR decomposition. See appendix B.1 for more details. The decomposition will split the matrix A in QR, where Q is a unitary matrix and R is an upper triangular matrix with entries only on and above the diagonal. The Q is reshaped in the form of the

original A, and R is multiplied into the next A.

$$\begin{aligned}
|\psi\rangle &= p \sum_{\boldsymbol{\sigma}} \sum_{a_1,\ldots} A_{(\sigma_1,a_0),a_1} A^{\sigma_2}_{a_1,a_2} A^{\sigma_3}_{a_2,a_3} \ldots |\boldsymbol{\sigma}\rangle \\
&= p \sum_{\boldsymbol{\sigma}} \sum_{a_1,\ldots} \sum_{r_1} Q_{(\sigma_1,r_0),r_1} R_{r_1,a_1} A^{\sigma_2}_{a_1,a_2} A^{\sigma_3}_{a_2,a_3} \ldots |\boldsymbol{\sigma}\rangle \\
&= p \sum_{\boldsymbol{\sigma}} \sum_{a_1,\ldots} \sum_{r_1} Q^{\sigma_1}_{r_0,r_1} \left(R_{r_1,a_1} A^{\sigma_2}_{a_1,a_2} \right) A^{\sigma_3}_{a_2,a_3} \ldots |\boldsymbol{\sigma}\rangle \\
&= p \sum_{\boldsymbol{\sigma}} \sum_{a_2,\ldots} \sum_{r_1} Q^{\sigma_1}_{r_0,r_1} C^{\sigma_2}_{r_1,a_2} A^{\sigma_3}_{a_2,a_3} \ldots |\boldsymbol{\sigma}\rangle \quad , \qquad (3.15)
\end{aligned}$$

where $C^{\sigma_2}_{r_1,a_2} = \sum_{a_1} R_{r_1,a_1} A^{\sigma_2}_{a_1,a_2}$. In the second line we silently renamed the dummy index a_0 into the dummy index r_0. By repeating this process from A^{σ_1} to $A^{\sigma_{N-1}}$, one can left-normalize all the matrices in the MPS. The left-normalizing of the last step requires some special attention, since there is no subsequent matrix in which to multiply the R. However, in the last step the dimensions of R are 1×1, which means it is only a scalar. This scalar can be multiplied into the prefactor p.

$$\begin{aligned}
|\psi\rangle &= p \sum_{\boldsymbol{\sigma}} \sum_{r_1\ldots} \ldots Q^{\sigma_{N-1}}_{r_{N-2},r_{N-1}} C^{\sigma_N}_{r_{N-1},a_N} |\boldsymbol{\sigma}\rangle \\
&= p \sum_{\boldsymbol{\sigma}} \sum_{r_1\ldots} \ldots Q^{\sigma_{N-1}}_{r_{N-2},r_{N-1}} C_{(\sigma_N,r_{N-1}),a_N} |\boldsymbol{\sigma}\rangle \\
&= p \sum_{\boldsymbol{\sigma}} \sum_{r_1\ldots} \ldots Q^{\sigma_{N-1}}_{r_{N-2},r_{N-1}} Q_{(\sigma_N,r_N-1),r_N} R_{r_N,a_N} |\boldsymbol{\sigma}\rangle \\
&= \tilde{p} \sum_{\boldsymbol{\sigma}} \sum_{r_1\ldots} \ldots Q^{\sigma_{N-1}}_{r_{N-2},r_{N-1}} Q^{\sigma_N}_{r_N-1,r_N} |\boldsymbol{\sigma}\rangle \qquad (3.16)
\end{aligned}$$

In the last step, we set the prefactor $\tilde{p} = p \cdot R_{r_N,a_N}$. From (3.7), it is clear that this prefactor is 1 for a normalized state, since it has to obey $\langle\psi|\psi\rangle = 1$ and due to the left normalization from (3.13), all the M-matrices in (3.7) are only identity matrices.

Right canonizing an MPS works in a similar fashion. Instead of using the QR decomposition we will use the LQ decomposition, which splits any matrix into a lower triangular matrix L and a unitary matrix Q. Also, the matrices in

the MPS have to be processed from last until first.

$$
\begin{aligned}
|\psi\rangle &= p \sum_{\boldsymbol{\sigma}} \sum_{\ldots,a_{N-1}} \ldots A^{\sigma_{N-2}}_{a_{N-3},a_{N-2}} A^{\sigma_{N-1}}_{a_{N-2},a_{N-1}} A^{\sigma_N}_{a_{N-1},a_N} |\boldsymbol{\sigma}\rangle \\
&= p \sum_{\boldsymbol{\sigma}} \sum_{\ldots,a_{N-1}} \ldots A^{\sigma_{N-2}}_{a_{N-3},a_{N-2}} A^{\sigma_{N-1}}_{a_{N-2},a_{N-1}} A_{a_{N-1},(\sigma_N,a_N)} |\boldsymbol{\sigma}\rangle \\
&= p \sum_{\boldsymbol{\sigma}} \sum_{\ldots,a_{N-1}} \sum_{\ell_{N-1}} \ldots A^{\sigma_{N-2}}_{a_{N-3},a_{N-2}} A^{\sigma_{N-1}}_{a_{N-2},a_{N-1}} L_{a_{N-1},\ell_{N-1}} \quad \times \\
&\qquad Q_{\ell_{N-1},(\sigma_N,a_N)} |\boldsymbol{\sigma}\rangle \\
&= p \sum_{\boldsymbol{\sigma}} \sum_{\ldots,a_{N-1}} \sum_{\ell_{N-1}} \ldots A^{\sigma_{N-2}}_{a_{N-3},a_{N-2}} \left(A^{\sigma_{N-1}}_{a_{N-2},a_{N-1}} L_{a_{N-1},\ell_{N-1}} \right) \quad \times \\
&\qquad Q_{\ell_{N-1},(\sigma_N,a_N)} |\boldsymbol{\sigma}\rangle \\
&= p \sum_{\boldsymbol{\sigma}} \sum_{\ldots,a_{N-2}} \sum_{\ell_{N-1}} \ldots A^{\sigma_{N-2}}_{a_{N-3},a_{N-2}} C^{\sigma_{N-1}}_{a_{N-2},\ell_{N-1}} Q^{\sigma_N,\sigma'_N}_{\ell_{N-1},a_N} |\boldsymbol{\sigma}\rangle
\end{aligned} \tag{3.17}
$$

Repeating this process from A^{σ_N} to A^{σ_1} results in an MPO containing only right-normalized matrices. In the final step we are once again left with a scalar, $L_{1,1}$, which can be absorbed into the prefactor p.

3.6 Canonizing an MPO

By using the same procedure as for the canonizing of the MPS, the MPO can also be canonized. First one needs to define a new norm, since there is an extra spin index involved.

- Left normalization:
$$\sum_{\sigma_i,\sigma'_i} B^{\sigma_i,\sigma'_i\dagger} B^{\sigma_i,\sigma'_i} = 1 \tag{3.18}$$
- Right normalization:
$$\sum_{\sigma_i,\sigma'_i} B^{\sigma_i,\sigma'_i} B^{\sigma_i,\sigma'_i\dagger} = 1 \tag{3.19}$$

The canonizing can be done by reshaping $B^{\sigma_1,\sigma'_1}_{1,b_1}$ to $B_{(\sigma_1,\sigma'_1,1),b_1}$ and, as before, performing a QR decomposition on this.

$$
\begin{aligned}
\hat{O} &= q \sum_{\boldsymbol{\sigma},\boldsymbol{\sigma'}} \sum_{b_1,\ldots} B_{(\sigma_1,\sigma'_1,b_0),b_1} B^{\sigma_2,\sigma'_2}_{b_1,b_2} B^{\sigma_3,\sigma'_3}_{b_2,b_3} \ldots |\boldsymbol{\sigma}\rangle\langle\boldsymbol{\sigma'}| \\
&= q \sum_{\boldsymbol{\sigma},\boldsymbol{\sigma'}} \sum_{b_1,\ldots} \sum_{r_1} Q_{(\sigma_1,\sigma'_1,r_0),r_1} R_{r_1,b_1} B^{\sigma_2,\sigma'_2}_{b_1,b_2} B^{\sigma_3,\sigma'_3}_{b_2,b_3} \ldots |\boldsymbol{\sigma}\rangle\langle\boldsymbol{\sigma'}| \\
&= q \sum_{\boldsymbol{\sigma},\boldsymbol{\sigma'}} \sum_{b_1,\ldots} \sum_{r_1} Q^{\sigma_1,\sigma'_1}_{r_0,r_1} \left(R_{r_1,b_1} B^{\sigma_2,\sigma'_2}_{b_1,b_2} \right) B^{\sigma_3,\sigma'_3}_{b_2,b_3} \ldots |\boldsymbol{\sigma}\rangle\langle\boldsymbol{\sigma'}| \\
&= q \sum_{\boldsymbol{\sigma},\boldsymbol{\sigma'}} \sum_{b_2,\ldots} \sum_{r_1} Q^{\sigma_1,\sigma'_1}_{r_0,r_1} C^{\sigma_2,\sigma'_2}_{r_1,b_2} B^{\sigma_3,\sigma'_3}_{b_2,b_3} \ldots |\boldsymbol{\sigma}\rangle\langle\boldsymbol{\sigma'}|
\end{aligned} \tag{3.20}
$$

This results in a new MPO with $C^{\sigma_2,\sigma'_2}_{r_1,b_2} = \sum_{b_1} R_{r_1,b_1} B^{\sigma_2,\sigma'_2}_{b_1,b_2}$. Also note that we renamed b_0 into r_0, which is allowed since both are dummy indices anyway.

This process can be iterated so that the end MPO contains only left-normalized $Q^{\sigma_i,\sigma_i'}_{s_{i-1},s_i}$ matrices. The last step needs a bit of special treatment as before.

$$\begin{aligned}
\hat{O} &= q \sum_{\boldsymbol{\sigma},\boldsymbol{\sigma'}} \sum_{s_1 \ldots} \ldots Q^{\sigma_{N-1},\sigma'_{N-1}}_{r_{N-2},r_{N-1}} C^{\sigma_N,\sigma'_N}_{s_{N-1},b_N} |\boldsymbol{\sigma}\rangle\langle\boldsymbol{\sigma'}| \\
&= q \sum_{\boldsymbol{\sigma},\boldsymbol{\sigma'}} \sum_{s_1 \ldots} \ldots Q^{\sigma_{N-1},\sigma'_{N-1}}_{r_{N-2},r_{N-1}} C_{(\sigma_N,\sigma'_N,r_{N-1}),b_N} |\boldsymbol{\sigma}\rangle\langle\boldsymbol{\sigma'}| \\
&= q \sum_{\boldsymbol{\sigma},\boldsymbol{\sigma'}} \sum_{r_1 \ldots} \ldots Q^{\sigma_{N-1},\sigma'_{N-1}}_{s_{N-2},s_{N-1}} Q_{(\sigma_N,\sigma'_N,r_N-1),r_N} R_{r_N,b_N} |\boldsymbol{\sigma}\rangle\langle\boldsymbol{\sigma'}| \\
&= \tilde{q} \sum_{\boldsymbol{\sigma},\boldsymbol{\sigma'}} \sum_{r_1 \ldots} \ldots U^{\sigma_{N-1},\sigma'_{N-1}}_{r_{N-2},r_{N-1}} Q^{\sigma_N,\sigma'_N}_{r_N-1,r_N} |\boldsymbol{\sigma}\rangle\langle\boldsymbol{\sigma'}| \quad , \qquad (3.21)
\end{aligned}$$

where $\tilde{q} = q \cdot R_{r_N,b_N}$ is simply a scalar, since r_N and b_N are both dummy indices.

For right-canonizing the same principle applies as for left-canonizing, but instead of starting on the first matrix, we now start on the last matrix, $B^{\sigma_N,\sigma'_N}_{b_{N-1},b_N}$, keeping all the L's from the LQ decomposition.

$$\begin{aligned}
\hat{O} &= q \sum_{\boldsymbol{\sigma},\boldsymbol{\sigma'}} \sum_{\ldots,b_{N-1}} \ldots B^{\sigma_{N-2},\sigma'_{N-2}}_{b_{N-3},b_{N-2}} B^{\sigma_{N-1},\sigma'_{N-1}}_{b_{N-2},b_{N-1}} B^{\sigma_N,\sigma'_N}_{b_{N-1},b_N} |\boldsymbol{\sigma}\rangle\langle\boldsymbol{\sigma'}| \\
&= q \sum_{\boldsymbol{\sigma},\boldsymbol{\sigma'}} \sum_{\ldots,b_{N-1}} \ldots B^{\sigma_{N-2},\sigma'_{N-2}}_{b_{N-3},b_{N-2}} B^{\sigma_{N-1},\sigma'_{N-1}}_{b_{N-2},b_{N-1}} B_{b_{N-1},(\sigma_N,\sigma'_N,b_N)} |\boldsymbol{\sigma}\rangle\langle\boldsymbol{\sigma'}| \\
&= q \sum_{\boldsymbol{\sigma},\boldsymbol{\sigma'}} \sum_{\ldots,b_{N-1}} \sum_{\ell_{N-1}} \ldots B^{\sigma_{N-2},\sigma'_{N-2}}_{b_{N-3},b_{N-2}} B^{\sigma_{N-1},\sigma'_{N-1}}_{b_{N-2},b_{N-1}} L_{b_{N-1},\ell_{N-1}} \quad \times \\
&\qquad Q_{\ell_{N-1},(\sigma_N,\sigma'_N,b_N)} |\boldsymbol{\sigma}\rangle\langle\boldsymbol{\sigma'}| \\
&= q \sum_{\boldsymbol{\sigma},\boldsymbol{\sigma'}} \sum_{\ldots,b_{N-1}} \sum_{\ell_{N-1}} \ldots B^{\sigma_{N-2},\sigma'_{N-2}}_{b_{N-3},b_{N-2}} \left(B^{\sigma_{N-1},\sigma'_{N-1}}_{b_{N-2},b_{N-1}} L_{b_{N-1},\ell_{N-1}} \right) \quad \times \\
&\qquad Q_{\ell_{N-1},(\sigma_N,\sigma'_N,b_N)} |\boldsymbol{\sigma}\rangle\langle\boldsymbol{\sigma'}| \\
&= q \sum_{\boldsymbol{\sigma},\boldsymbol{\sigma'}} \sum_{\ldots,b_{N-2}} \sum_{\ell_{N-1}} \ldots B^{\sigma_{N-2},\sigma'_{N-2}}_{b_{N-3},b_{N-2}} C^{\sigma_{N-1},\sigma'_{N-1}}_{b_{N-2},\ell_{N-1}} (V^\dagger)^{\sigma_N,\sigma'_N}_{\ell_{N-1},b_N} |\boldsymbol{\sigma}\rangle\langle\boldsymbol{\sigma'}| \qquad (3.22)
\end{aligned}$$

Repeating this process results in an MPO containing only right normalized matrices. In the final step the remaining scalar is again absorbed into the prefactor $\tilde{q} = q \cdot U_{b_N,\ell_N}$.

3.7 Compression

To speed up calculations involving MPOs and MPSs, it is crucial to try to make the constituent matrices of the MPOs or MPSs smaller. This is in fact the very essence of the DMRG: discarding the states that matter least. In order to reduce the size of the matrices, we need some sort of compression algorithm. We can further enhance that algorithm by trying to minimize the 'distance' between the original MPS/MPO and the new compressed one.

3.7.1 Singular value compression

The first choice would be to look at the canonization process in section 3.5 and section 3.6. If instead of a QR/LQ decomposition one were to employ a Singular Value Decomposition (SVD), one obtains a tool to decrease the size of the matrices in an MPO/MPS, as we will see. An SVD splits any matrix into two unitary matrices and a diagonal matrix, $A = USV^\dagger$, where S is the diagonal matrix containing the singular values of A. See B.2 for more details. For example, the left canonization of an MPS, as was done in (3.15), looks similar using an SVD.

$$\begin{aligned}
|\psi\rangle &= p \sum_{\boldsymbol{\sigma}} \sum_{a_1,\dots} A_{(\sigma_1,a_0),a_1} A^{\sigma_2}_{a_1,a_2} A^{\sigma_3}_{a_2,a_3} \dots |\boldsymbol{\sigma}\rangle \\
&= p \sum_{\boldsymbol{\sigma}} \sum_{a_1,\dots} \sum_{s_1} U_{(\sigma_1,s_0),s_1} S_{s_1,s_1} (V^\dagger)_{s_1,a_1} A^{\sigma_2}_{a_1,a_2} A^{\sigma_3}_{a_2,a_3} \dots |\boldsymbol{\sigma}\rangle \\
&= p \sum_{\boldsymbol{\sigma}} \sum_{a_1,\dots} \sum_{s_1} U^{\sigma_1}_{s_0,s_1} \left(S_{s_1,s_1} (V^\dagger)_{s_1,a_1} A^{\sigma_2}_{a_1,a_2} \right) A^{\sigma_3}_{a_2,a_3} \dots |\boldsymbol{\sigma}\rangle \\
&= p \sum_{\boldsymbol{\sigma}} \sum_{a_2,\dots} \sum_{s_1} U^{\sigma_1}_{s_0,s_1} C^{\sigma_2}_{s_1,a_2} A^{\sigma_3}_{a_2,a_3} \dots |\boldsymbol{\sigma}\rangle
\end{aligned} \tag{3.23}$$

This time $C^{\sigma_2}_{s_1,a_2}$ stands for $\sum_{a_1} S_{s_1,s_1} (V^\dagger)_{s_1,a_1} A^{\sigma_2}_{a_1,a_2}$ and a_0 is renamed to s_0. The right canonizing procedure can be adjusted in the same way by using SVD instead of LQ decomposition, shown this time for an MPO:

$$\begin{aligned}
\hat{O} &= q \sum_{\boldsymbol{\sigma},\boldsymbol{\sigma}'} \sum_{b_1,\dots} B_{(\sigma_1,\sigma_1',b_0),b_1} B^{\sigma_2,\sigma_2'}_{b_1,b_2} B^{\sigma_3,\sigma_3'}_{b_2,b_3} \dots |\boldsymbol{\sigma}\rangle\langle\boldsymbol{\sigma}'| \\
&= q \sum_{\boldsymbol{\sigma},\boldsymbol{\sigma}'} \sum_{b_1,\dots} \sum_{s_1} U_{(\sigma_1,\sigma_1',s_0),s_1} S_{s_1,s_1} (V^\dagger)_{s_1,b_1} B^{\sigma_2,\sigma_2'}_{b_1,b_2} B^{\sigma_3,\sigma_3'}_{b_2,b_3} \dots |\boldsymbol{\sigma}\rangle\langle\boldsymbol{\sigma}'| \\
&= q \sum_{\boldsymbol{\sigma},\boldsymbol{\sigma}'} \sum_{b_1,\dots} \sum_{s_1} U^{\sigma_1,\sigma_1'}_{s_0,s_1} \left(S_{s_1,s_1} (V^\dagger)_{s_1,b_1} B^{\sigma_2,\sigma_2'}_{b_1,b_2} \right) B^{\sigma_3,\sigma_3'}_{b_2,b_3} \dots |\boldsymbol{\sigma}\rangle\langle\boldsymbol{\sigma}'| \\
&= q \sum_{\boldsymbol{\sigma},\boldsymbol{\sigma}'} \sum_{b_2,\dots} \sum_{s_1} U^{\sigma_1,\sigma_1'}_{s_0,s_1} C^{\sigma_2,\sigma_2'}_{s_1,b_2} B^{\sigma_3,\sigma_3'}_{b_2,b_3} \dots |\boldsymbol{\sigma}\rangle\langle\boldsymbol{\sigma}'|
\end{aligned} \tag{3.24}$$

The basis for the compression lies in the singular values in the matrix S. These singular values represent the importance of the associated columns and rows in U and $V^\dagger$ respectively. (There is a deeper connection, since in the case of MPSs, the singular values are equal to the squares of the eigenvalues of the reduced density matrices[10]). Discarding the lowest singular values and associated rows and columns for every matrix in an MPS will therefore result in a good approximation to the original MPO/S. That is, in (3.23) and (3.24), the sum $\sum_{s_1}$ runs only over the d largest singular values with d being the desired maximum dimension of the matrices, i.e. bond dimension. After the compression, all of the matrices in the now left/right canonized MPO/MPS will have a dimension of at most $d \times d$. It will turn out that this compression works remarkably well, even when only retaining a small number of singular values. Discarding some singular values effects the norm of an MPS, but this is easily corrected for by renormalizing.

3.7.2 Iterative compression of an MPS

To further improve the compression, one can try to minimize the 'distance' between an MPS, $|\psi\rangle$, and some reduced MPS, $|\tilde{\psi}\rangle$, to get the reduced MPS as 'close' to the original MPS as possible. For that we will need a measure of distance. The most obvious choice for MPSs is:

$$\left\| |\psi\rangle - |\tilde{\psi}\rangle \right\|_2^2 \tag{3.25}$$

This needs to be minimized with respect to all the matrices in the MPS and their conjugates. This would lead to a huge system of linear equations. Even though that would be solvable it is easier to use a more efficient iterative method.

Iterative compression using one matrix

Instead of calculating the minimum by taking the derivative with respect to all of the elements, we set all of the matrices to be fixed except for some $\tilde{A}^{\tilde{\sigma}_i}_{m,n}$ and take the derivative of (3.25) with respect to $\tilde{A}^{\tilde{\sigma}_i\star}_{n,m}$. Setting the latter to zero, one can solve for $\tilde{A}^{\tilde{\sigma}_i}_{m,n}$. Doing this one by one for all the matrices in the MPS allows us to approximate the optimally compressed MPS. By repeating this process, one should be able to further minimize the distance to the optimally compressed MPS. Calculating the derivative:

$$\begin{aligned}
& \frac{\partial}{\partial \tilde{A}^{\sigma_i\star}_{n,m}} \left\| |\psi\rangle - |\tilde{\psi}\rangle \right\|_2^2 = 0 \\
\Rightarrow \quad & \frac{\partial}{\partial \tilde{A}^{\tilde{\sigma}_i\star}_{n,m}} \Big(\langle\psi|\psi\rangle - \langle\psi|\tilde{\psi}\rangle + \langle\tilde{\psi}|\tilde{\psi}\rangle - \langle\tilde{\psi}|\psi\rangle \Big) = 0 \\
\Rightarrow \quad & \frac{\partial}{\partial \tilde{A}^{\tilde{\sigma}_i\star}_{n,m}} \Big(\langle\tilde{\psi}|\tilde{\psi}\rangle - \langle\tilde{\psi}|\psi\rangle \Big) = 0
\end{aligned} \tag{3.26}$$

In the last step, the two terms $\langle\psi|\psi\rangle$ and $\langle\psi|\tilde{\psi}\rangle$ were dropped, since they do not contain any $\tilde{A}^{\tilde{\sigma}_i\star}_{n,m}$. Continuing from (3.26) and using the second step in (3.7):

$$\begin{aligned}
\Rightarrow \quad & \tilde{p}^\star \tilde{p} \sum_{\boldsymbol{\sigma}\setminus\sigma_i} \Big(\tilde{A}^{\sigma_N\dagger} \dots \tilde{A}^{\sigma_{i+1}\dagger} \Big)_{a_N,n} \Big(\tilde{A}^{\sigma_{i-1}\dagger} \dots \tilde{A}^{\sigma_1\dagger} \Big)_{m,1} \times \\
& \tilde{A}^{\sigma_1} \dots \tilde{A}^{\sigma_{i-1}} \tilde{A}^{\tilde{\sigma}_i} \tilde{A}^{\sigma_{i+1}} \dots \tilde{A}^{\sigma_N} \quad - && (3.27) \\
& \tilde{p}^\star p \sum_{\boldsymbol{\sigma}\setminus\sigma_i} \Big(\tilde{A}^{\sigma_N\dagger} \dots \tilde{A}^{\sigma_{i+1}\dagger} \Big)_{a_N,n} \Big(\tilde{A}^{\sigma_{i-1}\dagger} \dots \tilde{A}^{\sigma_1\dagger} \Big)_{m,1} \times && (3.28) \\
& A^{\sigma_1} \dots A^{\sigma_{i-1}} A^{\tilde{\sigma}_i} A^{\sigma_{i+1}} \dots A^{\sigma_N} = 0 && (3.29)
\end{aligned}$$

Even though it is possible to solve this equation to obtain $\tilde{A}^{\sigma_i}$, it looks quite hideous. This expression simplifies dramatically if $|\tilde{\psi}\rangle$ is right-canonized from $i+1$ to N and left-canonized from 1 to $i-1$. In addition, the iterative approach will be numerically more stable. To show this, we calculate the first term in

(3.26).

$$
\begin{aligned}
&\frac{\partial}{\partial \tilde{A}^{\tilde{\sigma}_i\star}_{n,m}} \langle\tilde{\psi}|\tilde{\psi}\rangle \\
&= \frac{\partial}{\partial \tilde{A}^{\tilde{\sigma}_i\star}_{n,m}} \tilde{p}^\star \tilde{p} \sum_{\boldsymbol{\sigma}} \sum_{\tilde{a}_1,\ldots,\tilde{a}_{N-1}} \tilde{A}^{\sigma_N\star}_{\tilde{a}_N,\tilde{a}_{N-1}} \ldots \tilde{A}^{\sigma_2\star}_{\tilde{a}_2,\tilde{a}_1} \tilde{A}^{\sigma_1\star}_{\tilde{a}_1,\tilde{a}_0} \quad \times \\
&\qquad \sum_{a_1,\ldots,a_{N-1}} \tilde{A}^{\sigma_1}_{1,a_1} \tilde{A}^{\sigma_2}_{a_1,a_2} \ldots \tilde{A}^{\sigma_N}_{a_{N-1},a_N} \\
&= |\tilde{p}|^2 \sum_{\boldsymbol{\sigma}\setminus\sigma_i} \sum_{\substack{\tilde{a}_1,\ldots,\tilde{a}_{i-2},\tilde{a}_{i+1},\ldots,\tilde{a}_{N-1} \\ a_1,\ldots,a_{N-1}}} \tilde{A}^{\sigma_N\star}_{\tilde{a}_N,\tilde{a}_{N-1}} \ldots \tilde{A}^{\sigma_{i+1}\star}_{\tilde{a}_{i+1},n} \quad \times \\
&\qquad \tilde{A}^{\sigma_{i-1}\star}_{m,\tilde{a}_{i-2}} \ldots \tilde{A}^{\sigma_1\star}_{\tilde{a}_1,\tilde{a}_0} \tilde{A}^{\sigma_1}_{a_0,a_1} \tilde{A}^{\sigma_2}_{a_1,a_2} \ldots \tilde{A}^{\sigma_{i-1}}_{a_{i-2},a_{i-1}} \quad \times \\
&\qquad \tilde{A}^{\tilde{\sigma}_i}_{a_{i-1},a_i} \tilde{A}^{\sigma_{i+1}}_{a_i,a_{i+1}} \ldots \tilde{A}^{\sigma_N}_{a_{N-1},a_N}
\end{aligned} \tag{3.30}
$$

In the last step $\tilde{A}^{\sigma_i\star}_{\tilde{a}_i,\tilde{a}_{i-1}}$ has disappeared due to the derivative. This equation can be simplified, because we know that $|\tilde{\psi}\rangle$ is left-canonized from 1 to $i-1$. Looking at the sum over σ_1 in the above equation, one can simplify due to $\tilde{A}^{\sigma_1}$ being left-normalized, using (3.13) and noting that $\tilde{a}_0 = a_0 = 1$.

$$
\sum_{\sigma_1} \tilde{A}^{\sigma_1\star}_{\tilde{a}_1,\tilde{a}_0} \tilde{A}^{\sigma_1}_{a_0,a_1} = \delta_{\tilde{a}_1,a_1} \tag{3.31}
$$

Looking at the next sum over σ_2 shows another simplification due to $\tilde{A}^{\sigma_2}$ being left-normalized.

$$
\sum_{\sigma_2} \sum_{a_1,\tilde{a}_1} \tilde{A}^{\sigma_2\star}_{\tilde{a}_2,\tilde{a}_1} \delta_{\tilde{a}_1,a_1} \tilde{A}^{\sigma_2}_{a_1,a_2} = \sum_{\sigma_2} \sum_{a_1} \tilde{A}^{\sigma_2\star}_{\tilde{a}_2,a_1} \tilde{A}^{\sigma_2}_{a_1,a_2} = \delta_{\tilde{a}_2,a_2} \tag{3.32}
$$

This can be continued until σ_{i-1} and one is left with simply $\delta_{\tilde{a}_{i-1},a_{i-1}}$. One can use the right canonization in a similar fashion. Since $\tilde{A}^{\sigma_N}$ is right-normalized, one can simplify the sum over σ_N using (3.5).

$$
\sum_{\sigma_N} \tilde{A}^{\sigma_N\star}_{\tilde{a}_N,\tilde{a}_{N-1}} \tilde{A}^{\sigma_N}_{a_{N-1},a_N} = \sum_{\sigma_N} \tilde{A}^{\sigma_N}_{a_{N-1},a_N} \tilde{A}^{\sigma_N\star}_{\tilde{a}_N,\tilde{a}_{N-1}} = \delta_{\tilde{a}_{N-1},a_{N-1}} \tag{3.33}
$$

Repeatedly applying this for all of the right-normalized matrices leaves one with only $\delta_{\tilde{a}_i,a_i}$. Therefore (3.30) reduces to:

$$
|\tilde{p}|^2 \sum_{a_i,a_{i-1}} \delta_{m,a_{i-1}} \tilde{A}^{\tilde{\sigma}_i}_{a_{i-1},a_i} \delta_{n,a_i} = |\tilde{p}|^2 \tilde{A}^{\tilde{\sigma}_i}_{m,n} \tag{3.34}
$$

Now the complicated expression of (3.30) has been simplified to one simple matrix, which is exactly the matrix we were looking for, so the solution of (3.26) is simply:

$$
\tilde{A}^{\tilde{\sigma}_i}_{m,n} = \frac{1}{|\tilde{p}|^2} \frac{\partial}{\partial \tilde{A}^{\tilde{\sigma}_i\star}_{n,m}} \langle\tilde{\psi}|\psi\rangle \tag{3.35}
$$

This can be written out explicitly:

$$
\begin{aligned}
&= \frac{1}{|\tilde{p}|^2}\frac{\partial}{\partial \tilde{A}^{\tilde{\sigma}_i\star}_{n,m}}\tilde{p}^\star p \sum_{\boldsymbol{\sigma}} \sum_{\tilde{a}_1,\ldots,\tilde{a}_{N-1}} \tilde{A}^{\sigma_N\star}_{\tilde{a}_N,\tilde{a}_{N-1}}\ldots\tilde{A}^{\sigma_2\star}_{\tilde{a}_2,\tilde{a}_1}\tilde{A}^{\sigma_1\star}_{\tilde{a}_1,\tilde{a}_0} \quad \times \\
&\qquad \sum_{a_1,\ldots,a_{N-1}} A^{\sigma_1}_{a_0,a_1}A^{\sigma_2}_{a_1,a_2}\ldots A^{\sigma_N}_{a_{N-1},a_N} \\
&= \frac{p}{\tilde{p}} \sum_{\boldsymbol{\sigma}\setminus\sigma_i} \sum_{\substack{\tilde{a}_1,\ldots,\tilde{a}_{i-2},\tilde{a}_{i+1},\ldots,\tilde{a}_{N-1} \\ a_1,\ldots,a_{N-1}}} \tilde{A}^{\sigma_N\star}_{\tilde{a}_N,\tilde{a}_{N-1}}\ldots\tilde{A}^{\sigma_{i+1}\star}_{\tilde{a}_{i+1},n} \quad \times \\
&\qquad \tilde{A}^{\sigma_{i-1}\star}_{m,\tilde{a}_{i-2}}\ldots\tilde{A}^{\sigma_1\star}_{\tilde{a}_1,\tilde{a}_0}A^{\sigma_1}_{a_0,a_1}A^{\sigma_2}_{a_1,a_2}\ldots A^{\sigma_{i-1}}_{a_{i-2},a_{i-1}}A^{\tilde{\sigma}_i}_{a_{i-1},a_i} \quad \times \\
&\qquad A^{\sigma_{i+1}}_{a_i,a_{i+1}}\ldots A^{\sigma_N}_{a_{N-1},a_N} \\
&= \frac{p}{\tilde{p}} \sum_{a_{i-1},a_i} \Bigg(\sum_{\sigma_{i-1}} \tilde{A}^{\sigma_{i-1}\dagger} \Bigg(\ldots \sum_{\sigma_2} \tilde{A}^{\sigma_2\dagger} \Bigg(\sum_{\sigma_1} \tilde{A}^{\sigma_1\dagger} \quad \times \\
&\qquad A^{\sigma_1}\Bigg) A^{\sigma_2}\ldots\Bigg) A^{\sigma_{i-1}}\Bigg)_{m,a_{i-1}} \quad \times \quad A^{\tilde{\sigma}_i}_{a_{i-1},a_i} \quad \times \\
&\qquad \sum_{\sigma_{i+1},\ldots,\sigma_N} (A^{\sigma_{i+1}}\ldots A^{\sigma_N})_{a_i,a_N} \left(\tilde{A}^{\sigma_N\dagger}\ldots\tilde{A}^{\sigma_{i+1}\dagger}\right)_{\tilde{a}_N,n} \\
&= \frac{p}{\tilde{p}} \sum_{a_{i-1},a_i} \Bigg(\sum_{\sigma_{i-1}} \tilde{A}^{\sigma_{i-1}\dagger} \Bigg(\ldots \sum_{\sigma_2} \tilde{A}^{\sigma_2\dagger} \Bigg(\sum_{\sigma_1} \tilde{A}^{\sigma_1\dagger} \quad \times \\
&\qquad A^{\sigma_1}\Bigg) A^{\sigma_2}\ldots\Bigg) A^{\sigma_{i-1}}\Bigg)_{m,a_{i-1}} \quad \times \quad A^{\tilde{\sigma}_i}_{a_{i-1},a_i} \quad \times \\
&\qquad \left(\sum_{\sigma_{i+1}} A^{\sigma_{i+1}} \left(\ldots \sum_{\sigma_N} A^{\sigma_N}\tilde{A}^{\sigma_N\dagger}\ldots \right) \tilde{A}^{\sigma_{i+1}\dagger} \right)_{a_i,n}
\end{aligned}
\tag{3.36}
$$

In the last step, all the summations over the spins have been grouped in a similar fashion as for the overlap in section 3.2.

Iterative compression using two matrices

One can also perform iterative compression with two matrices simultaneously, which has the advantage of being a little more crude and therefore has the potential to get out of possible local minima. It works basically the same as with one matrix, but instead of differentiating with respect to the composite matrix $\left(\tilde{A}^{\sigma_{k+1}\dagger}\tilde{A}^{\sigma_k\dagger}\right)_{n,m}$. This time we still need $\tilde{A}^{\sigma_1}$ to $\tilde{A}^{\sigma_{i-1}}$ to be left-canonized, but only $\tilde{A}^{\sigma_{i+2}}$ to $\tilde{A}^{\sigma_N}$ need to be right-canonized. This then leaves us with an expression similar to (3.35).

$$
\left(\tilde{A}^{\tilde{\sigma}_i}\tilde{A}^{\tilde{\sigma}_{i+1}}\right)_{m,n} = \frac{1}{|\tilde{p}|^2}\frac{\partial}{\partial\left(\tilde{A}^{\tilde{\sigma}_{i+1}\dagger}\tilde{A}^{\tilde{\sigma}_i\dagger}\right)_{n,m}}\langle\tilde{\psi}|\psi\rangle \tag{3.37}
$$

This can be worked out in a similar fashion as (3.36), but now leaving two matrices in the middle.

$$
\begin{aligned}
=&\frac{p}{\tilde{p}} \sum_{a_{i-1},a_i,a_{i+1}} \left(\sum_{\sigma_{i-1}} \tilde{A}^{\sigma_{i-1}\dagger} \left(\dots \sum_{\sigma_2} \tilde{A}^{\sigma_2\dagger} \left(\sum_{\sigma_1} \tilde{A}^{\sigma_1\dagger} \quad \times \right.\right.\right. \\
&\left.\left.\left. A^{\sigma_1} \right) A^{\sigma_2} \dots \right) A^{\sigma_{i-1}} \right)_{m,a_{i-1}} \times \quad A^{\tilde{\sigma}_i}_{a_{i-1},a_i} A^{\tilde{\sigma}_{i+1}}_{a_i,a_{i+1}} \quad \times \\
&\left(\sum_{\sigma_{i+2}} A^{\sigma_{i+2}} \left(\dots \sum_{\sigma_N} A^{\sigma_N} \tilde{A}^{\sigma_N\dagger} \dots \right) \tilde{A}^{\sigma_{i+1}\dagger} \right)_{a_i,n}
\end{aligned} \tag{3.38}
$$

Since this will give us the product of two matrices, we still need to unravel it to give us the two separate matrices. We can obtain those by reshaping, performing an SVD, multiplying the square root of the singular values into both matrices and finally reshaping back.

$$
\begin{aligned}
\left(\tilde{A}^{\tilde{\sigma}_i} \tilde{A}^{\tilde{\sigma}_{i+1}} \right)_{m,n} &= \left(\tilde{A}^{[i]} \tilde{A}^{[i+1]} \right)_{(\tilde{\sigma}_i,m),(\tilde{\sigma}_{i+1},n)} \\
&= \left(USV^\dagger \right)_{(\tilde{\sigma}_i,m),(\tilde{\sigma}_{i+1},n)} \\
&= \sum_\ell \left(U\sqrt{S} \right)_{(\tilde{\sigma}_i,m),\ell} \left(\sqrt{S}V^\dagger \right)_{\ell,(\tilde{\sigma}_{i+1},n)}
\end{aligned} \tag{3.39}
$$

This way, we obtain the new A-matrices, $\tilde{A}^{\tilde{\sigma}_i}_{m,\ell} = \left(U\sqrt{S} \right)_{(\tilde{\sigma}_i,m),\ell}$ and $\tilde{A}^{\tilde{\sigma}_{i+1}}_{\ell,n} = \left(\sqrt{S}V^\dagger \right)_{\ell,(\tilde{\sigma}_{i+1},n)}$. One could also choose not to factor the matrix S in $\sqrt{S}\sqrt{S}$, but instead to multiply the entire matrix S in either $\tilde{A}^{\tilde{\sigma}_i}_{m,\ell}$ or $\tilde{A}^{\tilde{\sigma}_{i+1}}_{\ell,n}$. The advantage of that is that the other matrix is immediately right- or left-canonized respectively.

3.7.3 Iterative compression of an MPO

The formal way of finding the best compression is to minimize the distance between the original operator, $\hat{O}$, and a compressed operator, $\hat{\tilde{O}}$. Following the example of the states, we now seek to minimize the distance between operators. Of course, we need some way of specifying what the distance is between operators. One way this can be done is by rewriting the operator as a 'state' (compare to (3.6)), since this will allow us to follow the same procedure as for MPSs in section 3.7.2.

$$
|O\rangle = \sum_{\boldsymbol{\sigma},\boldsymbol{\sigma}'} \sum_{b_1,\dots} B^{\sigma_1,\sigma_1'}_{b_0,b_1} B^{\sigma_2,\sigma_2'}_{b_1,b_2} B^{\sigma_3,\sigma_3'}_{b_2,b_3} \dots |\boldsymbol{\sigma}\rangle \otimes |\boldsymbol{\sigma'}\rangle \tag{3.40}
$$

After writing the operator in this form, we can calculate the distance between the 'states', which is effectively just the sum of the squares of the difference of

every matrix element.

$$\left\| |O\rangle - |\tilde{O}\rangle \right\|_2^2 \tag{3.41}$$

$$= \Big(\langle O| - \langle \tilde{O}| \Big) \Big(|O\rangle - |\tilde{O}\rangle \Big) \tag{3.42}$$

$$= \langle O|O\rangle - \langle \tilde{O}|O\rangle - \langle O|\tilde{O}\rangle + \langle \tilde{O}|\tilde{O}\rangle \tag{3.43}$$

This norm is in effect the same as the Frobenius norm for matrices, since it is equal to the quadratic sum of all matrix elements of the operator $\hat{O} - \hat{\tilde{O}}$.

Iterative compression using one matrix

To find the minimum of (3.43) we can take the derivative of it with respect to every matrix element $\tilde{B}^{\sigma_i,\sigma_i'\star}_{b_i,b_{i-1}}$ and $\tilde{B}^{\sigma_i,\sigma_i'}_{b_{i-1},b_i}$ and set this to zero. As before, instead of doing this for every matrix element at the same time, we do this iteratively by starting with a proposed compressed state and then minimizing only with respect to one $\tilde{B}^{\sigma_i,\sigma_i'}$. Since we will only be taking the derivative of (3.43) with respect to $\tilde{B}^{\sigma_i,\sigma_i'\star}_{b_i,b_{i-1}}$, only the terms with $\langle \tilde{O}|$ are needed.

$$\begin{aligned}
&\frac{\partial}{\partial \tilde{B}^{\sigma_i,\sigma_i'\star}_{b_i,b_{i-1}}} \Big(\langle \tilde{O}|\tilde{O}\rangle - \langle \tilde{O}|O\rangle \Big) = 0 \\
\Rightarrow &\sum_{\boldsymbol{\sigma}\boldsymbol{\sigma}'} \Big(\tilde{B}^{\sigma_N\sigma_N'\dagger} \dots \tilde{B}^{\sigma_{i+1}\sigma_{i+1}'\dagger} \Big)_{b_N,b_i} \Big(\tilde{B}^{\sigma_{i-1}\sigma_{i-1}'\dagger} \dots \tilde{B}^{\sigma_1\sigma_1'\dagger} \Big)_{b_{i-1},b_0} \quad \times \\
&\qquad \tilde{B}^{\sigma_1\sigma_1'} \dots \tilde{B}^{\sigma_N\sigma_N'} \; - \sum_{\boldsymbol{\sigma}\boldsymbol{\sigma}'} \Big(\tilde{B}^{\sigma_N\sigma_N'\dagger} \dots \tilde{B}^{\sigma_{i+1}\sigma_{i+1}'\dagger} \Big)_{b_N,b_i} \quad \times \\
&\Big(\tilde{B}^{\sigma_{i-1}\sigma_{i-1}'\dagger} \dots \tilde{B}^{\sigma_1\sigma_1'\dagger} \Big)_{b_{i-1},b_0} B^{\sigma_1\sigma_1'} \dots B^{\sigma_N\sigma_N'} = 0
\end{aligned} \tag{3.44}$$

This expression again simplifies dramatically if $|\tilde{O}\rangle$ is right-canonized from $i+1$ to N and left-canonized from 1 to $i-1$. First note that:

$$\langle \tilde{O}| = \sum_{\boldsymbol{\sigma},\boldsymbol{\sigma}'} \sum_{\tilde{b}_1,\dots,\tilde{b}_{N-1}} \tilde{B}^{\sigma_N,\sigma_N'\star}_{\tilde{b}_N,\tilde{b}_{N-1}} \dots \tilde{B}^{\sigma_3,\sigma_3'\star}_{\tilde{b}_3,\tilde{b}_2} \tilde{B}^{\sigma_2,\sigma_2'\star}_{\tilde{b}_2,\tilde{b}_1} \tilde{B}^{\sigma_1,\sigma_1'\star}_{\tilde{b}_1,\tilde{b}_0} \langle \boldsymbol{\sigma}| \otimes \langle \boldsymbol{\sigma}'| \tag{3.45}$$

Then calculate the first term in (3.44).

$$\begin{aligned}
&\frac{\partial}{\partial \tilde{B}^{\sigma_i,\sigma_i'\star}_{\tilde{b}_i,\tilde{b}_{i-1}}} \langle \tilde{O}|\tilde{O}\rangle \\
&= \frac{\partial}{\partial \tilde{B}^{\sigma_i,\sigma_i'\star}_{\tilde{b}_i,\tilde{b}_{i-1}}} \sum_{\boldsymbol{\sigma},\boldsymbol{\sigma}'} \sum_{\tilde{b}_1,\dots,\tilde{b}_{N-1}} \tilde{B}^{\sigma_N,\sigma_N'\star}_{\tilde{b}_N,\tilde{b}_{N-1}} \dots \tilde{B}^{\sigma_2,\sigma_2'\star}_{\tilde{b}_2,\tilde{b}_1} \tilde{B}^{\sigma_1,\sigma_1'\star}_{\tilde{b}_1,\tilde{b}_0} \quad \times \\
&\qquad \sum_{b_1,\dots,b_{N-1}} \tilde{B}^{\sigma_1,\sigma_1'}_{b_0,b_1} \tilde{B}^{\sigma_2,\sigma_2'}_{b_1,b_2} \dots \tilde{B}^{\sigma_N,\sigma_N'}_{b_{N-1},b_N} \\
&= \sum_{\substack{\sigma_1,\dots,\sigma_{i-1},\sigma_{i+1},\dots,\sigma_N \\ \sigma_1',\dots,\sigma_{i-1}',\sigma_{i+1}',\dots,\sigma_N'}} \sum_{\tilde{b}_1,\dots,\tilde{b}_{i-2},\tilde{b}_{i+1},\dots,\tilde{b}_{N-1}} \sum_{b_1,\dots,b_{N-1}} \tilde{B}^{\sigma_N,\sigma_N'\star}_{\tilde{b}_N,\tilde{b}_{N-1}} \dots \times \\
&\qquad \dots \tilde{B}^{\sigma_{i+1},\sigma_{i+1}'\star}_{\tilde{b}_{i+1},\tilde{b}_i} \tilde{B}^{\sigma_{i-1},\sigma_{i-1}'\star}_{\tilde{b}_{i-1},\tilde{b}_{i-2}} \dots \tilde{B}^{\sigma_1,\sigma_1'\star}_{\tilde{b}_1,\tilde{b}_0} \tilde{B}^{\sigma_1,\sigma_1'}_{b_0,b_1} \quad \times \\
&\qquad \tilde{B}^{\sigma_2,\sigma_2'}_{b_1,b_2} \dots \tilde{B}^{\sigma_N,\sigma_N'}_{b_{N-1},b_N}
\end{aligned} \tag{3.46}$$

Using the fact that the matrices are left/right-normalized, we can use (3.18), (3.19) to simplify this equation, just as we used the normalization to simplify (3.30). Remember that b_0, $\tilde{b}_0$, b_N and $\tilde{b}_N$ are all dummy indices equal to one. This allows us to use

$$\sum_{\sigma_1,\sigma_1'} \tilde{B}^{\sigma_1,\sigma_1'\star}_{\tilde{b}_1,\tilde{b}_0} \tilde{B}^{\sigma_1,\sigma_1'}_{b_0,b_1} = \delta_{\tilde{b}_1,b_1} \tag{3.47}$$

and

$$\sum_{\sigma_N,\sigma_N'} \tilde{B}^{\sigma_N,\sigma_N'\star}_{\tilde{b}_N,\tilde{b}_{N-1}} \tilde{B}^{\sigma_N,\sigma_N'}_{b_{N-1},b_N} = \sum_{\sigma_N,\sigma_N'} \tilde{B}^{\sigma_N,\sigma_N'}_{b_{N-1},b_N} \tilde{B}^{\sigma_N,\sigma_N'\star}_{\tilde{b}_N,\tilde{b}_{N-1}} = \delta_{\tilde{b}_{N-1},b_{N-1}} \quad . \tag{3.48}$$

By using this canonization, (3.46) can be reduced to

$$\begin{aligned} \frac{\partial}{\partial \tilde{B}^{\sigma_i,\sigma_i'\star}_{\tilde{b}_i,\tilde{b}_{i-1}}} \langle \tilde{O}|\tilde{O}\rangle &= \sum_{b_{i-1},b_i} \delta_{\tilde{b}_{i-1},b_{i-1}} \delta_{\tilde{b}_i,b_i} \tilde{B}^{\sigma_i,\sigma_i'}_{b_{i-1},b_i} \\ &= \tilde{B}^{\sigma_i,\sigma_i'}_{\tilde{b}_{i-1},\tilde{b}_i} \end{aligned} \tag{3.49}$$

Putting this result back in (3.44) results in:

$$\tilde{B}^{\sigma_i,\sigma_i'}_{\tilde{b}_{i-1},\tilde{b}_i} = \frac{\partial}{\partial \tilde{B}^{\sigma_i,\sigma_i'\star}_{\tilde{b}_i,\tilde{b}_{i-1}}} \langle \tilde{O}|O\rangle \tag{3.50}$$

$$= \sum_{\substack{\sigma_1,\ldots,\sigma_{i-1},\sigma_{i+1},\ldots,\sigma_N \\ \sigma_1',\ldots,\sigma_{i-1}',\sigma_{i+1}',\ldots,\sigma_N'}} \left(\tilde{B}^{\sigma_N\sigma_N'\dagger} \ldots \tilde{B}^{\sigma_{i+1}\sigma_{i+1}'\dagger}\right)_{\tilde{b}_N,\tilde{b}_i} \times \left(\tilde{B}^{\sigma_{i-1}\sigma_{i-1}'\dagger} \ldots \tilde{B}^{\sigma_1\sigma_1'\dagger}\right)_{\tilde{b}_{i-1},\tilde{b}_0} B^{\sigma_1\sigma_1'} \ldots B^{\sigma_N\sigma_N'} \tag{3.51}$$

Note that the expression $B^{\sigma_1\sigma_1'} \ldots B^{\sigma_N\sigma_N'}$ keeps the $B^{\sigma_i\sigma_i'}$ explicitly. That is, it's not being summed over. It's more efficient to split out the summations, grouping the sums over a specific σ_j as was done with calculating the overlap

between two MPSs.

$$
\begin{aligned}
\tilde{B}^{\sigma_i,\sigma_i'}_{\tilde{b}_{i-1},\tilde{b}_i} = & \\
& \sum_{b_{i-1},b_i} \left(\sum_{\sigma_{i-1},\sigma_{i-1}'} \tilde{B}^{\sigma_{i-1}\sigma_{i-1}'\dagger} \left(\dots \sum_{\sigma_2,\sigma_2'} \tilde{B}^{\sigma_2\sigma_2'\dagger} \left(\sum_{\sigma_1,\sigma_1'} \tilde{B}^{\sigma_1\sigma_1'\dagger} \quad \times \right. \right. \right. \\
& \left. \left. \left. B^{\sigma_1\sigma_1'} \right) B^{\sigma_2\sigma_2'} \dots \right) B^{\sigma_{i-1}\sigma_{i-1}'} \right)_{\tilde{b}_{i-1},b_{i-1}} \times \quad B^{\sigma_i\sigma_i'}_{b_{i-1},b_i} \quad \times \\
& \sum_{\substack{\sigma_{i+1},\dots,\sigma_N \\ \sigma_{i+1}',\dots,\sigma_N'}} \left(B^{\sigma_{i+1}\sigma_{i+1}'} \dots B^{\sigma_N\sigma_N'} \right)_{b_i,b_N} \quad \times \\
& \left(\tilde{B}^{\sigma_N\sigma_N'\dagger} \dots \tilde{B}^{\sigma_{i+1}\sigma_{i+1}'\dagger} \right)_{\tilde{b}_N,\tilde{b}_i} \\
= & \sum_{b_{i-1},b_i} \left(\sum_{\sigma_{i-1},\sigma_{i-1}'} \tilde{B}^{\sigma_{i-1}\sigma_{i-1}'\dagger} \left(\dots \sum_{\sigma_2,\sigma_2'} \tilde{B}^{\sigma_2\sigma_2'\dagger} \left(\sum_{\sigma_1,\sigma_1'} \tilde{B}^{\sigma_1\sigma_1'\dagger} \right. \right. \right. \\
& \left. \left. \left. B^{\sigma_1\sigma_1'} \right) B^{\sigma_2\sigma_2'} \dots \right) B^{\sigma_{i-1}\sigma_{i-1}'} \right)_{\tilde{b}_{i-1},b_{i-1}} \times \quad B^{\sigma_i\sigma_i'}_{b_{i-1},b_i} \quad \times \\
& \left(\sum_{\sigma_{i+1},\sigma_{i+1}'} B^{\sigma_{i+1}\sigma_{i+1}'} \left(\dots \sum_{\sigma_N,\sigma_N'} B^{\sigma_N\sigma_N'} \quad \times \right. \right. \\
& \left. \left. \tilde{B}^{\sigma_N\sigma_N'\dagger} \dots \right) \tilde{B}^{\sigma_{i+1}\sigma_{i+1}'\dagger} \right)_{b_i,\tilde{b}_{i+1}}
\end{aligned} \tag{3.52}
$$

Iterative compression using two matrices

For an MPO it again works the same as with one matrix, but instead, we differentiate with respect to the composite matrix $\left(\tilde{B}^{\sigma_{k+1}\sigma_{k+1}'\dagger}\tilde{B}^{\sigma_k\sigma_k'\dagger}\right)_{n,m}$. This time we still need $\tilde{B}^{\sigma_1\sigma_1'}$ to $\tilde{B}^{\sigma_{i-1}\sigma_{i-1}'}$ to be left-canonized, but only $\tilde{B}^{\sigma_{i+2}\sigma_{i+2}'}$ to $\tilde{B}^{\sigma_N\sigma_N'}$ needs to be right-canonized. This then leaves us with an expression similar to (3.35).

$$
\left(\tilde{B}^{\tilde{\sigma}_i\tilde{\sigma}_i'}\tilde{B}^{\tilde{\sigma}_{i+1}\tilde{\sigma}_{i+1}'}\right)_{m,n} = \frac{1}{|\tilde{q}|^2} \frac{\partial}{\partial \left(\tilde{B}^{\tilde{\sigma}_{i+1}\tilde{\sigma}_{i+1}'\dagger}\tilde{B}^{\tilde{\sigma}_i\tilde{\sigma}_i'\dagger}\right)_{n,m}} \langle \tilde{O}|O\rangle \tag{3.53}
$$

This can be worked out in a similar fashion to (3.36), but now leaving two matrices in the middle.

$$
\begin{aligned}
= \frac{q}{\tilde{q}} & \sum_{b_{i-1},b_i,b_{i+1}} \left(\sum_{\sigma_{i-1}\sigma_{i-1}'} \tilde{B}^{\sigma_{i-1}\sigma_{i-1}'\dagger} \left(\dots \sum_{\sigma_2\sigma_2'} \tilde{B}^{\sigma_2\sigma_2'\dagger} \left(\sum_{\sigma_1\sigma_1'} \tilde{B}^{\sigma_1\sigma_1'\dagger} \quad \times \right. \right. \right. \\
& \left. \left. \left. B^{\sigma_1\sigma_1'} \right) B^{\sigma_2\sigma_2'} \dots \right) B^{\sigma_{i-1}\sigma_{i-1}'} \right)_{m,b_{i-1}} \times \quad B^{\tilde{\sigma}_i\tilde{\sigma}_i'}_{b_{i-1},b_i} B^{\tilde{\sigma}_{i+1}\tilde{\sigma}_{i+1}'}_{b_i,b_{i+1}} \quad \times \\
& \left(\sum_{\sigma_{i+2}\sigma_{i+2}'} B^{\sigma_{i+2}\sigma_{i+2}'} \left(\dots \sum_{\sigma_N\sigma_N'} B^{\sigma_N\sigma_N'} \tilde{B}^{\sigma_N\sigma_N'\dagger} \dots \right) \tilde{B}^{\sigma_{i+1}\sigma_{i+1}'\dagger} \right)_{b_i,n}
\end{aligned} \tag{3.54}
$$

Since this will give us the product of two matrices, we still need to unravel it to give us the two seperate matrices. We can obtain those by reshaping, performing an SVD, multiplying the square root of the singular values into both matrices and finally reshaping back.

$$\begin{aligned}\left(\tilde{B}^{\tilde{\sigma}_i\tilde{\sigma}'_i}\tilde{B}^{\tilde{\sigma}_{i+1}\tilde{\sigma}'_{i+1}}\right)_{m,n} &= \left(\tilde{B}^{[i]}\tilde{B}^{[i+1]}\right)_{(\tilde{\sigma}_i,\tilde{\sigma}'_i,m),(\tilde{\sigma}_{i+1},\tilde{\sigma}'_{i+1},n)} \\ &= \left(USV^\dagger\right)_{(\tilde{\sigma}_i,\tilde{\sigma}'_i,m),(\tilde{\sigma}_{i+1},\tilde{\sigma}'_{i+1},n)} \\ &= \sum_\ell \left(U\sqrt{S}\right)_{(\tilde{\sigma}_i,\tilde{\sigma}'_i,m),\ell}\left(\sqrt{S}V^\dagger\right)_{\ell,(\tilde{\sigma}_{i+1},\tilde{\sigma}'_{i+1},n)} \end{aligned} \tag{3.55}$$

This way, we obtain the new B-matrices, $\tilde{B}^{\tilde{\sigma}_i,\tilde{\sigma}'_i}_{m,\ell} = \left(U\sqrt{S}\right)_{(\tilde{\sigma}_i,\tilde{\sigma}'_i,m),\ell}$ and $\tilde{B}^{\tilde{\sigma}_{i+1}\tilde{\sigma}'_{i+1}}_{\ell,n} = \left(\sqrt{S}V^\dagger\right)_{\ell,(\tilde{\sigma}_{i+1},\tilde{\sigma}'_{i+1},n)}$.

3.8 Other distance measures

One could possibly use other more 'proper' distance measures for the matrix, such as trace distance or fidelity. However, these have the disadvantage that they do not have such simple expressions as the currently used distance measure. As an example, let us look at trace distance.

$$\begin{aligned}\mathrm{Tr}|\hat{\tilde{O}} - \hat{O}| &= \mathrm{Tr}\sqrt{(\hat{\tilde{O}} - \hat{O})^\dagger(\hat{\tilde{O}} - \hat{O})} \\ &= \mathrm{Tr}\sqrt{\hat{\tilde{O}}^\dagger\hat{\tilde{O}} - \hat{\tilde{O}}^\dagger\hat{O} - \hat{O}^\dagger\hat{\tilde{O}} + \hat{O}^\dagger\hat{O}} \\ &= \sum_{\boldsymbol{\sigma}} \left\langle \boldsymbol{\sigma} \left| \sqrt{\hat{\tilde{O}}^\dagger\hat{\tilde{O}} - \hat{\tilde{O}}^\dagger\hat{O} - \hat{O}^\dagger\hat{\tilde{O}} + \hat{O}^\dagger\hat{O}} \right| \boldsymbol{\sigma} \right\rangle \end{aligned} \tag{3.56}$$

There is no easy way to take the derivative of this with respect to some $B^{\sigma_k,\sigma'_k\star}_{n,m}$ as in (3.44), because of the square root. Other distance measures suffer from the same problem.

3.9 Matrix element

Calculating a matrix element can be done by using the overlap calculation in conjunction with the section about applying an MPO.

$$\langle\tilde{\psi}|\hat{O}|\psi\rangle = \langle\tilde{\psi}|\left(\hat{O}|\psi\rangle\right) \tag{3.57}$$

While this works well when either MPOs or MPSs has a small bond dimension, it breaks down when the bond dimension for both the MPO and the MPS are large. This is due to the fact that the bond dimension of $\hat{O}|\psi\rangle$ grows as the product of the bond dimension of $\hat{O}$ and $|\psi\rangle$. The amount of memory required to store this, is then extremely large and impractical. A better solution is to

calculate $\langle\tilde{\psi}|\hat{O}|\psi\rangle$ directly:

$$\begin{aligned}
\langle\tilde{\psi}|\hat{O}|\psi\rangle &= \tilde{p}\,q\,p\sum_{\boldsymbol{\sigma}}\left(\tilde{A}^{\sigma_N}\right)^\dagger\ldots\left(\tilde{A}^{\sigma_1}\right)^\dagger\langle\boldsymbol{\sigma}| \quad\times\\
&\qquad\sum_{\boldsymbol{\sigma''\sigma'''}}B^{\sigma''_1\sigma'''_1}\ldots B^{\sigma''_N\sigma'''_N}|\boldsymbol{\sigma''}\rangle\langle\boldsymbol{\sigma'''}|\sum_{\boldsymbol{\sigma'}}A^{\sigma'_1}\ldots A^{\sigma'_N}|\boldsymbol{\sigma'}\rangle\\
&= \tilde{p}\,q\,p\sum_{\boldsymbol{\sigma,\sigma'}}\left(\tilde{A}^{\sigma_N}\right)^\dagger\ldots\left(\tilde{A}^{\sigma_1}\right)^\dagger B^{\sigma_1\sigma'_1}\ldots B^{\sigma_N\sigma'_N}A^{\sigma'_1}\ldots A^{\sigma'_N}\\
&= \tilde{p}\,q\,p\sum_{\boldsymbol{\sigma,\sigma'}}\sum_{\tilde{a}_1\ldots}\left(\tilde{A}^{\sigma_N\dagger}\right)_{\tilde{a}_N,\tilde{a}_{N-1}}\ldots\left(\tilde{A}^{\sigma_1\dagger}\right)_{\tilde{a}_1,\tilde{a}_0} \quad\times\\
&\qquad\sum_{b_1\ldots}B^{\sigma_1\sigma'_1}_{b_0,b_1}\ldots B^{\sigma_N\sigma'_N}_{b_{N-1},b_N}\sum_{a_1\ldots}A^{\sigma'_1}_{a_0,a_1}\ldots A^{\sigma'_N}_{a_{N-1},a_N}\\
&= \tilde{p}\,q\,p\sum_{\substack{\boldsymbol{\sigma}\setminus\boldsymbol{\sigma_1}\\ \boldsymbol{\sigma'}\setminus\boldsymbol{\sigma'_1}}}\sum_{\tilde{a}_1\ldots}\left(\tilde{A}^{\sigma_N\dagger}\right)_{\tilde{a}_N,\tilde{a}_{N-1}}\ldots\left(\tilde{A}^{\sigma_2\dagger}\right)_{\tilde{a}_2,\tilde{a}_1} \quad\times\\
&\qquad\sum_{b_1\ldots}B^{\sigma_2\sigma'_2}_{b_1,b_2}\ldots B^{\sigma_N\sigma'_N}_{b_{N-1},b_N}\sum_{a_1\ldots}A^{\sigma'_2}_{a_1,a_2}\ldots A^{\sigma'_N}_{a_{N-1},a_N} \quad\times\\
&\qquad\sum_{\sigma_1,\sigma'_1}\left(\tilde{A}^{\sigma_1\dagger}\right)_{\tilde{a}_1,\tilde{a}_0}B^{\sigma_1\sigma'_1}_{b_0,b_1}A^{\sigma'_1}_{a_0,a_1}
\end{aligned}\tag{3.58}$$

In the last step everything related to site 1 has been separated out and can be calculated and replaced by the tensor $M^{(1)}_{\tilde{a}_1,b_1,a_1}$:

$$\begin{aligned}
&= \tilde{p}\,q\,p\sum_{\substack{\boldsymbol{\sigma}\setminus\boldsymbol{\sigma_1}\\ \boldsymbol{\sigma'}\setminus\boldsymbol{\sigma'_1}}}\sum_{\tilde{a}_1\ldots}\left(\tilde{A}^{\sigma_N\dagger}\right)_{\tilde{a}_N,\tilde{a}_{N-1}}\ldots\left(\tilde{A}^{\sigma_2\dagger}\right)_{\tilde{a}_2,\tilde{a}_1} \quad\times\\
&\qquad\sum_{b_1\ldots}B^{\sigma_2\sigma'_2}_{b_1,b_2}\ldots B^{\sigma_N\sigma'_N}_{b_{N-1},b_N}\sum_{a_1\ldots}A^{\sigma'_2}_{a_1,a_2}\ldots A^{\sigma'_N}_{a_{N-1},a_N}M^{(1)}_{\tilde{a}_1,b_1,a_1}\\
&= \tilde{p}\,q\,p\sum_{\substack{\boldsymbol{\sigma}\setminus\boldsymbol{\sigma_1,\sigma_2}\\ \boldsymbol{\sigma'}\setminus\boldsymbol{\sigma'_1,\sigma'_2}}}\sum_{\tilde{a}_2\ldots}\left(\tilde{A}^{\sigma_N\dagger}\right)_{\tilde{a}_N,\tilde{a}_{N-1}}\ldots\left(\tilde{A}^{\sigma_3\dagger}\right)_{\tilde{a}_3,\tilde{a}_2} \quad\times\\
&\qquad\sum_{b_2\ldots}B^{\sigma_3\sigma'_3}_{b_2,b_3}\ldots B^{\sigma_N\sigma'_N}_{b_{N-1},b_N}\sum_{a_2\ldots}A^{\sigma'_3}_{a_2,a_3}\ldots A^{\sigma'_N}_{a_{N-1},a_N} \quad\times\\
&\qquad\sum_{\substack{\sigma_2,\sigma'_2\\ \tilde{a}_1,b_1,a_1}}M^{(1)}_{\tilde{a}_1,b_1,a_1}\left(\tilde{A}^{\sigma_2\dagger}\right)_{\tilde{a}_2,\tilde{a}_1}B^{\sigma_2\sigma'_2}_{b_1,b_2}A^{\sigma'_2}_{a_1,a_2}\\
&= \tilde{p}\,q\,p\sum_{\substack{\boldsymbol{\sigma}\setminus\boldsymbol{\sigma_1,\sigma_2}\\ \boldsymbol{\sigma'}\setminus\boldsymbol{\sigma'_1,\sigma'_2}}}\sum_{\tilde{a}_2\ldots}\left(\tilde{A}^{\sigma_N\dagger}\right)_{\tilde{a}_N,\tilde{a}_{N-1}}\ldots\left(\tilde{A}^{\sigma_3\dagger}\right)_{\tilde{a}_3,\tilde{a}_2} \quad\times\\
&\qquad\sum_{b_2\ldots}B^{\sigma_3\sigma'_3}_{b_2,b_3}\ldots B^{\sigma_N\sigma'_N}_{b_{N-1},b_N}\sum_{a_2\ldots}A^{\sigma'_3}_{a_2,a_3}\ldots A^{\sigma'_N}_{a_{N-1},a_N}M^{(2)}_{\tilde{a}_2,b_2,a_2}
\end{aligned}\tag{3.59}$$

Doing this for all the sites, one eventually ends up with a scalar:

$$\langle\tilde{\psi}|\hat{O}|\psi\rangle = \tilde{p}\,q\,p\,M^{(N)}_{\tilde{a}_N,b_N,a_N}\tag{3.60}$$

with

$$M^{(i)}_{\tilde{a}_i,b_i,a_i} = \sum_{\substack{\sigma_i,\sigma_i' \\ \tilde{a}_{i-1},b_{i-1},a_{i-1}}} M^{(i-1)}_{\tilde{a}_{i-1},b_{i-1},a_{i-1}} \left(\tilde{A}^{\sigma_i\dagger}\right)_{\tilde{a}_i,\tilde{a}_{i-1}} B^{\sigma_i\sigma_i'}_{b_{i-1},b_i} A^{\sigma_i'}_{a_{i-1},a_i} \quad . \tag{3.61}$$

Since *Lapack++* does not work well with tensors, we need matrices:

$$M^{(i)}_{\tilde{a}_i,b_i,a_i} = \sum_{\substack{\sigma_i,\sigma_i' \\ \tilde{a}_{i-1},b_{i-1},a_{i-1}}} \left(\tilde{A}^{\sigma_i\dagger}\right)_{\tilde{a}_i,\tilde{a}_{i-1}} M^{(i-1)}_{\tilde{a}_{i-1},(b_{i-1},a_{i-1})} B^{\sigma_i\sigma_i'}_{b_{i-1},b_i} A^{\sigma_i'}_{a_{i-1},a_i} \tag{3.62}$$

$$= \sum_{\substack{\sigma_i,\sigma_i' \\ \tilde{a}_{i-1},b_{i-1}}} P^{\sigma_i}_{\tilde{a}_i,(b_{i-1},a_{i-1})} B^{\sigma_i\sigma_i'}_{b_{i-1},b_i} A^{\sigma_i'}_{a_{i-1},a_i} \tag{3.63}$$

$$= \sum_{\substack{\sigma_i' \\ \tilde{a}_{i-1}}} \left(\sum_{\sigma_i,b_{i-1}} P^{\sigma_i}_{(\tilde{a}_i,a_{i-1}),b_{i-1}} B^{\sigma_i\sigma_i'}_{b_{i-1},b_i} \right) A^{\sigma_i'}_{a_{i-1},a_i} \tag{3.64}$$

$$= \sum_{\substack{\sigma_i' \\ \tilde{a}_{i-1}}} Q^{\sigma_i'}_{(\tilde{a}_i,a_{i-1}),b_i} A^{\sigma_i'}_{a_{i-1},a_i} \tag{3.65}$$

$$= \sum_{\substack{\sigma_i' \\ \tilde{a}_{i-1}}} Q^{\sigma_i'}_{(\tilde{a}_i,b_i),a_{i-1}} A^{\sigma_i'}_{a_{i-1},a_i} \tag{3.66}$$

$$= M^{(i)}_{(\tilde{a}_i,b_i),a_i} \tag{3.67}$$

Since every $M^{(i)}$ is calculated per site, far less memory is needed compared to calculating (3.57) directly, because the complete MPS $\hat{O}|\psi\rangle$, which contains $N \times d^2_{\hat{O}} \times d^2_{|\psi\rangle} \times \dim(\sigma)$ elements does not need to be stored. Instead we only need the intermediate matrix $M^{(i)}$, which only has a size of $d_{\hat{O}} \times d^2_{|\psi\rangle}$. Here $d_{\hat{O}}$ stands for the bond dimension of the MPO $\hat{O}$ and $d_{|\psi\rangle}$ stands for the bond dimension of $|\psi\rangle$.

Chapter 4

Physical operators

The MPOs as defined in chapter 3 need to be made explicit for us to be able to perform simulations. To do so, the Hamiltonian of the system under investigation is needed, since this determines the time evolution, amongst other things. In the case of a spin-$\frac{1}{2}$ XX chain, the Hamiltonian is well known:

$$\hat{H} = -\frac{J}{2}\sum_i \left(\hat{S}_i^+\hat{S}_{i+1}^- + \hat{S}_{i+1}^+\hat{S}_i^-\right) \tag{4.1}$$

Another Hamiltonian we will make use of is the spin-$\frac{1}{2}$ XXZ chain.

$$\hat{H} = -\frac{J}{2}\sum_i \left(\hat{S}_i^+\hat{S}_{i+1}^- + \hat{S}_{i+1}^+\hat{S}_i^-\right) + J_z\sum_i \hat{S}_i^z\hat{S}_{i+1}^z \tag{4.2}$$

And finally the one for the Bose-Hubbard model.

$$\hat{H} = -t\sum_i \left(\hat{b}_i^\dagger\hat{b}_{i+1} + \hat{b}_{i+1}^\dagger\hat{b}_i\right) - \frac{U}{2}\sum_i \hat{n}_i\left(\hat{n}_i - 1\right) - \mu\sum_i \hat{n}_i \tag{4.3}$$

4.1 Single site operator

Many of the operators that are going to be used are operators that work only on a single site, $\hat{O}_i$. These operators are easily written in the form of an MPO.

$$\hat{O}_i = 1 \otimes 1 \otimes \ldots 1 \otimes O_i \otimes 1 \ldots \otimes 1 \tag{4.4}$$

Where O_j (without the hat) stands for the local operator working in the Hilbert-space of a single site. All the $1^{\sigma_i,\sigma_i'}$ can be written as $\delta_{\sigma_i,\sigma_i'}$. and the dimension of every matrix is just 1×1.

$$p = 1 \tag{4.5}$$

$$B^{[n]} = \begin{bmatrix} O_j \end{bmatrix} \tag{4.6}$$

$$B^{[i]} = \begin{bmatrix} 1 \end{bmatrix} \qquad \text{for } i \neq j \tag{4.7}$$

The local operators we will use for the XX(Z)-chains are:

- $\hat{S}^z = \frac{1}{2}\sigma_z$, with σ_z being the Pauli matrix. This means $B_{1,1}^{\sigma_j,\sigma_j'} = \frac{1}{2}\sigma_z^{\sigma_j,\sigma_j'}$.

- $\hat{S}^+$ raises the spin, which is written as an MPO matrix as $B_{1,1}^{\sigma_j,\sigma_j'} = \delta_{\sigma_j,\uparrow}\delta_{\sigma_j',\downarrow}$.
- $\hat{S}^-$ lowers the spin, which is written as an MPO matrix as $B_{1,1}^{\sigma_j,\sigma_j'} = \delta_{\sigma_j,\downarrow}\delta_{\sigma_j',\uparrow}$.

For the Bose-Hubbard model we are only interested in the expectation value of the number operator:

- $\hat{n}_j$ counts the number of bosons at site j. $B_{1,1}^{\sigma_j,\sigma_j'} = \sigma_j\delta_{\sigma_j,\sigma_j'}$.

Note that in the case of the Bose-Hubbard model, σ_i does not stand for the spin at site i, but rather for the number of bosons at site i.

4.2 Total magnetization MPO

We are going to need (4.4) with S^z, but now added up for every site:

$$\hat{M} = hS_1^z \otimes 1 \otimes 1 \otimes \ldots \otimes 1 + 1 \otimes hS_2^z \otimes 1 \otimes 1 \otimes \ldots \otimes 1 + \ldots + 1 \otimes \ldots \otimes 1 \otimes hS_N^z \quad (4.8)$$

The following B-matrices will accomplish just that:

$$q = h \quad (4.9)$$

$$B^{[1]} = \begin{bmatrix} S^z & 1 \end{bmatrix} \quad (4.10)$$

$$B^{[i]} = \begin{bmatrix} 1 & 0 \\ S^z & 1 \end{bmatrix} \quad (4.11)$$

$$B^{[N]} = \begin{bmatrix} 1 \\ S^z \end{bmatrix} \quad (4.12)$$

Indeed, we see that:

$$B^{[1]}B^{[2]}\ldots B^{[N]} = h\sum_i S_i^z = M \quad (4.13)$$

4.3 Energy

The energy for the XX chain can be obtained by applying the Hamiltonian, which has to be written as an MPO. Converting the Hamiltonian from (4.1) into an MPO is similar to the procedure for the total magnetization MPO. The $-\frac{J}{2}$ can be put in the q again. The operators themselves are more difficult this time though, since we now have a sum of operators that work on two sites. Call one such two-site operator $\hat{h}_i$.

$$\hat{h}_i = \hat{S}_i^+\hat{S}_{i+1}^- + \hat{S}_{i+1}^+\hat{S}_i^- \quad (4.14)$$

This can still easily be written in MPO form:

$$q = 1 \tag{4.15}$$

$$B^{[1]} = \begin{bmatrix} 1 \end{bmatrix} \tag{4.16}$$

$$B^{[i]} = \begin{bmatrix} \hat{S}^+ & \hat{S}^- \end{bmatrix} \tag{4.17}$$

$$B^{[i+1]} = \begin{bmatrix} \hat{S}^- \\ \hat{S}^+ \end{bmatrix} \tag{4.18}$$

$$B^{[N]} = \begin{bmatrix} 1 \end{bmatrix} \tag{4.19}$$

Adding $\hat{h}_1$ and $\hat{h}_2$ will require us to enlarge the matrices, since $B^{[1]}$ of $\hat{h}_1 + \hat{h}_2$ needs to contain the 1 from the $B^{[1]}$ of $\hat{h}_2$. We can accommodate both the $B^{[2]}$s from $\hat{h}_1$ and $\hat{h}_2$ by writing it as:

$$\begin{bmatrix} B^{[2]}_{\hat{h}_1} & 0 \\ 0 & B^{[2]}_{\hat{h}_2} \end{bmatrix} \tag{4.20}$$

Supplementing $B^{[1]}$ and $B^{[3]}$ appropriately, the sum of $\hat{h}_1$ and $\hat{h}_2$ can be written as an MPO:

$$q = 1 \tag{4.21}$$

$$B^{[1]} = \begin{bmatrix} \hat{S}^+ & \hat{S}^- & 1 \end{bmatrix} \tag{4.22}$$

$$B^{[2]} = \begin{bmatrix} \hat{S}^- & 0 & 0 \\ \hat{S}^+ & 0 & 0 \\ 0 & \hat{S}^+ & \hat{S}^- \end{bmatrix} \tag{4.23}$$

$$B^{[3]} = \begin{bmatrix} 1 \\ \hat{S}^- \\ \hat{S}^+ \end{bmatrix} \tag{4.24}$$

$$B^{[i]} = \begin{bmatrix} 1 \end{bmatrix} \tag{4.25}$$

When we add $\hat{h}_3$, we can leave $B^{[1]}$ unchanged, since it already contains a 1. This means we can leave the row dimension of $B^{[2]}_{\hat{h}_1+\hat{h}_2+\hat{h}_3}$ unchanged. We still need to expand the column dimension though, since $B^{[2]}_{\hat{h}_3}$ contains a 1 that $B^{[2]}_{\hat{h}_1+\hat{h}_2}$ does not. $B^{[3]}$ needs to be adjusted too and for that we can use the

same tactic as in (4.20). We are then left with the MPO form of $\hat{h}_1 + \hat{h}_2 + \hat{h}_3$.

$$q = 1 \tag{4.26}$$

$$B^{[1]} = \begin{bmatrix} \hat{S}^+ & \hat{S}^- & 1 \end{bmatrix} \tag{4.27}$$

$$B^{[2]} = \begin{bmatrix} \hat{S}^- & 0 & 0 & 0 \\ \hat{S}^+ & 0 & 0 & 0 \\ 0 & \hat{S}^+ & \hat{S}^- & 1 \end{bmatrix} \tag{4.28}$$

$$B^{[3]} = \begin{bmatrix} 1 & 0 & 0 \\ \hat{S}^- & 0 & 0 \\ \hat{S}^+ & 0 & 0 \\ 0 & \hat{S}^+ & \hat{S}^- \end{bmatrix} \tag{4.29}$$

$$B^{[4]} = \begin{bmatrix} 1 \\ \hat{S}^- \\ \hat{S}^+ \end{bmatrix} \tag{4.30}$$

$$B^{[i]} = \begin{bmatrix} 1 \end{bmatrix} \tag{4.31}$$

We can keep repeating this process and we will eventually end up with the same $B^{[i]}$ for $i \notin \{1, 2, N-1, N\}$.

$$B^{[i]} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ \hat{S}^- & 0 & 0 & 0 \\ \hat{S}^+ & 0 & 0 & 0 \\ 0 & \hat{S}^+ & \hat{S}^- & 1 \end{bmatrix} \tag{4.32}$$

We can actually write $B^{[2]}$ and $B^{[N-1]}$ in this form too, if we add a 0 to $B^{[1]}$ and $B^{[N]}$. Finally, we can write the full Hamiltonian in a neat compact form.

$$q = -\frac{J}{2} \tag{4.33}$$

$$B^{[1]} = \begin{bmatrix} 0 & \hat{S}^+ & \hat{S}^- & 1 \end{bmatrix} \tag{4.34}$$

$$B^{[i]} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ \hat{S}^- & 0 & 0 & 0 \\ \hat{S}^+ & 0 & 0 & 0 \\ 0 & \hat{S}^+ & \hat{S}^- & 1 \end{bmatrix} \tag{4.35}$$

$$B^{[L]} = \begin{bmatrix} 1 \\ \hat{S}^- \\ \hat{S}^+ \\ 0 \end{bmatrix} \tag{4.36}$$

Note that this choice of matrices is not unique. For example, one could swap all of the $\hat{S}^-$ and $\hat{S}^+$ or the ones and zeros on the first rows/columns and end up with the exact same Hamiltonian.

The energy for the XXZ chain becomes a bit more complicated, since this also involves the $\hat{S}^z$ operator, but following the same steps as before, this Hamil-

tonian can also be written in MPO form:

$$q = 1 \tag{4.37}$$

$$B^{[1]} = \begin{bmatrix} 0 & -\frac{J}{2}\hat{S}^{+} & -\frac{J}{2}\hat{S}^{-} & J_z\hat{S}^{z} & 1 \end{bmatrix} \tag{4.38}$$

$$B^{[i]} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ \hat{S}^{-} & 0 & 0 & 0 & 0 \\ \hat{S}^{+} & 0 & 0 & 0 & 0 \\ \hat{S}^{z} & 0 & 0 & 0 & 0 \\ 0 & -\frac{J}{2}\hat{S}^{+} & -\frac{J}{2}\hat{S}^{-} & J_z\hat{S}^{z} & 1 \end{bmatrix} \tag{4.39}$$

$$B^{[L]} = \begin{bmatrix} 1 \\ \hat{S}^{-} \\ \hat{S}^{+} \\ \hat{S}^{z} \\ 0 \end{bmatrix} \tag{4.40}$$

4.4 Time evolution using Suzuki-Trotter decomposition

The time evolution operator evolves a state or operator from time t_0 to time t. If the Hamiltonian does not depend on time, then the operator has the following simple form:

$$\hat{U}(t) = \exp\left(-\frac{i}{\hbar}\hat{H}t\right) \tag{4.41}$$

In our case we will time-evolve a spin chain of length N and we will assume only interactions between nearest neighbours. In this case, we can write the Hamiltonian as a sum of local interaction Hamiltonians

$$\hat{U}(t) = \exp\left(-\frac{i}{\hbar}\sum_{i=1}^{N-1}\hat{h}_i t\right) \quad , \tag{4.42}$$

where $\hat{h}_i$ is the local operator that stands for the interaction between site i and site $i+1$. This needs to be converted into an MPO, but to keep the number of dimensions low, we ideally would want to split the time evolution operator into a product of operators that work only on two sites. However, we cannot just write the exponential of a sum as a product of exponentials because $\hat{h}_i$ generally does not commute with $\hat{h}_{i\pm1}$. We can group all of the local operators that do commute together, splitting the sum into an even and an odd part:

$$\hat{H}_E = \sum_{i=1}^{(N-1)/2}\hat{h}_{2i} \tag{4.43}$$

$$\hat{H}_O = \sum_{i=1}^{(N-1)/2}\hat{h}_{2i-1} \tag{4.44}$$

We want to see what the error would be, if we would simply treat $\hat{H}_E$ and $\hat{H}_O$ as commuting. To find that out, we can simply look at the Taylor expansion of

$\exp\left(\hat{A}t + \hat{B}t\right)$ and $\exp\left(\hat{A}t\right)\exp\left(\hat{B}t\right)$

$$\begin{aligned}\exp\left(\hat{A}t + \hat{B}t\right) &= 1 + \left(\hat{A}t + \hat{B}t\right) + \frac{1}{2}\left(\hat{A}t + \hat{B}t\right)^2 + \dots \\ &= 1 + \hat{A}t + \hat{B}t + \frac{1}{2}\hat{A}^2t^2 + \frac{1}{2}\hat{B}^2t^2 + \frac{1}{2}\hat{A}\hat{B}t^2 \\ &\quad + \frac{1}{2}\hat{B}\hat{A}t^2 + \dots \end{aligned} \tag{4.45}$$

$$\begin{aligned}\exp\left(\hat{A}t\right)\exp\left(\hat{B}t\right) &= \left(1 + \hat{A}t + \frac{1}{2}\hat{A}^2t^2 + \dots\right)\left(1 + \hat{B}t + \frac{1}{2}\hat{B}^2t^2 + \dots\right) \\ &= 1 + \hat{B}t + \frac{1}{2}\hat{B}^2t^2 + \hat{A}t + \hat{A}\hat{B}t^2 + \frac{1}{2}\hat{A}^2t^2 + \dots \end{aligned} \tag{4.46}$$

Subtracting these two expressions gives us the error we would make by assuming $\hat{A}$ and $\hat{B}$ commute.

$$\exp\left(\hat{A}t + \hat{B}t\right) - \exp\left(\hat{A}t\right)\exp\left(\hat{B}t\right) = \frac{1}{2}\left(\hat{B}\hat{A} - \hat{A}\hat{B}\right)t^2 + \dots$$

Using small time steps Δt in the expression above and plugging in the even and odd part of the Hamiltonian, allows us to approximate $\hat{U}(\Delta t)$.

$$\begin{aligned}\hat{U}(\Delta t) &= \exp\left(-\frac{i}{\hbar}\hat{H}_E\Delta t + \hat{H}_O\Delta t\right) \\ &= \exp\left(-\frac{i}{\hbar}\hat{H}_E\Delta t\right)\exp\left(-\frac{i}{\hbar}\hat{H}_O\Delta t\right) + \mathcal{O}(\Delta t^2)\end{aligned} \tag{4.47}$$

This is called a first order Trotter step. By applying these steps T times, one can simulate for an arbitrary time $t = T\,\Delta t$, resulting in a total accuracy of $\mathcal{O}(\Delta t)$, since $T \propto 1/\Delta t$. We can now proceed to build MPOs out of $\exp\left(-\frac{i}{\hbar}\hat{H}_E\Delta t\right)$ and $\exp\left(-\frac{i}{\hbar}\hat{H}_O\Delta t\right)$ separately. These both consist of a product of local operators, $\hat{U}_i = \exp(-\frac{i}{\hbar}\hat{h}_i t)$, that only work on two sites. Since all of these local operators work on different Hilbert spaces, it suffices to look at just one local operator, as all of the others will be the same[1]. As the Hilbert space one can use $|\sigma_i\sigma_{i+1}\rangle$, where σ_i stands for the possible spin orientations on site i. This local operator will still need to be transformed into an MPO where every matrix works on just one site. Reordering the indices for $\langle\sigma_i\sigma_{i+1}|\hat{U}_i|\sigma_i'\sigma_{i+1}'\rangle$ and performing an SVD on this allows us to do just that.

$$\begin{aligned}\langle\sigma_i\sigma_{i+1}|\hat{U}_i|\sigma_i'\sigma_{i+1}'\rangle &= U_{(\sigma_i,\sigma_i'),(\sigma_{i+1},\sigma_{i+1}')} \\ &= \sum_k \tilde{U}_{(\sigma_i,\sigma_i'),k} S_k \left(V^\dagger\right)_{k,(\sigma_{i+1},\sigma_{i+1}')}\end{aligned} \tag{4.48}$$

The singular values S_k can be multiplied into the $\tilde{U}$ and $V^\dagger$ resulting in the new matrices L and R.

$$L_k^{\sigma_i\sigma_i'} = \tilde{U}_{(\sigma_i,\sigma_i'),k}\sqrt{S_k} \tag{4.49}$$

$$R_k^{\sigma_{i+1}\sigma_{i+1}'} = \sqrt{S_k}\left(V^\dagger\right)_{k,(\sigma_{i+1},\sigma_{i+1}')} \tag{4.50}$$

[1]Except possibly at the boundaries in some special cases, but at least not in the ones treated in this thesis. Even then, it is only a boundary effect and as such can be neglected most of the times anyway.

Simply adding a dummy index, 1, gives a proper element for an MPO.

$$\hat{U}_i = \sum_k L_{1,k}^{\sigma_i \sigma_i'} R_{k,1}^{\sigma_{i+1}\sigma_{i+1}'} \tag{4.51}$$

Since this applies for all of the independent Hilbert spaces that the local operator worked on, one can write the even part of the Trotter step in MPO form as

$$\exp\left(-\frac{i}{\hbar}\hat{H}_E \Delta t\right) = \delta^{\sigma_1,\sigma_1'} L^{\sigma_2,\sigma_2'} R^{\sigma_3,\sigma_3'} L^{\sigma_4,\sigma_4'} R^{\sigma_5,\sigma_5'} L^{\sigma_6,\sigma_6'} \dots \quad , \tag{4.52}$$

and one can do the same for the odd part.

$$\exp\left(-\frac{i}{\hbar}\hat{H}_O \Delta t\right) = L^{\sigma_1,\sigma_1'} R^{\sigma_2,\sigma_2'} L^{\sigma_3,\sigma_3'} R^{\sigma_4,\sigma_4'} L^{\sigma_5,\sigma_5'} R^{\sigma_6,\sigma_6'} \dots \tag{4.53}$$

The full Trotter step can then be obtained by multiplying these two MPOs using the method from section 3.4.

The accuracy can be improved by using second order Trotter steps instead of first order Trotter steps:

$$\hat{U}(\Delta t) = \exp\left(-\frac{i}{2\hbar}\hat{H}_E \Delta t\right) \exp\left(-\frac{i}{\hbar}\hat{H}_O \Delta t\right) \exp\left(-\frac{i}{2\hbar}\hat{H}_E \Delta t\right) + \mathcal{O}(\Delta t^3) \tag{4.54}$$

The $\mathcal{O}(\Delta t^3)$ arises from the fact that $e^{A+B} - e^{A/2}e^B e^{A/2}$ only has terms of at least third order in Δt when one performs a Taylor expansion on it and substitutes A by $-\frac{i}{\hbar}\hat{H}_E \Delta t$ and B by $-\frac{i}{\hbar}\hat{H}_O \Delta t$. The total accuracy will be $\mathcal{O}(\Delta t^2)$, since one needs to perform T time steps and $T \propto 1/\Delta t$.
Another possibility is to use Forest-Ruth[20]:

$$\begin{aligned}\hat{U}(\Delta t) = &\exp\left(-\frac{i\theta}{2\hbar}\hat{H}_E \Delta t\right) \exp\left(-\frac{i\theta}{\hbar}\hat{H}_O \Delta t\right) \exp\left(-\frac{i(1-\theta)}{2\hbar}\hat{H}_E \Delta t\right) \times \\ &\exp\left(-\frac{i(1-2\theta)}{\hbar}\hat{H}_O \Delta t\right) \exp\left(-\frac{i(1-\theta)}{2\hbar}\hat{H}_E \Delta t\right) \times \qquad (4.55) \\ &\exp\left(-\frac{i\theta}{\hbar}\hat{H}_O \Delta t\right) \exp\left(-\frac{i\theta}{2\hbar}\hat{H}_E \Delta t\right) + \mathcal{O}(\Delta t^5) \qquad (4.56)\end{aligned}$$

The θ is a well chosen constant, $\theta = 1/(2 - 2^{1/3})$. Regardless of which MPO $U(\Delta t)$ is chosen, we can use it to evolve the MPOs or MPSs in time. In the Schrödinger picture, we can evolve the MPS by repeatedly applying the MPO $U(\Delta t)$.

$$|\psi(\tau\, \Delta t)\rangle = (U(\Delta t))^\tau\, |\psi\rangle \tag{4.57}$$

Similarly, we can evolve the MPO $\hat{O}$ in the Heisenberg picture by applying $U(\Delta t)$ repeatedly.

$$\hat{O}(\tau\, \Delta t)\rangle = \left(U^\dagger(\Delta t)\right)^\tau \hat{O}\, (U(\Delta t))^\tau \tag{4.58}$$

4.5 Current density

The current density can be derived from the equation of continuity.

$$\frac{d\hat{S}_i^z}{dt} + \frac{dj_i}{dx} = 0 \tag{4.59}$$

In this case the $\frac{d}{dx}$ does not mean much, since we don not have a continuous system, but we will find a discrete substitute later on. For the time derivative, we can make use of the Heisenberg equation of motion.

$$\frac{d\hat{S}_i^z}{dt} = i\left[\hat{H}, \hat{S}_i^z\right] + \frac{\partial \hat{S}_i^z}{\partial t} \tag{4.60}$$

Because there are two Hamiltonians being used, both need to be looked at separately. For the XX chain we use the Hamiltonian from (4.1):

$$\begin{aligned}\frac{d\hat{S}_i^z}{dt} &= i\left[-\frac{J}{2}\sum_j \left(\hat{S}_j^+\hat{S}_{j+1}^- + \hat{S}_{j+1}^+\hat{S}_j^-\right)\ , \quad \hat{S}_i^z\right] + 0 \\ &= -i\frac{J}{2}\Big(\left[\hat{S}_i^+, \hat{S}_i^z\right]\hat{S}_{i+1}^- + \hat{S}_{i+1}^+\left[\hat{S}_i^-, \hat{S}_i^z\right] + \\ &\qquad \hat{S}_{i-1}^+\left[\hat{S}_i^-, \hat{S}_i^z\right] + \left[\hat{S}_i^+, \hat{S}_i^z\right]\hat{S}_{i-1}^-\Big)\end{aligned} \tag{4.61}$$

Using the fact that $\left[\hat{S}_i^\pm, \hat{S}_i^z\right] = \mp\hat{S}_i^\pm$, we get the following expression:

$$\frac{d\hat{S}_i^z}{dt} = -i\frac{J}{2}\left(-\hat{S}_i^+\hat{S}_{i+1}^- + \hat{S}_{i-1}^+\hat{S}_i^- - \text{h.c.}\right) \tag{4.62}$$

Now we need to evaluate the spatial derivative in (4.59). Since the coordinates in our chain are the integer index i, we cannot evaluate the continuous derivative. We therefore simply define

$$\frac{dj_i}{dx} = \frac{1}{a}\left(\hat{j}_{i+1} - \hat{j}_i\right) \quad , \tag{4.63}$$

where a is the lattice constant[2]. Putting all of this in (4.59) gives a recursive relation for the current density operator:

$$\hat{j}_i = \hat{j}_{i+1} - i\frac{aJ}{2}\left(-\hat{S}_i^+\hat{S}_{i+1}^- + \hat{S}_{i-1}^+\hat{S}_i^- - \text{h.c.}\right) \tag{4.64}$$

To get a proper expression, one can look at $\hat{j}_N$.

$$\hat{j}_N = 0 - i\frac{aJ}{2}\left(-0 + \hat{S}_{N-1}^+\hat{S}_N^- - \text{h.c.}\right) = -i\frac{aJ}{2}\left(\hat{S}_{N-1}^+\hat{S}_N^- - \text{h.c.}\right) \tag{4.65}$$

[2]There are various possibilities to define this, one could also use the more symmetric $\frac{dj_i}{dx} = \frac{1}{2a}\left(\hat{j}_{i+1} - \hat{j}_{i-1}\right)$, but since we are only interested in comparing simulations, this does not matter.

The next step would be

$$\begin{aligned}\hat{j}_{N-1} &= \hat{j}_N - i\frac{aJ}{2}\left(-\hat{S}^+_{N-1}\hat{S}^-_N + \hat{S}^+_{N-2}\hat{S}^-_{N-1} - \text{h.c.}\right) \\ &= -i\frac{aJ}{2}\left(\hat{S}^+_{N-1}\hat{S}^-_N - \text{h.c.}\right) - i\frac{aJ}{2}\left(-\hat{S}^+_{N-1}\hat{S}^-_N + \hat{S}^+_{N-2}\hat{S}^-_{N-1} - \text{h.c.}\right) \\ &= -i\frac{aJ}{2}\left(\hat{S}^+_{N-2}\hat{S}^-_{N-1} - \text{h.c.}\right) \end{aligned} \tag{4.66}$$

Continuing this, we eventually end up with:

$$\hat{j}_i = -\frac{iaJ}{2}\left(\hat{S}^+_{i-1}\hat{S}^-_i - \text{h.c.}\right) = -\frac{iaJ}{2}\left(\hat{S}^+_{i-1}\hat{S}^-_i - \hat{S}^+_i\hat{S}^-_{i-1}\right) \tag{4.67}$$

Only $\hat{j}_1$ will just be zero, since $\hat{S}^\pm_{i-1}$ does not exist. For the XXZ chain, the Hamiltonian from (4.2) is used.

$$\begin{aligned}\frac{d\hat{S}^z_i}{dt} &= i\left[\hat{H}, \hat{S}^z_i\right] + \frac{\partial \hat{S}^z_i}{\partial t} \\ &= i\left[-\frac{J}{2}\sum_j\left(\hat{S}^+_j\hat{S}^-_{j+1} + \hat{S}^+_{j+1}\hat{S}^-_j\right) + J_z\sum_j \hat{S}^z_j\hat{S}^z_{j+1}\,, \quad \hat{S}^z_i\right] + 0 \\ &= -i\frac{J}{2}\Big(\left[\hat{S}^+_i, \hat{S}^z_i\right]\hat{S}^-_{i+1} + \hat{S}^+_{i+1}\left[\hat{S}^-_i, \hat{S}^z_i\right] + \\ &\qquad \hat{S}^+_{i-1}\left[\hat{S}^-_i, \hat{S}^z_i\right] + \left[\hat{S}^+_i, \hat{S}^z_i\right]\hat{S}^-_{i-1}\Big) + \\ &\qquad iJ_z\sum_j\left(\hat{S}^z_j\left[\hat{S}^z_{j+1}, \hat{S}^z_i\right] + \left[\hat{S}^z_j, \hat{S}^z_i\right]\hat{S}^z_{j+1}\right) \\ &= -i\frac{J}{2}\Big(\left[\hat{S}^+_i, \hat{S}^z_i\right]\hat{S}^-_{i+1} + \hat{S}^+_{i+1}\left[\hat{S}^-_i, \hat{S}^z_i\right] + \\ &\qquad \hat{S}^+_{i-1}\left[\hat{S}^-_i, \hat{S}^z_i\right] + \left[\hat{S}^+_i, \hat{S}^z_i\right]\hat{S}^-_{i-1}\Big) + 0 \end{aligned} \tag{4.68}$$

This is in fact the same expression as (4.61), so we get exactly the same operator for the current density. The final step for both Hamiltonians is to write this operator in MPO form:

$$q = -\frac{iaJ}{2} \tag{4.69}$$

$$B^{[j]} = \begin{bmatrix} 1 \end{bmatrix} \qquad \text{for } j \neq i \wedge j \neq i-1 \tag{4.70}$$

$$B^{[i-1]} = \begin{bmatrix} \hat{S}^+ & -\hat{S}^- \end{bmatrix} \tag{4.71}$$

$$B^{[i]} = \begin{bmatrix} \hat{S}^- \\ \hat{S}^+ \end{bmatrix} \tag{4.72}$$

The procedure for the Bose-Hubbard model is very similar. Instead of the operator $\hat{S}^z_i$ we only need to use the number operator $\hat{n}_i$ in the equation of continuity and the Heisenberg equation of motion, which means we need to

calculate $\left[\hat{H}, \hat{n}_i\right]$.

$$\begin{aligned}
\left[\hat{H}, \hat{n}_i\right] &= \left[-t \sum_j \left(\hat{b}_j^\dagger \hat{b}_{j+1} + \hat{b}_{j+1}^\dagger \hat{b}_j\right) - \frac{\hat{U}}{2} \sum_j \hat{n}_j \left(\hat{n}_j - 1\right) - \mu \sum_j \hat{n}_j \ , \hat{n}_i\right] \\
&= \left[-t \sum_j \left(\hat{b}_j^\dagger \hat{b}_{\hat{j+1}} + \hat{b}_{j+1}^\dagger \hat{b}_j\right) \ , \hat{b}_i^\dagger \hat{b}_i\right] \\
&= -t \left(\hat{b}_i^\dagger \hat{b}_{i+1} \hat{b}_i^\dagger \hat{b}_i + \hat{b}_{i+1}^\dagger \hat{b}_i \hat{b}_i^\dagger \hat{b}_i - \hat{b}_i^\dagger \hat{b}_i \hat{b}_i^\dagger \hat{b}_{i+1} - \hat{b}_i^\dagger \hat{b}_i \hat{b}_{i+1}^\dagger \hat{b}_i + \right. \\
&\quad \left. \hat{b}_{i-1}^\dagger \hat{b}_i \hat{b}_i^\dagger \hat{b}_i + \hat{b}_i^\dagger \hat{b}_{i-1} \hat{b}_i^\dagger \hat{b}_i - \hat{b}_i^\dagger \hat{b}_i \hat{b}_{i-1}^\dagger \hat{b}_i - \hat{b}_i^\dagger \hat{b}_i \hat{b}_i^\dagger \hat{b}_{i-1}\right) \\
&= -t \left(\hat{b}_i^\dagger \hat{b}_i^\dagger \hat{b}_i \hat{b}_{i+1} + \left(1 + \hat{b}_i^\dagger \hat{b}_i\right) \hat{b}_i \hat{b}_{i+1}^\dagger - \hat{b}_i^\dagger \left(1 + \hat{b}_i^\dagger \hat{b}_i\right) \hat{b}_{i+1} - \right. \\
&\quad \hat{b}_i^\dagger \hat{b}_i \hat{b}_i \hat{b}_{i+1}^\dagger + \hat{b}_{i-1}^\dagger \left(1 + \hat{b}_i^\dagger \hat{b}_i\right) \hat{b}_i + \hat{b}_{i-1} \hat{b}_i^\dagger \hat{b}_i^\dagger \hat{b}_i - \\
&\quad \left. \hat{b}_{i-1}^\dagger \hat{b}_i^\dagger \hat{b}_i \hat{b}_i - \hat{b}_{i-1} \hat{b}_i^\dagger \left(1 + \hat{b}_i^\dagger \hat{b}_i\right)\right) \\
&= -t \left(\hat{b}_i \hat{b}_{i+1}^\dagger - \hat{b}_i^\dagger \hat{b}_{i+1} + \hat{b}_{i-1}^\dagger \hat{b}_i - \hat{b}_{i-1} \hat{b}_i^\dagger\right) \qquad (4.73)
\end{aligned}$$

Putting this in the Heisenberg equation of motion and subsequently in the equation of continuity gives us

$$\frac{dj_i}{dx} = i\, t \left(\hat{b}_i \hat{b}_{i+1}^\dagger - \hat{b}_i^\dagger \hat{b}_{i+1} + \hat{b}_{i-1}^\dagger \hat{b}_i - \hat{b}_{i-1} \hat{b}_i^\dagger\right) \quad . \qquad (4.74)$$

Using the same definition in equation (4.63) allows us to get rid of the continuous derivative.

$$\frac{1}{a} \left(\hat{j}_{i+1} - \hat{j}_i\right) = i\, t \left(\hat{b}_i \hat{b}_{i+1}^\dagger - \hat{b}_i^\dagger \hat{b}_{i+1} + \hat{b}_{i-1}^\dagger \hat{b}_i - \hat{b}_{i-1} \hat{b}_i^\dagger\right) \qquad (4.75)$$

Again applying the boundary condition $j_{N+1} = 0$, we have:

$$\hat{j}_N = i\, a\, t \left(0 - 0 + \hat{b}_{N-1}^\dagger \hat{b}_N - \hat{b}_{N-1} \hat{b}_N^\dagger\right) \qquad (4.76)$$

This leaves us with the final expression of the current density in the Bose-Hubbard model:

$$\hat{j}_i = i\, a\, t \left(\hat{b}_{i-1}^\dagger \hat{b}_i - \hat{b}_{i-1} \hat{b}_i^\dagger\right) \qquad (4.77)$$

This is very similar to (4.61), so it comes as no surprise that the MPO-form is also very similar.

$$q = -i\, a\, t \qquad (4.78)$$

$$B^{[j]} = \begin{bmatrix} 1 \end{bmatrix} \qquad \text{for } j \neq i \wedge j \neq i - 1 \qquad (4.79)$$

$$B^{[i-1]} = \begin{bmatrix} \hat{b}^\dagger & -\hat{b} \end{bmatrix} \qquad (4.80)$$

$$B^{[i]} = \begin{bmatrix} \hat{b} \\ \hat{b}^\dagger \end{bmatrix} \qquad (4.81)$$

Chapter 5

Schrödinger versus Heisenberg

5.1 The XX(Z) chain with initial inhomogeneous state

To compare the use of MPOs and MPSs in the Schrödinger picture and in the Heisenberg picture, we shall look at the time evolution of both a spin-$\frac{1}{2}$ XX chain and a spin-$\frac{1}{2}$ XXZ chain, given a specific initial state. Since the results for the spin-$\frac{1}{2}$ XX chain can be calculated for finite size spin chains (actually also for infinite size, but we will not use that here), we can compare the theory to the outcome of the simulations. The Hamiltonian for the XX chain was already given in (4.1):

$$\hat{H} = -\frac{J}{2}\sum_i \left(\hat{S}_i^+\hat{S}_{i+1}^- + \hat{S}_{i+1}^+\hat{S}_i^-\right) \tag{5.1}$$

The initial state we shall use is

$$|\psi(0)\rangle = |\uparrow\uparrow\ldots\uparrow\downarrow\downarrow\ldots\downarrow\rangle \quad . \tag{5.2}$$

That is to say, for a chain of N sites, the first $N/2$ sites will have spin $\uparrow$ and the remaining $N/2$ sites will have spin $\downarrow$. To study the difference between the Schrödinger and Heisenberg picture, we will compare the expectation values for the density at site n, $\langle\hat{S}_n^z\rangle$, the density correlation between two sites m and n, $\langle\hat{S}_m^z\hat{S}_n^z\rangle$, and the current density $\langle\hat{j}_n\rangle$. For ease of readability and for simplified calculations in the numerical simulation, we shall now set $\hbar = 1$, $J = 1$ and $a = 1$, which means the physical time in the simulations is in units of $\hbar$ and J, while the scale of the current density $\hat{j}_i$ is in units of $\hbar$, J and a.

5.1.1 Procedure

For convenience, we shall use the convention that site 1 refers to the first site to the right of the boundary between $\uparrow$ and $\downarrow$. This state can be represented as an MPS using a simple set of matrices. The matrices for the first site at $i = -N/2+1$ are

$$A^{\uparrow_{-N/2+1}} = \begin{pmatrix} 1 \end{pmatrix} \quad A^{\downarrow_{-N/2+1}} = \begin{pmatrix} 0 \end{pmatrix} \tag{5.3}$$

and the matrices for sites $i = -N/2 + 1$ to $i = 0$ are the same:

$$A^{\uparrow_i} = \begin{pmatrix} 1 \end{pmatrix} \quad A^{\downarrow_i} = \begin{pmatrix} 0 \end{pmatrix} \tag{5.4}$$

The matrices for sites $i = 1$ to $i = N/2 - 1$ are

$$A^{\uparrow_i} = \begin{pmatrix} 0 \end{pmatrix} \quad A^{\downarrow_i} = \begin{pmatrix} 1 \end{pmatrix} \tag{5.5}$$

and for the same for the last site at $i = N/2$:

$$A^{\uparrow_{N/2}} = \begin{pmatrix} 0 \end{pmatrix} \quad A^{\downarrow_{N/2}} = \begin{pmatrix} 1 \end{pmatrix} \tag{5.6}$$

The prefactor p for this MPS is simply 1.

For the numerical simulation, the Trotter steps need to be constructed according to the recipe from section 4.4. For the XX chain, one can easily get the local interaction Hamiltonians from (4.1):

$$\hat{h}_i = -\frac{1}{2}\left(\hat{S}_i^+ \hat{S}_{i+1}^- + \hat{S}_{i+1}^+ \hat{S}_i^-\right) \tag{5.7}$$

As basis we use $|\sigma_i \sigma_{i+1}\rangle = \{|\uparrow\uparrow\rangle, |\uparrow\downarrow\rangle, |\downarrow\uparrow\rangle, |\downarrow\downarrow\rangle\}$, which allows us to write $\hat{h}_i$ in matrix form.

$$\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & -\frac{1}{2} & 0 \\ 0 & -\frac{1}{2} & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{5.8}$$

For the XXZ chain one can likewise get the local interaction Hamiltonians from (4.2).

$$\hat{h}_i = -\frac{1}{2}\left(\hat{S}_i^+ \hat{S}_{i+1}^- + \hat{S}_{i+1}^+ \hat{S}_i^-\right) + J_z \hat{S}_i^z \hat{S}_{i+1}^z \tag{5.9}$$

Using the same basis as before, we arrive at $\hat{h}_i$ in matrix form.

$$\begin{bmatrix} \frac{J_z}{4} & 0 & 0 & 0 \\ 0 & -\frac{J_z}{4} & -\frac{1}{2} & 0 \\ 0 & -\frac{1}{2} & -\frac{J_z}{4} & 0 \\ 0 & 0 & 0 & \frac{J_z}{4} \end{bmatrix} \tag{5.10}$$

From either of these, we can calculate $\hat{U}_i = \exp(-i\hat{h}_i t)$ by diagonalizing.

$$\hat{U}_i = P \begin{bmatrix} e^{-ie_1\tau} & & & 0 \\ & e^{-ie_2\tau} & & \\ & & e^{-ie_3\tau} & \\ 0 & & & e^{-ie_n\tau} \end{bmatrix} P^{-1} \tag{5.11}$$

Here e_i are the four eigenvalues of h_i and P is the matrix with the eigenvectors. Inserting the matrix elements in (4.48), we can do the SVD to get the required $L^{\sigma_i,\sigma_i'}$ and $R^{\sigma_{i+1},\sigma_{i+1}'}$ matrices. From these matrices, we can construct the MPO $\hat{U}(\Delta t)$ using either (4.47), (4.54) or (4.55). In both the Schrödinger and the Heisenberg picture, the resulting MPS/MPO needs to be compressed after every multiplication using the methods from section 3.7. Otherwise the dimensions of the matrices making up the MPS/MPO, will grow like 4^T, where T is the number of time steps.

5.1.2 Theory

To be able to say anything about the validity of the simulations, they need to be compared to theoretical predictions, if available. Luckily, these exist in the case of the XX chain at zero temperature with initial state $|\uparrow\uparrow\ldots\uparrow\downarrow\ldots\downarrow\downarrow\rangle$. For this, the Hamiltonian from (4.1) needs to be written in a more convenient form, allowing for simpler calculations. This is called a Jordan-Wigner transformation[21]; it transforms the operators S_i^- and S_i^+ into the fermionic operators c_i and $c_i^\dagger$ respectively. The advantage of this form is that, during the calculation of the various expectation values, we can now use anticommutation relations for just two operators, instead of the more complicated commutation relations for the three operators $\hat{S}^+$, $\hat{S}^-$ and $\hat{S}^z$. The Jordan-Wigner transformation is defined as follows:

$$\hat{S}_n^+ = \hat{c}_n^\dagger (-1)^{\phi_n} \tag{5.12}$$

$$\hat{S}_n^- = (-1)^{\phi_n} \hat{c}_n \tag{5.13}$$

$$\hat{S}_n^z = \hat{c}_n^\dagger \hat{c}_n - \frac{1}{2} \quad , \tag{5.14}$$

with $\phi_n = \sum_{j=-N/2+1}^{n-1} \hat{c}_j^\dagger \hat{c}_j$. Putting this in the Hamiltonian gives

$$\hat{H} = -\frac{J}{2} \sum_{n=-N/2+1}^{N/2} \hat{c}_n^\dagger (-1)^{\phi_n} (-1)^{\phi_{n+1}} \hat{c}_{n+1} + \text{h.c.} \tag{5.15}$$

The part $(-1)^{\phi_n}(-1)^{\phi_{n+1}}$ can be simplified, since $[\phi_n, \phi_m] = 0$.

$$(-1)^{\phi_n}(-1)^{\phi_{n+1}} = (-1)^{\phi_n+\phi_{n+1}} = (-1)^{2\sum_{j=1}^{n-1} \hat{c}_j^\dagger \hat{c}_j + \hat{c}_n^\dagger \hat{c}_n} = (-1)^{\hat{c}_n^\dagger \hat{c}_n} \tag{5.16}$$

Continuing,

$$\begin{aligned}
\hat{H} &= -\frac{J}{2} \sum_{n=-N/2+1}^{N/2} \hat{c}_n^\dagger (-1)^{\hat{c}_n^\dagger \hat{c}_n} \hat{c}_{n+1} + \text{h.c.} \\
&= -\frac{J}{2} \sum_{n=-N/2+1}^{N/2} \big(\hat{c}_n^\dagger 1 \hat{c}_{n+1} + \hat{c}_n^\dagger i\pi \hat{c}_n^\dagger \hat{c}_n \hat{c}_{n+1} + \\
&\qquad \hat{c}_n^\dagger (-\pi^2) \hat{c}_n^\dagger \hat{c}_n \hat{c}_n^\dagger \hat{c}_n \hat{c}_{n+1} + \ldots \big) + \text{h.c.}
\end{aligned} \tag{5.17}$$

In the last step, a Taylor expansion was used for the $(-1)^{\hat{c}_n^\dagger \hat{c}_n}$. Every term except the first one starts with $\left(\hat{c}_n^\dagger\right)^2$, so all of these terms are zero, since $\hat{c}_n$ is a fermionic operator.

$$\hat{H} = -\frac{J}{2} \sum_{n=-N/2+1}^{N/2} \hat{c}_n^\dagger \hat{c}_{n+1} + \text{h.c.} \tag{5.18}$$

The time evolution of the initial state can now be calculated, since this Hamiltonian is of the form

$$\hat{H} = \sum_{i,j}^{N} T_{i,j} \hat{c}_i^\dagger \hat{c}_j \tag{5.19}$$

Hamiltonians of this form can be solved exactly by diagonalizing only the $N \times N$ matrix T instead of the usual $2^N \times 2^N$ matrix belonging to (4.1). The matrix T for (5.18) can simply be read off:

$$T = -\frac{J}{2} \begin{bmatrix} 0 & 1 & 0 & 0 & & & & 0 \\ 1 & 0 & 1 & 0 & & & & \\ 0 & 1 & 0 & 1 & & & & \\ 0 & 0 & 1 & 0 & & & & \\ & & & & \ddots & & & \\ & & & & & 0 & 1 & 0 & 0 \\ & & & & & 1 & 0 & 1 & 0 \\ & & & & & 0 & 1 & 0 & 1 \\ 0 & & & & & 0 & 0 & 1 & 0 \end{bmatrix} \tag{5.20}$$

The eigenvalues are the energies, ϵ_k, and the normalized eigenvectors of T form a unitary transformation U because T is Hermitian. We can use U to transform the creation operators, $\hat{c}_i$, in the original basis into the new diagonalized creation operators, $\hat{d}_i$. This yields a diagonal Hamiltonian:

$$\hat{H} = \sum_{i,j=-N/2+1}^{N/2} \hat{c}_i^\dagger T_{i,j} \hat{c}_j = \sum_{i,j} \sum_{k=1}^{N} \hat{c}_i^\dagger U_{i,k} \epsilon_k \left(U^\dagger\right)_{k,j} \hat{c}_j = \sum_{k=1}^{N} \epsilon_k \hat{d}_k^\dagger \hat{d}_k \quad , \tag{5.21}$$

where $\hat{d}_k^\dagger = \sum_i \hat{c}_i^\dagger U_{i,k}$ and $\hat{d}_k = \sum_j \left(U^\dagger\right)_{k,j} \hat{c}_j$. The old operators can thus be written as

$$\hat{c}_i^\dagger = \sum_{k=1}^{N} \hat{d}_k^\dagger \left(U^\dagger\right)_{k,i} \tag{5.22}$$

$$\hat{c}_i = \sum_{k=1}^{N} U_{i,k} \hat{d}_k \tag{5.23}$$

Note that the k-indices can simply run from 1 to N, since the index k does not have any special meaning. We can then proceed to study the time evolution of the initial state. In our case the initial state is $|\psi\rangle(0) = |\uparrow\uparrow \ldots \uparrow\downarrow \ldots \downarrow\downarrow\rangle$, which can be written as

$$|\psi\rangle(0) = \prod_{i=-N/2+1}^{0} \hat{c}_i^\dagger |\downarrow\rangle \tag{5.24}$$

Note that this notation has some ambiguity with the sign, since the $\hat{c}_i^\dagger$ are fermionic operators, but this does not matter, since we are interested in the expectation values only and the sign will drop out anyway. In the new basis (5.24) becomes

$$|\psi\rangle(0) = \prod_{i=-N/2+1}^{0} \sum_{k=1}^{N} \hat{d}_k^\dagger \left(U^\dagger\right)_{k,i} |\downarrow\rangle \tag{5.25}$$

We know the time evolution of $\hat{d}_k$, because of (5.21) and the Heisenberg equation of motion.

$$\begin{aligned}
\frac{d\,\hat{d}_k}{dt} &= i\left[\hat{H}, \hat{d}_k\right] + \frac{\partial \hat{d}_k}{\partial t} \\
&= i\sum_{\ell=1}^{N} \epsilon_\ell \left[\hat{d}_\ell^\dagger \hat{d}_\ell, \hat{d}_k\right] + 0 \\
&= i\sum_{\ell=1}^{N} \epsilon_\ell \left(\hat{d}_\ell^\dagger \hat{d}_\ell \hat{d}_k - \hat{d}_k \hat{d}_\ell^\dagger \hat{d}_\ell\right) \\
&= i\sum_{\ell=1}^{N} \epsilon_\ell \left(\hat{d}_\ell^\dagger \hat{d}_\ell \hat{d}_k - \left(\delta_{k,\ell} - \hat{d}_\ell^\dagger \hat{d}_k\right)\hat{d}_\ell\right) \\
&= i\sum_{\ell=1}^{N} \epsilon_\ell \left(\hat{d}_\ell^\dagger \hat{d}_\ell \hat{d}_k - \delta_{k,\ell}\hat{d}_\ell - \hat{d}_\ell^\dagger \hat{d}_\ell \hat{d}_k\right) \\
&= -i\,\epsilon_k \hat{d}_k
\end{aligned} \tag{5.26}$$

Solving this differential equation gives the time evolution of $\hat{d}_k$.

$$\hat{d}_k(t) = \hat{d}_k(0) e^{-i\epsilon_k t} \tag{5.27}$$

This can be inserted in (5.25) to get the time evolution of the initial state.

$$|\psi\rangle(t) = \prod_{i=-N/2+1}^{0} \sum_{k=1}^{N} \hat{d}_k^\dagger(0) e^{i\epsilon_k t} \left(U^\dagger\right)_{k,i} |\downarrow\rangle \tag{5.28}$$

Now we need to go back to the original operators again, since the expectation values we are interested in, such as $\langle \hat{S}_n^z \rangle = \langle \hat{c}_n^\dagger \hat{c}_n \rangle - \frac{1}{2}$, are all given in $\hat{c}_n$. We have,

$$|\psi\rangle(t) = \prod_{i=-N/2+1}^{0} \sum_{k=1}^{N} \sum_{\ell=-N/2+1}^{N/2} \hat{c}_\ell^\dagger U_{\ell,k} e^{i\epsilon_k t} \left(U^\dagger\right)_{k,i} |\downarrow\rangle \tag{5.29}$$

Density operator $\hat{S}_n^z$

From (5.14) it is already known what the density operator looks like expressed in the fermionic annihilation operators $\hat{c}_n$, which is $\hat{S}_n^z = \hat{c}_n^\dagger \hat{c}_n - \frac{1}{2}$. We can then proceed to calculate the expectation value.

$$\begin{aligned}
&\langle\psi|\hat{S}_n^z|\psi\rangle(t) = \\
&\left\langle \downarrow \left| \prod_{j=-N/2+1}^{0} \sum_{k'=1}^{N} \sum_{\ell'=-N/2+1}^{N/2} \hat{c}_{\ell'} \left(U^\dagger\right)_{k',\ell'} e^{i\epsilon_{k'} t} U_{j,k'} \right| \hat{c}_n^\dagger \times \right. \\
&\left. \hat{c}_n \left| \prod_{i=-N/2+1}^{0} \sum_{k=1}^{N} \sum_{\ell=-N/2+1}^{N/2} \hat{c}_\ell^\dagger U_{\ell,k} e^{-i\epsilon_k t} \left(U^\dagger\right)_{k,i} \right| \downarrow \right\rangle - \frac{1}{2}
\end{aligned} \tag{5.30}$$

Introducing $A_{\ell,i}(t) = \sum_{k=1}^{N} U_{\ell,k} e^{i\epsilon_k t} \left(U^\dagger\right)_{k,i}$ simplifies the notation somewhat. Note that this means that $A(t) = e^{iTt}$. We only need to look at the right half

of the above expression, since the left half is simply the complex conjugate of the right half.

$$
\begin{aligned}
|\Psi_n\rangle &= \hat{c}_n \prod_{i=-N/2+1}^{0} \sum_{k=1}^{N} \sum_{\ell=-N/2+1}^{N/2} \hat{c}_\ell^\dagger U_{\ell,k} e^{-i\epsilon_k t} \left(U^\dagger\right)_{k,i} |\downarrow\rangle \\
&= \hat{c}_n \prod_{i=-N/2+1}^{0} \sum_{\ell=-N/2+1}^{N/2} \hat{c}_\ell^\dagger A_{\ell,i}(t) |\downarrow\rangle \\
&= \hat{c}_n \left(\sum_{\ell_1=-N/2+1}^{N/2} \hat{c}_{\ell_1}^\dagger A_{\ell_1,-N/2+1}(t) \right) \quad \times \\
&\qquad \left(\sum_{\ell_2=-N/2+1}^{N/2} \hat{c}_{\ell_2}^\dagger A_{\ell_2,-N/2+2}(t) \right) \dots \quad \times \\
&\qquad\qquad \dots \left(\sum_{\ell_{N/2}=-N/2+1}^{N/2} \hat{c}_{\ell_{N/2}}^\dagger A_{\ell_{N/2},0}(t) \right) |\downarrow\rangle \\
&= \left(A_{n,-N/2+1}(t) - \sum_{\ell_1} \hat{c}_{\ell_1}^\dagger A_{\ell_1,-N/2+1}(t) \hat{c}_n \right) \quad \times \\
&\qquad \left(\sum_{\ell_2} \hat{c}_{\ell_2}^\dagger A_{\ell_2,-N/2+2}(t) \right) \dots \quad \times \\
&\qquad\qquad \dots \left(\sum_{\ell_{N/2}} \hat{c}_{\ell_{N/2}}^\dagger A_{\ell_{N/2},0}(t) \right) |\downarrow\rangle
\end{aligned} \tag{5.31}
$$

Introducing another shorthand $\hat{B}_i^\dagger = \sum_{\ell=-N/2+1}^{N/2} \hat{c}_\ell^\dagger A_{\ell,i}(t)$, we get a more compact form:

$$
\begin{aligned}
&= A_{n,-N/2+1}(t) \hat{B}_{-N/2+2}^\dagger \dots \hat{B}_0^\dagger |\downarrow\rangle - \hat{B}_{-N/2+1}^\dagger \hat{c}_n \quad \times \\
&\qquad \left(\sum_{\ell_2=1}^{N} \hat{c}_{\ell_2}^\dagger A_{\ell_2,-N/2+2}(t) \right) \dots \left(\sum_{\ell_{N/2}=1}^{N} \hat{c}_{\ell_{N/2}}^\dagger A_{\ell_{N/2},0}(t) \right) |\downarrow\rangle \\
&= A_{n,-N/2+1}(t) \hat{B}_{-N/2+2}^\dagger \dots \hat{B}_0^\dagger |\downarrow\rangle \quad - \\
&\qquad \hat{B}_{-N/2+1}^\dagger A_{n,-N/2+2}(t) \hat{B}_{-N/2+3}^\dagger \dots \hat{B}_0^\dagger |\downarrow\rangle \quad + \\
&\qquad \hat{B}_{-N/2+1}^\dagger \hat{B}_{-N/2+2}^\dagger \hat{c}_n \left(\sum_{\ell_3=1}^{N} \hat{c}_{\ell_3}^\dagger A_{\ell_3,-N/2+3}(t) \right) \dots \quad \times \\
&\qquad \dots \left(\sum_{\ell_{N/2}=1}^{N} \hat{c}_{\ell_{N/2}}^\dagger A_{\ell_{N/2},0}(t) \right) |\downarrow\rangle \\
&= A_{n,-N/2+1}(t) \hat{B}_{-N/2+2}^\dagger \dots \hat{B}_0^\dagger |\downarrow\rangle \quad - \\
&\qquad \hat{B}_{-N/2+1}^\dagger A_{n,-N/2+2}(t) \hat{B}_{-N/2+3}^\dagger \dots \hat{B}_0^\dagger |\downarrow\rangle \quad + \\
&\qquad \hat{B}_{-N/2+1}^\dagger \dots \hat{B}_{-1}^\dagger A_{n,0}(t) |\downarrow\rangle
\end{aligned} \tag{5.32}
$$

The shorthand $\hat{B}_i^\dagger$ turns out to be a fermionic creation operator in itself.

$$\begin{aligned}\left\{\hat{B}_i^\dagger, \hat{B}_j^\dagger\right\} &= \left\{\sum_{\ell=-N/2+1}^{N/2} \hat{c}_\ell^\dagger A_{\ell,i}(t), \sum_{\ell'=-N/2+1}^{N/2} \hat{c}_{\ell'}^\dagger A_{\ell',j}(t)\right\} \\ &= \sum_{\ell=-N/2+1}^{N/2}\sum_{\ell'=-N/2+1}^{N/2} A_{\ell,i}(t)A_{\ell',j}(t)\left\{\hat{c}_\ell^\dagger, \hat{c}_{\ell'}^\dagger\right\} \\ &= 0 \end{aligned} \tag{5.33}$$

$$\begin{aligned}\left\{\hat{B}_i, \hat{B}_j\right\} &= \left\{\sum_{\ell=-N/2+1}^{N/2} \hat{c}_\ell A_{i,\ell}^*(t), \sum_{\ell'=-N/2+1}^{N/2} \hat{c}_{\ell'} A_{j,\ell'}^*(t)\right\} \\ &= \sum_{\ell=-N/2+1}^{N/2}\sum_{\ell'=-N/2+1}^{N/2} A_{i,\ell}^\star(t)A_{j,\ell'}^\star(t)\left\{\hat{c}_\ell, \hat{c}_{\ell'}\right\} \\ &= 0 \end{aligned} \tag{5.34}$$

$$\begin{aligned}\left\{\hat{B}_i, \hat{B}_j^\dagger\right\} &= \left\{\sum_{\ell=-N/2+1}^{N/2} \hat{c}_\ell A_{i,\ell}^\star(t), \sum_{\ell'=-N/2+1}^{N/2} \hat{c}_{\ell'}^\dagger A_{\ell',j}(t)\right\} \\ &= \sum_{\ell=-N/2+1}^{N/2}\sum_{\ell'=-N/2+1}^{N/2} A_{i,\ell}^\star(t)A_{\ell',j}(t)\left\{\hat{c}_\ell, \hat{c}_{\ell'}^\dagger\right\} \\ &= \sum_{\ell=-N/2+1}^{N/2}\sum_{\ell'=-N/2+1}^{N/2} A_{i,\ell}^\star(t)A_{\ell',j}(t)\delta_{\ell,\ell'} \\ &= \sum_{\ell=-N/2+1}^{N/2} A_{i,\ell}^\star(t)A_{\ell,j}(t) \\ &= \left(e^{-iT^\dagger t}e^{iTt}\right)_{i,j} \\ &= \left(e^{-iTt}e^{iTt}\right)_{i,j} \\ &= \delta_{i,j} \end{aligned} \tag{5.35}$$

In the last steps, we used the fact that T is Hermitian, i.e. $T = T^\dagger$. The fact that the $\hat{B}_i^\dagger$ are fermionic creation operator means that every term in (5.32) creates a different state, which are orthonormal to the states created by the other terms.

$$\begin{aligned}\langle\psi|\hat{S}_n^z|\psi\rangle(t) &= \langle\Psi_n|\Psi_n\rangle - \frac{1}{2} \\ &= \sum_{i=-N/2+1}^{0} |A_{n,i}(t)|^2 - \frac{1}{2} \\ &= \sum_{i=-N/2+1}^{0} \left|\sum_{k=1}^{N} U_{n,k}e^{i\epsilon_k t}\left(U^\dagger\right)_{k,i}\right|^2 - \frac{1}{2} \end{aligned} \tag{5.36}$$

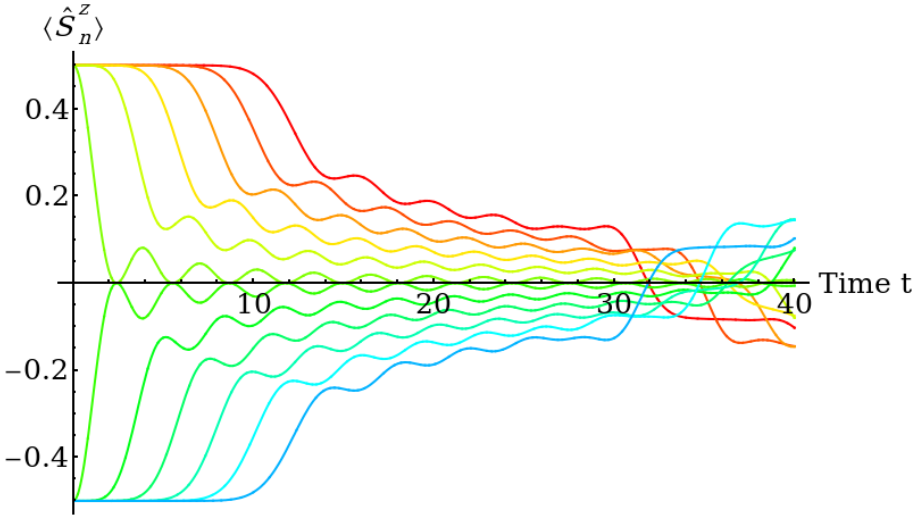

Figure 5.1: The analytical solution for various expectation values $\langle \hat{S}_n^z \rangle$. Shown is $n = -10$, -8, -6, -4, -2, 0, 1, 3, 5, 7, 9 and 11 from bottom to top on a chain with length $N = 40$. One can clearly see the boundary effect from $t = 30$. One can also see that the boundary effects come into play later for smaller $|n|$.

The result of this has been plotted in figure 5.1 for $N = 40$.

Correlation operator $\hat{S}_m^z \hat{S}_n^z$

Writing out the correlation operator in the fermionic creation/annihilation operators $\hat{c}_n^\dagger$ and $\hat{c}_n$ gives

$$\begin{aligned}\langle \psi | \hat{S}_m^z \hat{S}_n^z | \psi \rangle &= \langle \psi | \left(\hat{c}_m^\dagger \hat{c}_m - \frac{1}{2} \right) \left(\hat{c}_n^\dagger \hat{c}_n - \frac{1}{2} \right) | \psi \rangle \\ &= \langle \psi | \hat{c}_m^\dagger \hat{c}_m \hat{c}_n^\dagger \hat{c}_n | \psi \rangle - \frac{1}{2} \langle \psi | \hat{c}_m^\dagger \hat{c}_m | \psi \rangle - \frac{1}{2} \langle \psi | \hat{c}_n^\dagger \hat{c}_n | \psi \rangle + \frac{1}{4} \end{aligned} \tag{5.37}$$

The terms of the form $\langle \psi | \hat{c}_i^\dagger \hat{c}_i | \psi \rangle$ are already known from the calculation of the density operator. For the first term we can apply Wick's factorization.

$$\langle \psi | \hat{c}_m^\dagger \hat{c}_m \hat{c}_n^\dagger \hat{c}_n | \psi \rangle = \langle \psi | \hat{c}_m^\dagger \hat{c}_m | \psi \rangle \langle \psi | \hat{c}_n^\dagger \hat{c}_n | \psi \rangle - \langle \psi | \hat{c}_m^\dagger \hat{c}_n | \psi \rangle \langle \psi | \hat{c}_n^\dagger \hat{c}_m | \psi \rangle \tag{5.38}$$

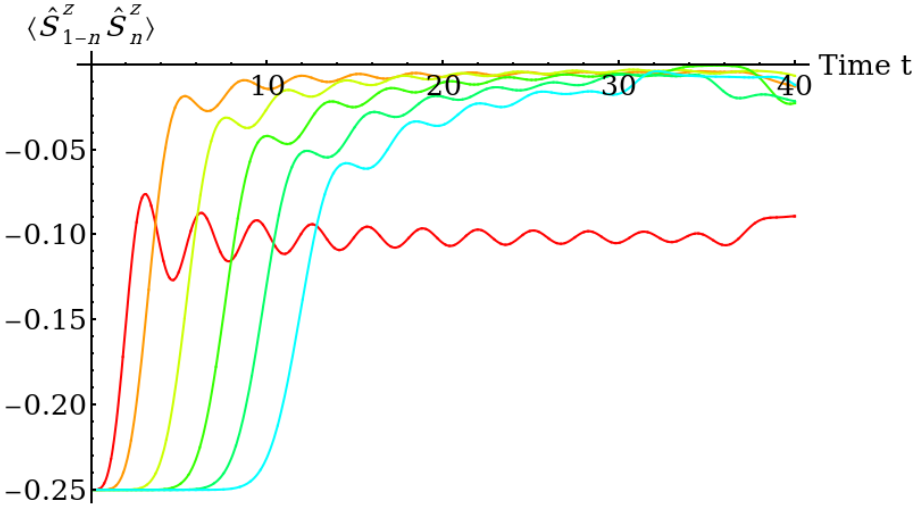

Figure 5.2: The analytical solution for various expectation values for the correlation, $\langle \hat{S}^z_{1-n} \hat{S}^z_n \rangle$. Show is $n = 1$, 3, 5, 7, 9 and 11 from left to right on a chain with length $N = 40$. The boundary effect comes into play again from roughly $t = 30$. It is also clearly visible that the solution for $n = 1$ does not suffer from boundary effects until at least $t = 34$. This is exactly what one would expect, since the sites that $\hat{S}^z_0 \hat{S}^z_1$ works on is further from the boundaries than the sites that, for example, $\hat{S}^z_{-10} \hat{S}^z_{11}$ works on.

We can then use (5.32) again to calculate the expectation value for the correlation:

$$\begin{aligned}
\langle \psi | \hat{S}^z_m \hat{S}^z_n | \psi \rangle (t) = \\
&\langle \Psi_m | \Psi_m \rangle \langle \Psi_n | \Psi_n \rangle - \langle \Psi_m | \Psi_n \rangle \langle \Psi_n | \Psi_m \rangle - \\
&\quad \frac{1}{2} \langle \Psi_m | \Psi_m \rangle - \frac{1}{2} \langle \Psi_n | \Psi_n \rangle + \frac{1}{4} \\
&= \left(\sum_{i=-N/2+1}^{0} |A_{n,i}(t)|^2 \right) \left(\sum_{j=-N/2+1}^{0} |A_{m,j}(t)|^2 \right) - \\
&\quad \left| \sum_{i=-N/2+1}^{0} A^\star_{m,i}(t) A_{n,i}(t) \right|^2 - \\
&\quad \frac{1}{2} \sum_{i=-N/2+1}^{0} |A_{m,i}(t)|^2 - \frac{1}{2} \sum_{i=-N/2+1}^{0} |A_{n,i}(t)|^2 + \frac{1}{4} \quad , \qquad (5.39)
\end{aligned}$$

the result of which has been plotted for a chain of length $N = 40$ in figure 5.2.

Current density $\hat{j}_n$

The current density as given in (4.67) needs to be converted using the Jordan-Wigner transformation as well. Recall that both J and a have been set to

1.

$$\begin{aligned}
\hat{j}_n &= \left(-\frac{i}{2}\hat{S}^+_{n-1}\hat{S}^-_n\right) + \text{h.c.} \\
&= \left(-\frac{i}{2}\hat{c}^\dagger_{n-1}(-1)^{\phi_{n-1}}(-1)^{\phi_n}\hat{c}_n\right) + \text{h.c.} \\
&= \left(-\frac{i}{2}\hat{c}^\dagger_{n-1}(-1)^{\sum_{i=-N/2+1}^{n-2}\hat{c}^\dagger_i\hat{c}_i}(-1)^{\sum_{i=-N/2+1}^{n-1}\hat{c}^\dagger_i\hat{c}_i}\hat{c}_n\right) + \text{h.c.} \\
&= \left(-\frac{i}{2}\hat{c}^\dagger_{n-1}(-1)^{\hat{c}^\dagger_{n-1}\hat{c}_{n-1}}\hat{c}_n\right) + \text{h.c.} \qquad (5.40)
\end{aligned}$$

Since $\left[\hat{c}^\dagger_{n-1}\hat{c}_{n-1}, \hat{c}_n\right] = 0$, we can just swap the last two terms.

$$\hat{j}_n = \left(-\frac{i}{2}\hat{c}^\dagger_{n-1}\hat{c}_n(-1)^{\hat{c}^\dagger_{n-1}\hat{c}_{n-1}}\right) + \text{h.c.}$$

The factor $(-1)^{\hat{c}^\dagger_{n-1}\hat{c}_{n-1}}$ can be left out, since the only non-zero contribution comes from states $|\sigma_{-N/2+1}, \sigma_{-N/2}, \ldots, \sigma_{N/2-1}, \sigma_{N/2}\rangle$ with $\sigma_{n-1} = \downarrow$ and $\sigma_n = \uparrow$. For any other state the operator $\hat{c}^\dagger_{n-1}\hat{c}_n$ will give zero. Thus, in the non-zero case we have

$$\begin{aligned}
&(-1)^{\hat{c}^\dagger_{n-1}\hat{c}_{n-1}}\,|\sigma_{-N/2+1}, \ldots, \sigma_{n-2}, \downarrow, \uparrow, \sigma_{n+1}, \ldots, \sigma_{N/2}\rangle \\
&\quad = (-1)^0\,|\sigma_{-N/2+1}, \ldots, \sigma_{n-2}, \downarrow, \uparrow, \sigma_{n+1}, \ldots, \sigma_{N/2}\rangle \\
&\quad = |\sigma_{-N/2+1}, \ldots, \sigma_{n-2}, \downarrow, \uparrow, \sigma_{n+1}, \ldots, \sigma_{N/2}\rangle \qquad (5.41)
\end{aligned}$$

We can thus just drop the whole $(-1)^{\hat{c}^\dagger_{n-1}\hat{c}_{n-1}}$ altogether.

$$\hat{j}_n = \left(-\frac{i}{2}\hat{c}^\dagger_{n-1}\hat{c}_n\right) + \text{h.c.} \qquad (5.42)$$

Calculating the expectation value for our initial state:

$$\begin{aligned}
\langle\psi|\hat{j}_n|\psi\rangle(t) &= \left(-\frac{i}{2}\langle\psi|\hat{c}^\dagger_{n-1}\hat{c}_n|\psi\rangle\right) + \text{h.c.} \\
&= \left(-\frac{i}{2}\langle\Psi_{n-1}|\Psi_n\rangle\right) + \text{h.c.} \\
&= 2\,\text{Im}\left(\frac{1}{2}\langle\Psi_{n-1}|\Psi_n\rangle\right) \\
&= \text{Im}\left(\sum_{i=-N/2+1}^{0} A^\star_{n-1,i}(t)A_{n,i}(t)\right) \qquad (5.43)
\end{aligned}$$

This equation has been plotted for a spin chain of length $N = 40$ in figure 5.3.

5.1.3 Parameters

There are several parameters that can influence the accuracy of the simulations:

- Time step Δt.

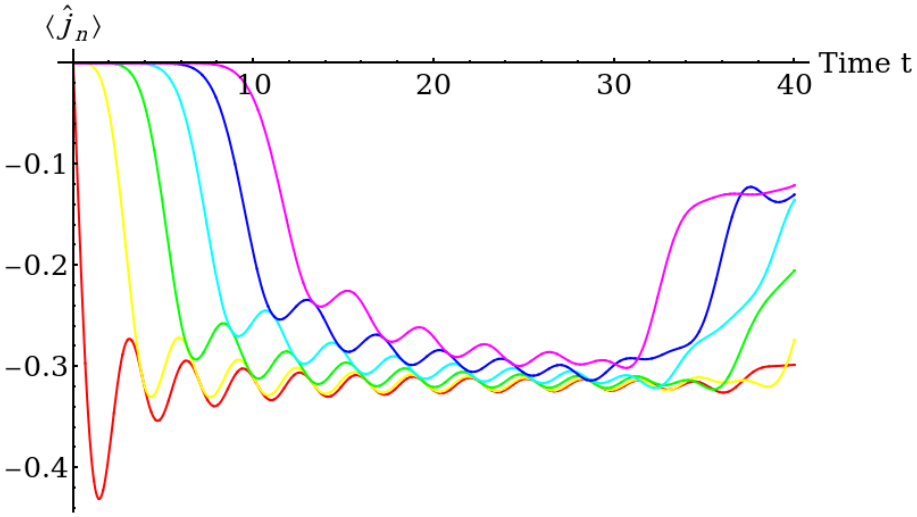

Figure 5.3: The various expectation values for the current density, $\langle \hat{j}_n \rangle$, according to the analytical solution. Shown is $n = 1$, 3, 5, 7, 9 and 11 from left to right on a chain with length $N = 40$. The current for $n = -1$ is the same as for $n = 3$ and the one for $n = -3$ is the same as for $n = 5$, etc.. The boundary effect is visible from $t = 30$.

- Chain length N.
- Total physical time.
- Order of Trotter step.
- Bond dimension d.
- Number of iterative compressions using one matrix.
- Number of iterative compressions using two matrices.

Ideally, the time step should approach zero, but this would cause the computation time required for a simulation to approach infinity. The accuracy depends on the order of the Trotter step, but in the case of a second order Trotter step, the inaccuracy due to a single time step will be $\mathcal{O}(\Delta t^3)$. Of course, the smaller Δt is, the more time steps we will need. Since the number of time steps scales as $1/\Delta t$, we will in effect have $\mathcal{O}(\Delta t^2)$. For these simulations a time step of $\Delta t = 0.005$ (in units of $\hbar$) was chosen, giving an inaccuracy of $\mathcal{O}(10^{-5})$, which will do just fine for our purposes.

The influence of the chain length comes from the interaction with the boundaries. These boundary effects have the potential to interfere with the comparison between the Schrödinger and Heisenberg picture, because, for example we will want to see later on how the correlation operators hold up in the Heisenberg picture as we increase the distance between the two sites. We will therefore aim to minimize the boundary effect. The effect of the boundary could already be seen in the graphs for the analytical results (in figures 5.1, 5.2 and 5.3). To determine when the boundary is interfering, we need to look at the infinite-length

chain. Fortunately there exists an analytical solution for that too. For the S_i^z operator the expectation value is given as[22]:

$$\langle \hat{S}_n^z \rangle(t) = -\frac{1}{2} \sum_{j=1-n}^{n-1} J_j^2(t) \tag{5.44}$$

$J_j(t)$ is the Bessel function of the first kind. Using this formula and the analytical solution from (5.36), it is possible to see how long a simulation can run, while remaining free of boundary effects. This is what is plotted in figure 5.4, where a graph is displayed showing runtime against chain length. The runtime is defined as the physical time that it takes for the finite case to differ from the infinite case by more then 10^{-8}. Since a physical time of 30 will suffice, we will

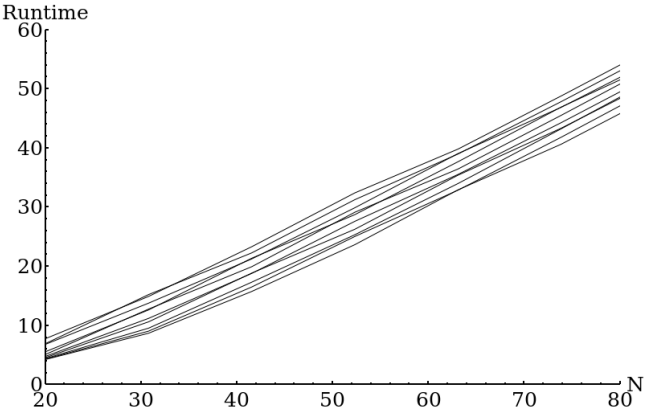

Figure 5.4: Comparison between the analytical solution of $\langle \hat{S}_n^z \rangle$ for a finite length chain, (5.36), and for the infinite-length chain, (5.44). On the x-axis the chain length is plotted and on the y-axis the runtime is plotted. Runtime is defined as the maximum physical time for which the difference between the finite and infinite case is still smaller than 10^{-8}. Plotted are sites $n = 1$ (uppermost line) to $n = 10$ (lowest line). Note that this shows that the boundary effects propagate through the system with a speed of roughly 1.4 ± 0.1 sites per physical time unit.

keep the length of the chain at $N = 64$, which gives a maximum physical time of 43.3 for $n = 1$ and 34.8 for $n = 11$. This is only a lower boundary though, since in the actual simulation we will be comparing the finite case model directly to the results of the simulation. How much the boundary effects will actually interfere with the comparison between the Schrödinger picture and the Heisenberg picture is difficult to specify.

All of the errors coming from the Trotter steps will also invariably add up, limiting the total physical time that the simulation can run accurately. However, the deviation between the simulation and the analytical solutions, resulting from the compression algorithm will be rather dramatic. The effect of the compression will completely drown out the noise from the Trotter steps, except of course in the rare case that either the MPO or MPS is an exact solution, where the compression will be without effect. The total physical time is in that case only limited by boundary effects and bond dimensions. As mentioned in section 4.4, the order of the Trotter step determines the inaccuracies coming from the size of

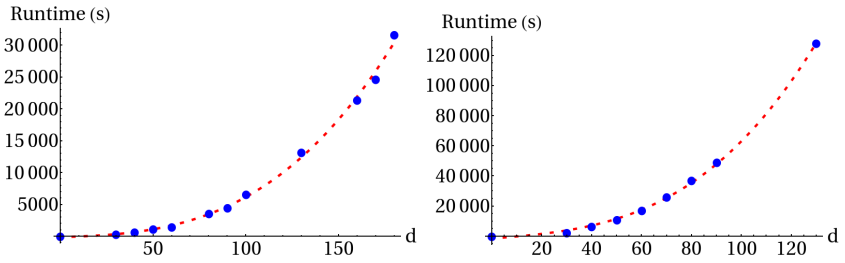

Figure 5.5: Computation time per physical time unit as function of d. The red dotted line is a fit with a 3rd order polynomial, indicating that the simulation is $\mathcal{O}(d^3)$. This is in the Schrödinger picture with a chain of length $N = 64$. Left is in the Schrödinger picture and right is in the Heisenberg picture.

the time step. It ranges from $\mathcal{O}(\Delta t^2)$ for a first order Trotter step to $\mathcal{O}(\Delta t^5)$ for a Forest-Ruth Trotter step. Of course the order of the Trotter step can be increased as much as one likes at the cost of many more MPO multiplications. For these simulations, we will stick with second order Trotter steps, $\mathcal{O}(\Delta t^3)$, since it gives satisfactory results while not increasing the computation time too much.

The bond dimension is the maximum size for the matrices making up an MPS/MPO as well as the maximum number of singular values per site in an MPS/MPO. It determines how rigorous the compression will be. This bond dimension, d, will be varied throughout the comparisons. The computational time a simulation takes in both the Schrödinger and the Heisenberg picture is plotted in figure 5.5. A fit through the data shows that the simulation is most likely $\mathcal{O}(d^3)$, which is what one would expect, since the bulk of the calculations during a simulation involves the SVD, which is $\mathcal{O}(N^3)$ with N being the dimension of the matrix. It also shows that simulations in the Schrödinger picture are roughly eight times faster than simulations in the Heisenberg picture.

Improving the MPS/MPO after it has been compressed by iteratively approaching the original MPS/MPO, gives a small but still appreciable increase in accuracy for large time steps, such as $\Delta t = 0.05$ and low bond dimension, such as $d = 8$. In the realm where we will be simulating, $\Delta t = 0.005$, $d = 40 \ldots 100$ the effect even becomes negligible, as can be seen in figure 5.6. However using two matrices to iteratively approach the compressed MPS/MPO did not have any effect during the simulations, if the same number of single matrix approaches was applied afterwards. This appears to indicate, that local minima are not a problem. In conclusion, even the small increase in accuracy that the single matrix approaches give, is drowned out by the major effect from the bond dimension. Since the iterative approach is quite a costly operation (SVDs are $\mathcal{O}(mn^2)$ for $m \times n$ matrices, $m \geq n$), the simulation is much faster without it. However, it does have its merits when the SVD algorithm fails, as discussed in section B.2. In that case, the iterative compression can do 'damage control' by repairing the more inaccurate fall-back SVD, allowing the simulation to continue unhindered.

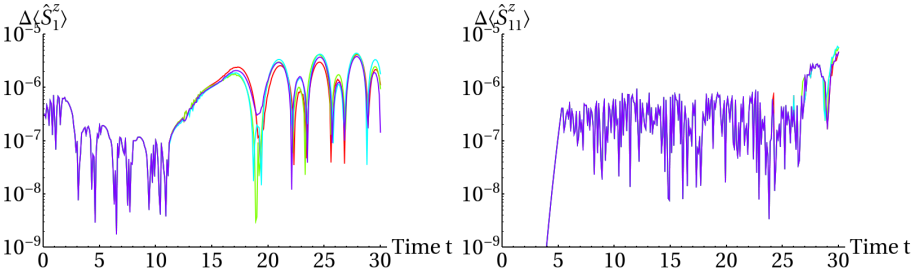

Figure 5.6: The effects of iterative compression. The deviation from the analytical solution of the XX model, starting from the inhomogeneous state, is shown for various levels of iterative compression. Red indicates no iterative compression, green indicates one sweep, blue is two sweeps and finally purple is three sweeps. The left graph is of the spin at site $n = 1$ and the right graph is of the spin at site $n = 11$. Both simulations were run with $\Delta t = 0.005$, $N = 40$ and $d = 60$ in the Schrödinger picture.

5.1.4 Results of comparison

With the analytical result and the effect of all possible parameters known, it is now time to turn our attention to the results of the simulations and to see how the simulations in the Schrödinger picture compare to simulations in the Heisenberg picture.

Density operator

In figure 5.7 the result of simulations in both the Schrödinger and the Heisenberg picture is plotted in one graph at $N = 20$ and $d = 40$. It clearly shows that the simulation in the Heisenberg picture just keeps going, which is exactly the effect described in [17]. The compression algorithm does nothing in this case, since the operator can always be described by an MPO with $d \geq 4$. The accuracy of a simulation depends mainly on the level of compression, so in this case, simulating in the Heisenberg picture will be far more accurate than simulating in the Schrödinger picture. In figure 5.8, the graph is shown for $\langle\hat{S}_1^z\rangle$ and $\langle\hat{S}_{11}^z\rangle$, but this time the evolution is in the XXZ-model.

Current density operator

In figure 5.9, a comparison is shown between a simulation in the Schrödinger picture and in the Heisenberg picture for $\langle\hat{j}_1\rangle$ and, in figure 5.10, for $\langle\hat{j}_{11}\rangle$. One can see that once again the simulation in the Heisenberg picture does not suffer from compression, since changing the bond dimension has no effect.

Correlation operator

For $\langle\hat{S}_0^z\hat{S}_1^z\rangle$, a comparison is shown between the Schrödinger and Heisenberg picture in figure 5.11. Runtimes have been plotted in 5.12. The runtime is defined as the simulated physical time that the simulation stays within 10^{-6} of the analytical result.

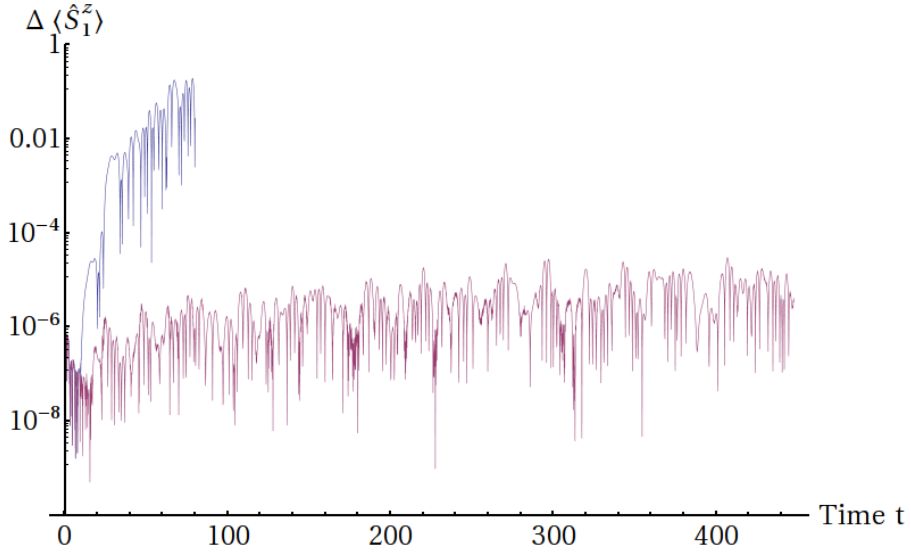

Figure 5.7: Simulations in the XX-model with $N = 20$ in both the Schrödinger picture (blue) and in the Heisenberg picture (purple) with the maximum bond dimension $d = 40$. The y-axis is the difference between simulations and the analytical solution for $\langle \hat{S}_1^z \rangle$.

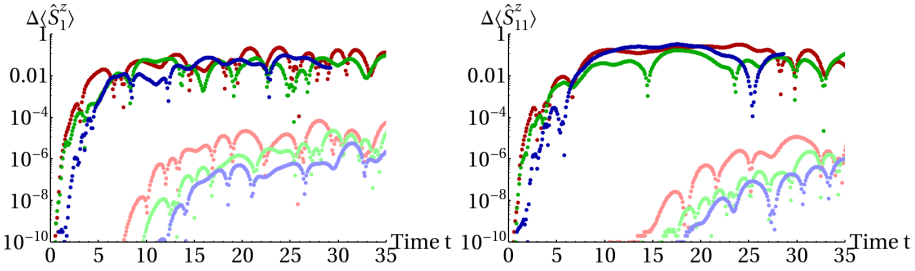

Figure 5.8: Simulations for $\langle \hat{S}_1^z \rangle$ (left) and $\langle \hat{S}_{11}^z \rangle$ (right) in the XXZ-model with $N = 64$ in both the Schrödinger picture (bright colours) and in the Heisenberg picture (dark colours). The y-axis is the difference between a simulation with bond dimension d and a simulation with bond dimension $d + 20$. Red is between $d = 40$ and $d = 60$, green between $d = 60$ and $d = 80$ and, finally, blue is between $d = 80$ and $d = 100$.

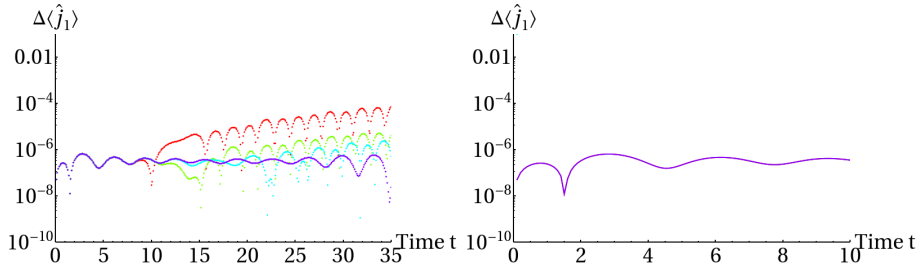

Figure 5.9: Simulations with $N = 64$ in the XX model in both the Schrödinger picture (left graph) and in the Heisenberg picture (right graph) for different values of the maximum bond dimension d: $d = 40$ (red), $d = 60$ (green), $d = 80$ (light blue) an $d = 100$ (purple). The y-axis is the difference between the simulations and analytical solution for $\langle \hat{j}_1 \rangle$. One can see that in the Heisenberg picture the simulation is independent of the bond dimension.

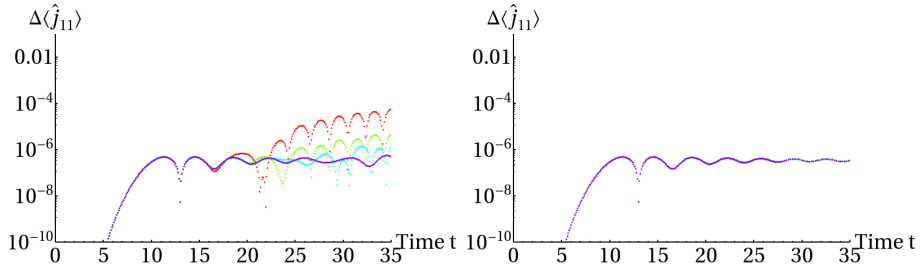

Figure 5.10: Simulations with $N = 64$ in the XX model in both the Schrödinger picture (left graph) and in the Heisenberg picture (right graph) for different values of the maximum bond dimension d: $d = 40$ (red), $d = 60$ (green), $d = 80$ (light blue) an $d = 100$ (purple). The y-axis is the difference between the simulations and analytical solution for $\langle \hat{j}_{11} \rangle$. One can see that in the Heisenberg picture the simulation is independent of the bond dimension.

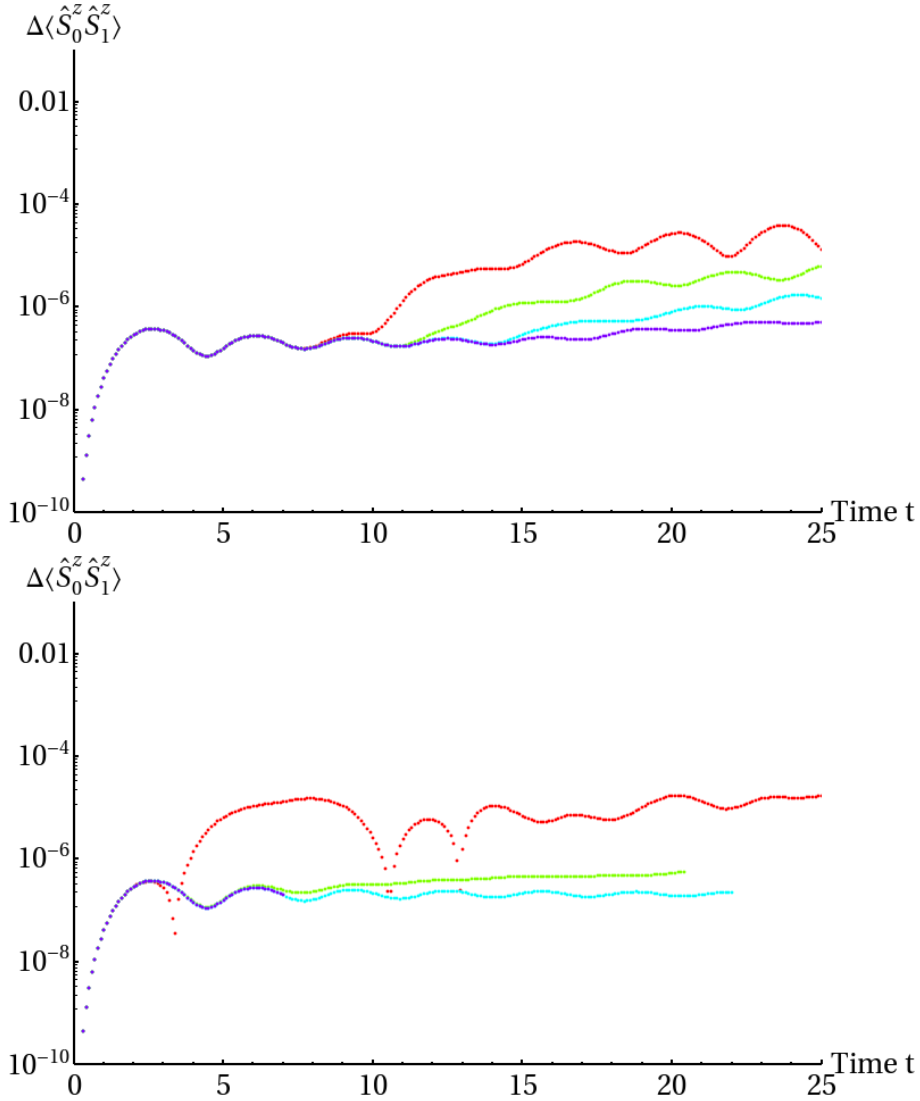

Figure 5.11: Simulations in the XX chain with $N = 64$ in both the Schrödinger picture (upper graph) and in the Heisenberg picture (lower graph) for different values of the maximum bond dimension d: $d = 40$ (red), $d = 60$ (green), $d = 80$ (light blue) an $d = 100$ (dark blue). The y-axis is the difference between the simulations and analytical solution for $\langle\hat{S}_0^z\hat{S}_1^z\rangle$.

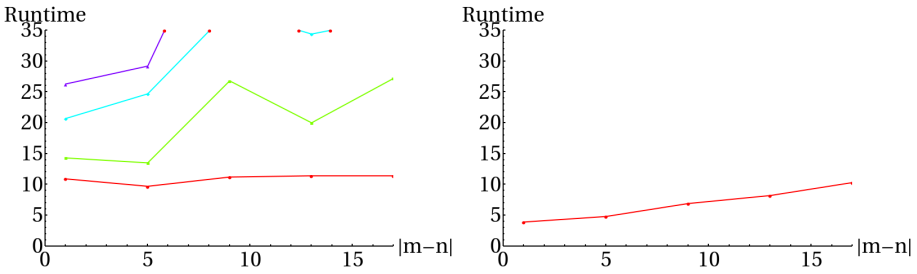

Figure 5.12: Physical time the simulations of $\langle \hat{S}^z_m \hat{S}^z_n \rangle$ in the XX chain stay within 10^{-6} of the analytical solution for increasing $|m-n|$. From bottom to top: $d = 40$, $d = 60$, $d = 80$ and $d = 100$. The left graph is in the Schrödinger picture and the right in the Heisenberg picture. On the right only the case of $d = 40$ is visible. The other cases run for longer than 60 physical time units and behaviour similar to figure 5.7 is expected. That is, the simulation will go on almost indefinitely.

5.1.5 Discussion

All the operators that have been simulated for the XX chain appear to have greater accuracy in the Heisenberg picture after a certain minimal bond dimension is used. In the case of the $\hat{S}^z_n$ operators, the simulation in the Heisenberg picture is already extremely accurate for $d \geq 4$. Even the more complex correlation operator $\hat{S}^z_m \hat{S}^z_n$ reaches a minimal d for which the simulation in the Heisenberg picture becomes extremely accurate, albeit for a higher minimal d which lies between 40 and 60. For the current density $\hat{j}_n$ this minimal d lies even beneath 40, based on [16] this minimal d is probably 16. So, in the case of the XX-chain, the simulation can best be run in the Heisenberg picture, provided that a minimal d is used, which varies per operator. For the XXZ-chain the story is unfortunately very different. In that system, the simulation in the Heisenberg picture fails very quickly even for a local operator. To investigate why this would be the case, a similar simulation was run in the Heisenberg picture that recorded the Frobenius norm of the difference between the MPO before and after compression. As can be seen from figure 5.13 the difference between the MPOs increases exponentially after a certain time, depending on the maximum bond dimension used. This is what causes the simulation in the Heisenberg picture to break down so fast. Attempts to correct for this using the iterative approach from subsection 3.7.3 did help somewhat, but not enough to stave off the large deviations seen in figure 5.8. A closer inspection of the singular values can also be seen in figure 5.13. One can see that at $t = 0$ only one non-zero singular value exists, but that as time progresses, other singular values increase significantly. This is more clearly visible in figure 5.14 where the decay of the singular values is shown as s_i versus i, with s_i being the sorted singular values. Even though there is still an exponential decay of the singular values, it is simply not strong enough, because after every compression, three quarters of the singular values will be thrown away. This is due to the fact that the Trotter MPO has a bond dimension of $d = 4$, so after every multiplication with the MPO that is being evolved in time, the bond dimension of the time evolving MPO will increase fourfold. Coupled with the fact that the singular values rise rapidly, a substantial loss of accuracy occurs. To further examine

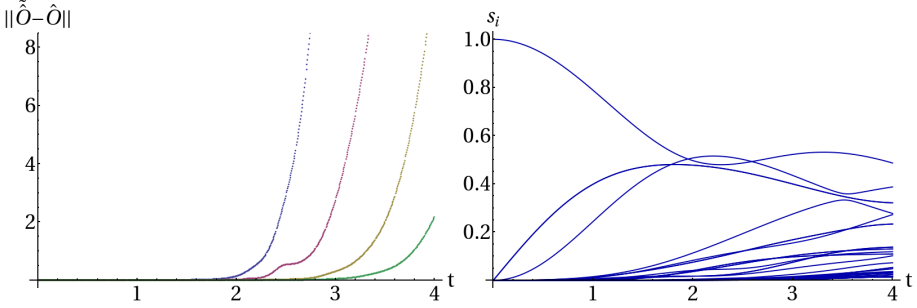

Figure 5.13: The left graph shows the Frobenius norm of the difference between the MPO before and after compression. From left to right, the bond dimension is $d = 40, 60, 80$ and 100. The graph on the right shows the singular values of the matrix of the MPO corresponding to site $n = 1$ before compression. Both simulations were run in the Heisenberg picture with a chain length of $N = 32$, while calculating the expectation value of $\hat{S}_1^z$ in the XXZ chain.

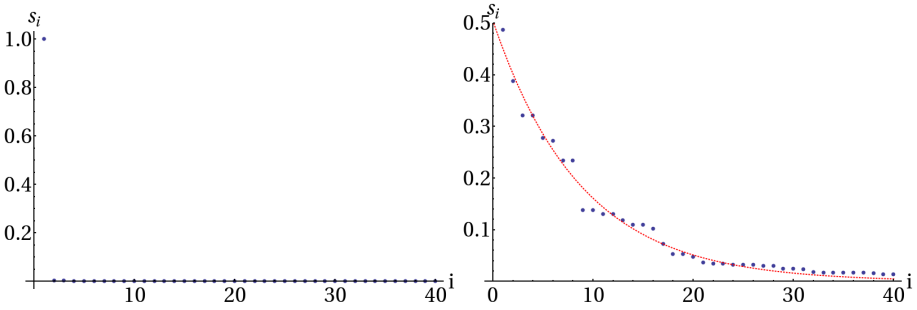

Figure 5.14: The left graph shows the distribution of singular values, s_i, for $t = 0$ and the right graph for $t = 4$. Both graphs are a slice of the right graph in figure 5.13 at $t = 0$ and $t = 4$ respectively. One can see that at $t = 0$ there is only one singular value that really matters, so the compression will not have much influence. However, as time progresses, the singular values become more distributed. Then the compression will throw away singular values that do matter. From the graph at $t = 4$, it is clear that it is still an exponential decay (red dotted line), but the decay is not as strong as it was in the beginning. Both graphs are for the $\hat{S}_1^z$ operator in the XXZ chain.

why this occurs, it is instructive to define something similar to entanglement entropy, but now for an MPO. The exact definition we will use, is given in appendix D. In figure 5.15, a plot is displayed for the 'entanglement entropy' of $\hat{S}_1^z$. One can see that in the XXZ model, the 'entanglement' increases linearly. In the XX model, however, the 'entanglement' becomes constant. Normally, in the case of the Schrödinger picture, entanglement indicates how fast one would need to make the bond dimension grow to keep the same level of accuracy, which goes as $d \propto \exp(\alpha S)$ [23]. Assuming we can make a similar statement in the case of MPOs, then one can see that linear behaviour in the entropy means and exponential increase in required bond dimension. Since we keep the bond dimension constant in our simulations, one can see that the accuracy will drop off quickly, as indeed it does.

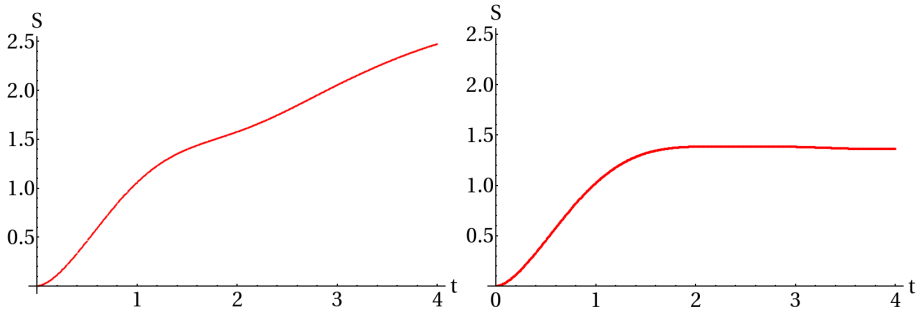

Figure 5.15: The 'entanglement entropy' for the operator $\hat{S}_1^z$ in the XXZ-chain (left) and the XX-chain (right).

5.2 The XXZ chain quench

Another system that may yield interesting results, is the quench of an XXZ chain by suddenly increasing $J_z = 0$ to $J_z = 1$. Recall from (4.2) the Hamiltonian,

$$\hat{H} = -\frac{J}{2}\sum_i \left(\hat{S}_i^+\hat{S}_{i+1}^- + \hat{S}_{i+1}^+\hat{S}_i^-\right) + J_z\sum_i \hat{S}_i^z\hat{S}_{i+1}^z \quad . \tag{5.45}$$

The initial state we shall use, is thus the ground state of the XX chain, since this is also the ground state for the XXZ chain with $J_z = 0$ and it has the bonus that this ground state can easily be constructed analytically. The time evolution for the XXZ chain is the same as in subsection 5.1.1.

5.2.1 Initial state

The ground state of the spin-$\frac{1}{2}$ XX chain can be calculated analytically. For this, we need to apply all of the $\hat{d}_k^\dagger$ operators from (5.21) that contribute a negative energy, i.e. all k for which $\epsilon_k < 0$. Since the matrix T was Hermitian, this amounts to the first half of all the eigenvalues, assuming they are sorted from lowest to highest. The ground state can then be written as,

$$|\mathrm{GS}\rangle = \left(\prod_{k=1}^{N/2}\hat{d}_k^\dagger\right)|\downarrow\rangle = \left(\prod_{k=1}^{N/2}\sum_{i=-N/2+1}^{N/2}\hat{c}_i^\dagger U_{i,k}\right)|\downarrow\rangle \quad . \tag{5.46}$$

There is no easy way to write this in MPS form. However, it is possible to convert the sum over the creation operators to an MPO and apply that repeatedly to $|\downarrow\rangle$. In MPS form $|\downarrow\rangle$ can be written as,

$$\begin{aligned} A^{\uparrow_{-N/2+1}} &= \begin{pmatrix} 0 \end{pmatrix} \quad A^{\uparrow_i} = \begin{pmatrix} 0 \end{pmatrix} \quad A^{\uparrow_{N/2}} = \begin{pmatrix} 0 \end{pmatrix} \\ A^{\downarrow_{-N/2+1}} &= \begin{pmatrix} 1 \end{pmatrix} \quad A^{\downarrow_i} = \begin{pmatrix} 1 \end{pmatrix} \quad A^{\downarrow_{N/2}} = \begin{pmatrix} 1 \end{pmatrix} \end{aligned} \quad . \tag{5.47}$$

The MPO representing $\hat{O}_k = \sum_{i=-N/2+1}^{N/2}\hat{c}_i^\dagger U_{i,k}$ is more difficult to write down, since the fermionic operator $\hat{c}_i^\dagger$ is not really a local site operator, because its sign depends on the spin of all of the previous sites. This can be solved by reversing the Jordan-Wigner transformation from (5.12):

$$\hat{c}_i^\dagger = \hat{S}_i^+(-1)^{\sum_j^{i-1}\left(\frac{1}{2}+\hat{S}_j^z\right)} \tag{5.48}$$

This is a product of local site operators and can therefore be written as an MPO. Actually, we are interested only in the whole sum, expressed in the $\hat{S}$ operators:

$$\hat{O}_k = \sum_{i=-N/2+1}^{N/2} \hat{S}_i^+ (-1)^{\sum_j^{i-1}\left(\frac{1}{2}+\hat{S}_j^z\right)} U_{i,k} \tag{5.49}$$

After a bit of puzzling, one can write this in an MPO form with a bond dimension of $d = 2$:

$$\hat{O}_k = \begin{cases} B_k^{[-N/2+1]} &= \begin{pmatrix} U_{-N/2+1,k}\hat{S}^+ & (-1)^{\frac{1}{2}+\hat{S}^z} \end{pmatrix} \\ B_k^{[i]} &= \begin{pmatrix} 1 & 0 \\ U_{i,k}\hat{S}^+ & (-1)^{\frac{1}{2}+\hat{S}^z} \end{pmatrix} \\ B_k^{[N/2]} &= \begin{pmatrix} 1 \\ U_{N/2,k}\hat{S}^+ \end{pmatrix} \end{cases} \tag{5.50}$$

The index k is the same k as used in the product in (5.46). Now that everything from (5.46) is either an MPO or an MPS, we can write down the ground state:

$$|\mathrm{GS}\rangle = \hat{O}_1 \cdot \ldots \cdot \hat{O}_{N/2} \cdot |\downarrow\rangle \tag{5.51}$$

This would however make the bond dimension very large, namely $d = 2^{N/2}$, because we need to apply an MPO with bond dimension $d = 2$ for $N/2$ times. Therefore, we will need to apply compression after every multiplication.

5.2.2 Operators

Since every $\hat{O}_i$ is a sum of creation operators, applying $N/2$ of them will make the ground state consist of all of the basis states that have $N/2$ spin $\uparrow$ and $N/2$ spin $\downarrow$. Furthermore, since the Hamiltonian of the XX-chain, 4.1, is symmetric under a spin flip of the whole system, for every basis state in the ground state there is a spin flipped basis state with the same amplitude. The consequence of this is that the expectation value of the $\hat{S}_n^z$ operator will always be zero, because for every contribution to the expectation value of one basis state, there is an inverse contribution by the spin-flipped version of that basis state. Since the expectation value of the spin operator is always zero for every site, the current density, $\hat{j}_n$, will also be zero for every site, because it depends on $\frac{d\hat{S}^z}{dt}$ through the equation of continuity. The correlation operator, $\hat{S}_m^z \hat{S}_n^z$, however, is an operator that can have a non-zero expectation value. The correlation will therefore be the only operator that we will study. To examine the behaviour of the simulation in both the Schrödinger and the Heisenberg picture, we will restrict ourselves to the expectation values $\langle \hat{S}_0^z \hat{S}_1^z \rangle$, $\langle \hat{S}_{-2}^z \hat{S}_3^z \rangle$, $\langle \hat{S}_{-4}^z \hat{S}_5^z \rangle$, $\langle \hat{S}_{-6}^z \hat{S}_7^z \rangle$ and $\langle \hat{S}_{-8}^z \hat{S}_9^z \rangle$.

5.2.3 Results

The results of $\langle \hat{S}_0^z \hat{S}_1^z \rangle$ can be seen in figure 5.16. It shows clearly that, in this case, the simulation in the Schrödinger picture is better than the simulation in the Heisenberg picture. In figure 5.17, the runtimes are plotted as a function of $|i - j|$ for all of the operators $\hat{S}_i^z \hat{S}_j^z$, where runtime is defined as the physical

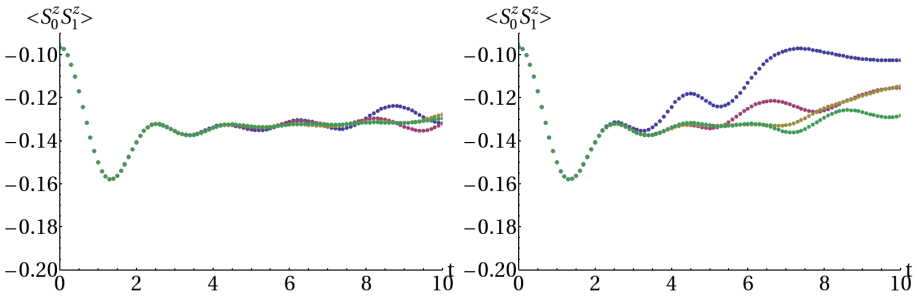

Figure 5.16: Simulation of the expectation value $\langle \hat{S}_0^z \hat{S}_1^z \rangle$ on a chain of length $N = 64$ in both the Schrödinger (left) and the Heisenberg picture (right) for various bond dimensions: $d = 40$ (blue), $d = 60$ (purple), $d = 80$ (brown) and $d = 100$ (green).

time that the simulations stay within either 10^{-4} or 10^{-6} of the simulation with a higher bond dimension. Of course this does not portray the 'real' accuracy, as it would with a comparison with an analytical solutions, since the simulation with a higher bond dimension already has some inaccuracy in itself. However, without an analytical solution available, this is the next best option. Also, it is clear from figure 5.16 how the simulations in both pictures behave. One can see that the accuracy of either picture varies with the distance $|i - j|$ and even with the tolerance. In this case, the simulation in the Schrödinger picture would still be better because it requires far less computational resources than the simulation in the Heisenberg picture. A plot was also made of the 'entanglement entropy' in the Heisenberg picture as discussed in 5.1.5 and more detailed in appendix D. This plot can be seen in figure 5.18. It shows that the entanglement increases linearly. After a while, it becomes constant at roughly $S = 3.5$, but this is purely due to the upper limit created by the compression algorithm as described in appendix D. We used a bond dimension of $d = 40$, so the upper limit would be $S = \ln 40 = 3.69$. The graph also shows that the further m and n are apart in $\hat{S}_m^z \hat{S}_n^z$, the longer the entropy stays low. This explains why the runtime of the simulations in the Heisenberg picture increases in the left graph of figure 5.17. However, the obtained accuracy is similar to the accuracy in the simulations in the Schrödinger picture. Thus, the few cases where the simulations in the Heisenberg picture performed better then the simulations in the Schrödinger picture, certainly do not weigh up to the fact that the simulations in the Heisenberg picture take roughly a factor sixteen longer than the same simulations in the Schrödinger picture.

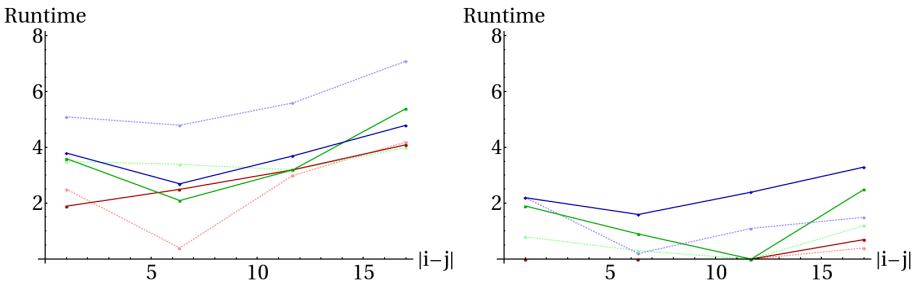

Figure 5.17: Runtime as a function of $|i-j|$ of the expectation value $\langle \hat{S}_i^z \hat{S}_j^z \rangle$. Runtime is defined as the physical time the simulation with bond dimension d stays within either 10^{-4} (left) or 10^{-6} (right) of a simulation with a bond dimension of $d+20$. The dotted bright lines correspond to simulations in the Schrödinger picture and the dark solid lines to simulation in the Heisenberg picture. Red is for $d = 40$, green for $d = 60$ and blue for $d = 80$.

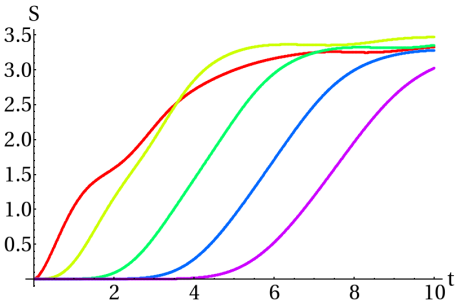

Figure 5.18: The 'entanglement entropy' as a function of time if the XXZ chain in the Heisenberg picture with bond dimension $d = 40$. From left to right, the 'entropy' of the operators $\hat{S}_0^z \hat{S}_1^z$, $\hat{S}_{-2}^z \hat{S}_3^z$, $\hat{S}_{-4}^z \hat{S}_5^z$, $\hat{S}_{-6}^z \hat{S}_7^z$ and $\hat{S}_{-8}^z \hat{S}_9^z$ are shown. It shows that the further m and n are apart in $\hat{S}_m^z \hat{S}_n^z$, the longer it takes for the entropy to increase. Note that the effect of the upper limit of $S = 3.68$, which was calculated in appendix D.

5.3 XX chain with excitation on ground state, $\hat{S}_1^+|\mathrm{GS}\rangle$

Another interesting case is that of the XX chain with a disturbance of the ground state. One starts with the ground state of the XX chain and one applies the spin raising operator on the site $n = 1$. The system will be kicked out of its ground state and the ensuing dynamics can be followed.

5.3.1 Initial state

For the initial state, one can use (5.51) again, but this time apply the MPO from (4.4) with $\hat{S}_1^+$.

$$|\Psi\rangle = \hat{S}_1^+ \hat{O}_1 \cdot \ldots \cdot \hat{O}_{N/2} \cdot |\downarrow\rangle \tag{5.52}$$

The state still needs to be normalized to arrive at the actual excited state.

$$|\mathrm{ES}\rangle = \frac{1}{\|\Psi\|}|\Psi\rangle \tag{5.53}$$

5.3.2 Operator

We will try to calculate the expectation value,

$$\langle \mathrm{GS}|\hat{S}_n^-(t)\hat{S}_1^+(0)|\mathrm{GS}\rangle = \langle \mathrm{GS}|\hat{S}_n^-(t)|\mathrm{ES}\rangle \quad . \tag{5.54}$$

This expectation value normally can be used to calculate the spectrum after a Fourier transform in both space and time, but here a disadvantage of the Heisenberg picture comes in, which is that only one operator can be simulated per simulation. To get the spectrum, one would have to run the simulation for all N sites. However, since we are only interested in how the DMRG in the Schrödinger picture compares to the Heisenberg picture, we will only examine a few sites, namely $n \in \{-10, 0, 1, 10\}$.

5.3.3 Results

In figure 5.19, the expectation value $\langle \mathrm{GS}|\hat{S}_{11}^-(t)|\mathrm{ES}\rangle$ has been plotted. In figure 5.20 one can see a plot of the runtime. Runtime is the physical time that a simulation with bond dimension d stays within 10^{-4} of a simulation with bond dimension $d + 20$. From both figures, it is clear that the simulation performs far better in the Schrödinger picture than it does in the Heisenberg picture. This is rather unexpected, since the time evolution is still that of the XX chain. Based on the results of section 5.1, one might expect that there is a certain bond dimension, d, for which the Heisenberg picture would become exact. Especially since $\hat{S}_n^-$ is a local operator that is being evolved in time, similar to the case of $\hat{S}^z$. However this turns out not to be the case for $\hat{S}^\pm$. See appendix C for the detailed calculation. In short, the Jordan-Wigner transformation introduces a troublesome factor which cancels out in the case of $\hat{S}^z$, but unfortunately does not for $\hat{S}^\pm$. Figure 5.21 show the 'entanglement entropy' of the operator as defined in appendix D.

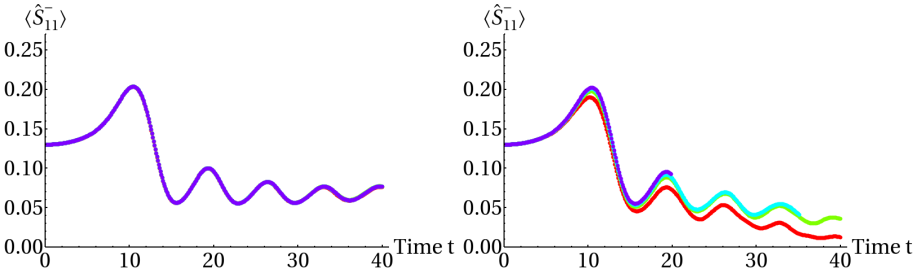

Figure 5.19: The time evolution of the expectation value $\langle \mathrm{GS}|\hat{S}^-_{11}(t)|\mathrm{ES}\rangle$ in the Schrödinger picture (left) and in the Heisenberg picture (right). Red indicates bond dimension $d = 40$, green for $d = 60$, cyan for $d = 80$ and purple for $d = 100$. It is clear that the simulation in the Heisenberg picture performs much worse than in the Schrödinger picture.

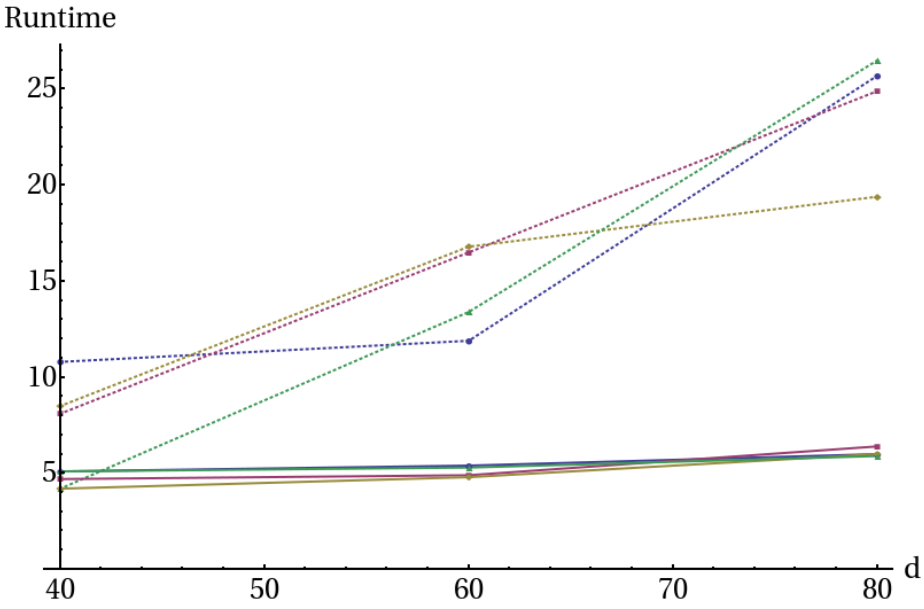

Figure 5.20: Runtime versus bond dimension. Runtime is defined as the physical time that a simulation with bond dimension d stays within 10^{-4} of a simulation with bond dimension $d + 20$. The dotted lines are from the simulation in the Schrödinger picture, while the solid lines are from the simulation in the Heisenberg picture. This is for several expectation values, $\langle \mathrm{GS}|\hat{S}^-_n(t)|\mathrm{ES}\rangle$, for $n \in \{-10, 0, 1, 11\}$.

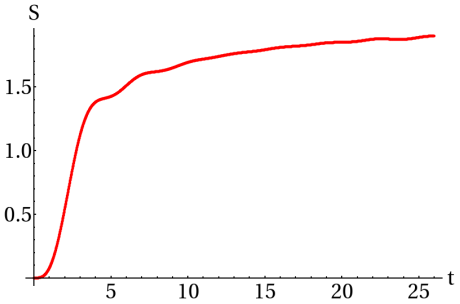

Figure 5.21: The 'entanglement entropy' of the $\hat{S}^-$ operator in the XX chain of length $N = 64$.

5.4 Bose-Hubbard Model

A completely different system to consider, is the Bose-Hubbard model, a system where the DMRG normally fails after simulating only a few physical time units. The Hamiltonian for this system is:

$$\hat{H} = -t \sum_i \left(\hat{b}_i^\dagger \hat{b}_{i+1} + \hat{b}_{i+1}^\dagger \hat{b}_i \right) - \frac{U}{2} \sum_i \hat{n}_i \left(\hat{n}_i - 1 \right) - \mu \sum_i \hat{n}_i \tag{5.55}$$

Here, $\hat{b}_i$ is the bosonic annihilation operator and $\hat{n}_i$ is the number operator $\hat{b}_i^\dagger \hat{b}_i$. As the initial state we will use $|0, 1, 0, 1, \ldots\rangle$. Only in certain cases does there exist an analytical solution for this system, for example for $U = \mu = 1$, in which case the method of solving the system is essentially the same as for the XX-chain.

5.4.1 Initial state

Writing down the initial state $|0, 1, 0, 1, \ldots\rangle$ as an MPS is not as straightforward as it was for the spin-$\frac{1}{2}$ particles, since there is now an infinite number of basis states, $|\boldsymbol{\sigma}\rangle$. One needs to limit how many bosons are allowed at a single site, to get a finite number of basis states.

$$\begin{array}{ll} A^{0_{\text{odd}}} = \begin{pmatrix} 1 \end{pmatrix} & A^{0_{\text{even}}} = \begin{pmatrix} 0 \end{pmatrix} \\ A^{1_{\text{odd}}} = \begin{pmatrix} 0 \end{pmatrix} & A^{1_{\text{even}}} = \begin{pmatrix} 1 \end{pmatrix} \\ A^{\sigma_{\text{odd}}} = \begin{pmatrix} 0 \end{pmatrix} & A^{\sigma_{\text{even}}} = \begin{pmatrix} 0 \end{pmatrix} \quad \text{For } \sigma \geq 2 \end{array} \tag{5.56}$$

5.4.2 Time evolution

To create the Trotter steps, we need a local interaction operator again[1]:

$$\hat{h}_i = -t \left(\hat{b}_i^\dagger \hat{b}_{i+1} + \hat{b}_{i+1}^\dagger \hat{b}_i \right) - \frac{U}{2} \hat{n}_i \left(\hat{n}_i - 1 \right) - \mu \hat{n}_i \tag{5.57}$$

The last site needs some special treatment, since we need to account for $\hat{n}_{N/2}$.

$$\begin{aligned} \hat{h}_{N/2-1} = &-t \left(\hat{b}_{N/2-1}^\dagger \hat{b}_{N/2} + \hat{b}_{N/2}^\dagger \hat{b}_{N/2-1} \right) - \frac{U}{2} \hat{n}_{N/2-1} \left(\hat{n}_{N/2-1} - 1 \right) \\ &- \mu \hat{n}_{N/2-1} - \frac{U}{2} \hat{n}_{N/2} \left(\hat{n}_{N/2} - 1 \right) - \mu \hat{n}_{N/2} \end{aligned} \tag{5.58}$$

The size of the resulting matrix depends on the maximum number of bosons we will allow per site during the simulation. In the case of at most 2 bosons, the matrix will already be 9×9. To wit, using the basis $\{ |0,0\rangle, |0,1\rangle, |0,2\rangle, |1,0\rangle,$

[1] There are other possibilities. One could for example distribute the $\hat{n}_i$ evenly, in which case one would get $\hat{h}_i = -t \left(\hat{b}_i^\dagger \hat{b}_{i+1} + \hat{b}_{i+1}^\dagger \hat{b}_i \right) - \frac{1}{2} \left(\frac{U}{2} \hat{n}_i \left(\hat{n}_i - 1 \right) - \mu \hat{n}_i \right) - \frac{1}{2} \left(\frac{U}{2} \hat{n}_{i+1} \left(\hat{n}_{i+1} - 1 \right) - \mu \hat{n}_{i+1} \right)$, but then one would have to deal with both $h_{-N/2+1}$ and $h_{N/2}$ separately.

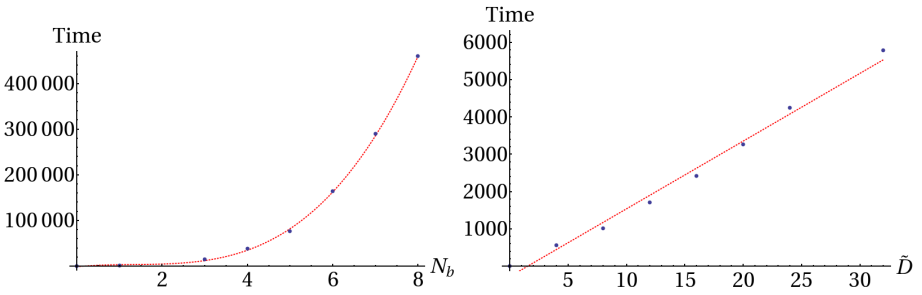

Figure 5.22: Computation time in seconds versus the maximum number of bosons per site (left) and versus the bond dimension of the Trotter MPO, $\tilde{D}$ (right). The red dotted line in the left graph is a fit with a third degree polynomial, suggesting that the algorithm is $\mathcal{O}(N_b^3)$ in the number of bosons per site. In the right graph a fit was made with a first degree polynomial, which shows that the algorithm is $\mathcal{O}(\tilde{D})$ in the bond dimension of the Trotter MPO.

$|1,1\rangle$, $|1,2\rangle$, $|2,0\rangle$, $|2,1\rangle$, $|2,2\rangle$ }, we have:

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -t & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -\sqrt{2}t & 0 & 0 & 0 & 0 \\ 0 & -t & 0 & -\mu & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -\sqrt{2}t & 0 & -\mu & 0 & -\sqrt{2}t & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -\mu & 0 & -2t & 0 \\ 0 & 0 & 0 & 0 & -\sqrt{2}t & 0 & -U-2\mu & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -2t & 0 & -U-2\mu & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -U-2\mu \end{bmatrix} \tag{5.59}$$

This can be entered in (4.42) to get the time evolution MPO. The maximum number of bosons per site, N_b, can play an important role during the simulation. In figure 5.22, the computation time is plotted against the maximum number of bosons per site. From this graph, it can be concluded that our algorithm is $\mathcal{O}(N_b^3)$. It turns out that even for the chosen initial state with only at most one boson per site, the simulation will not be accurate until at least seven bosons per site are allowed, and even then only up to three physical time units. This is evident from figure 5.23, where simulations are shown for various values of N_b. However, this will lead to a rather large matrix of 64×64. Fortunately, for the case of $t = 1, U = 0, \mu = 0$, a singular value decomposition will result in eight singular values that are zero. We can therefore compress the matrix down to 56×56 without any ill effects. This should already provide a speed-up of 12.5%, since the program is $\mathcal{O}(\tilde{D})$ in the bond dimension of the Trotter MPO, $\tilde{D}$, according to figure 5.22. Further compression is of course desirable and for that reason, several simulations were run at $d = 100$ and $N = 32$ with varying compression of the Trotter MPO. The results of these can be seen in figure 5.24. One can see that a $\tilde{D}$ of 16 works well. Higher $\tilde{D}$ are only drowned out by the inaccuracies arising from the fact that we only allow at most seven bosons per site.

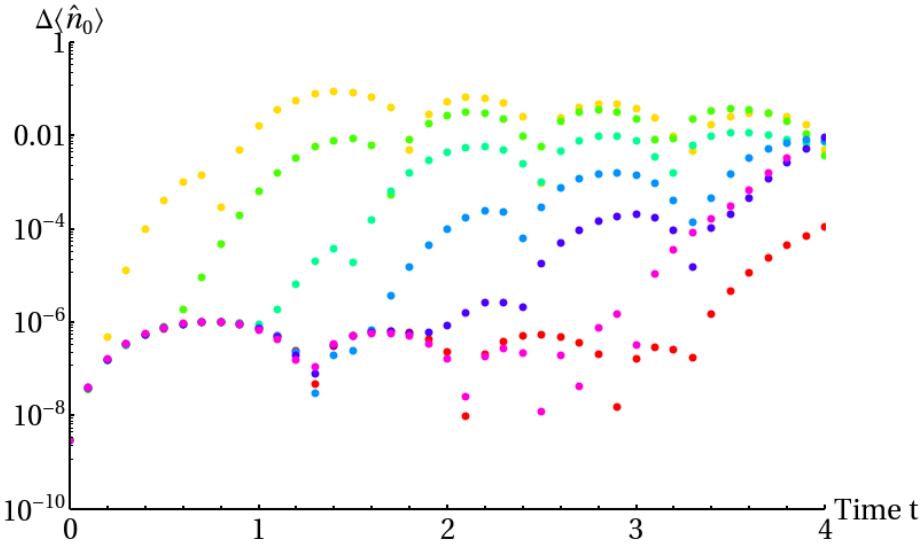

Figure 5.23: The results of several simulations of $\hat{n}_0$ in the Schrödinger picture on a chain of length $N = 32$, using a bond dimension of $d = 100$ and a trotter bond dimensions of $\tilde{D} = 16$. From left to right, the deviation from the analytical solution is plotted for $N_b = 1$ to $N_b = 7$. It clearly shows that the maximum number of bosons per site, N_b, has a great impact on the physical time scale one can simulate accurately.

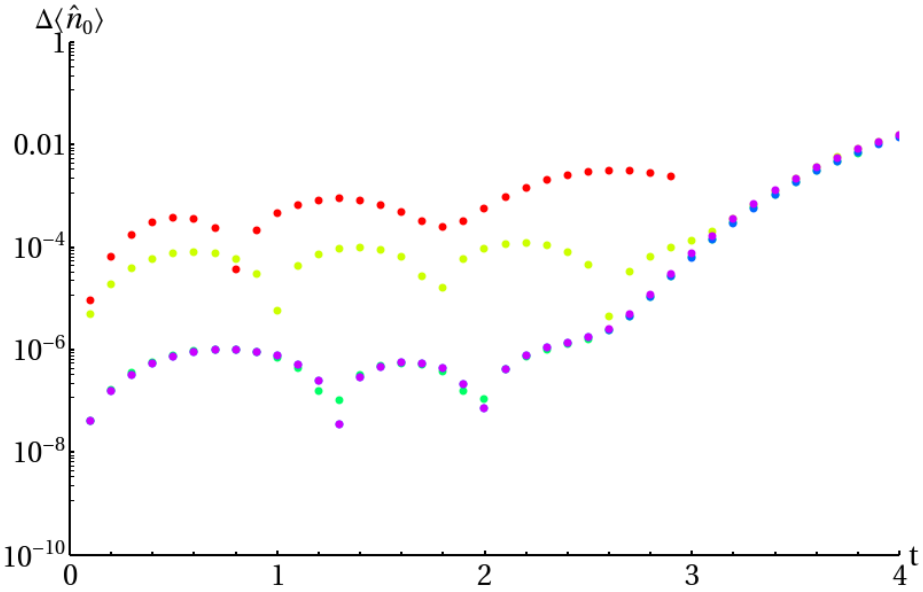

Figure 5.24: Comparison of several different maximum bond dimensions, $\tilde{D}$, of the Trotter MPO with a maximum of 7 bosons per site. $\tilde{D} \in \{4, 8, 16, 32, 42\}$. The simulation was run on a lattice of $N = 32$ and bond dimension $d = 100$ in the Schrödinger picture. One can see that starting from $\tilde{D} = 16$, the simulations are equivalent. The cut-off of at most seven bosons per site becomes the main source of inaccuracy at $t = 2.6$.

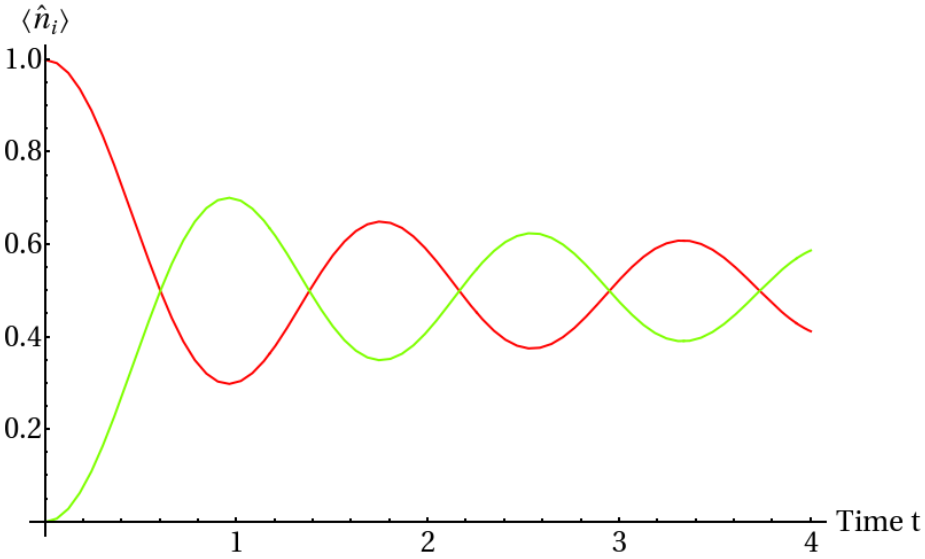

Figure 5.25: Analytical calculation of $\langle \hat{n}_0 \rangle$ (red) and $\langle \hat{n}_1 \rangle$ (green) as a function of physical time in the Bose-Hubbard model. The initial state was $|0, 1, 0, 1, \ldots\rangle$ on a system consisting of 32 sites. The parameters for this calculation are $t = 1$, $U = 0$ and $\mu = 0$.

5.4.3 Analytical solution

As mentioned, there is an analytical solution in the case of $U = \mu = 0$. In fact one can follow the same derivation as for the density operator in subsection 5.1.2 to get the predicted expectation value $\langle \hat{b}_i^\dagger \hat{b}_i \rangle$. One needs only to throw away the extra $-\frac{1}{2}$ that one gets in the calculation in subsection 5.1.2 and, in addition, the sum does not run over all of the sites on the left, but rather over all of the even sites, since our initial state is now $|0, 1, 0, 1, \ldots\rangle$ instead of $|\uparrow\uparrow \ldots \uparrow\downarrow\downarrow \ldots \downarrow\rangle$:

$$\langle \hat{b}_i^\dagger \hat{b}_i \rangle(t) = \sum_{i=-N/4+1}^{N/4} \left| \sum_{k=1}^{N} U_{n,k} e^{i\epsilon_k t} \left(U^\dagger\right)_{k,2i} \right|^2$$

Since the DMRG fails quickly with the Bose-Hubbard model , it is sufficient to simulate a smaller system than before. A plot is shown for $N = 32$ in figure 5.25. The current density is practically the same as the one for the XX-chain in (5.42) except one uses bosonic operators instead of the fermionic ones. The calculation remains the same though, so the end result is the same, except for the different initial state. The result is shown in figure 5.26.

$$\langle \hat{j}_n \rangle(t) = \mathrm{Im} \left(\sum_{i=-N/4+1}^{N/4} A^\star_{n-1,2i}(t) A_{n,2i}(t) \right) \tag{5.60}$$

5.4.4 Results

In figure 5.27, the deviation from the analytical solution is plotted for a simulation of the number operator, $\hat{n}_i$, at sites -10, 0, 1 and 11 and in figure 5.28,

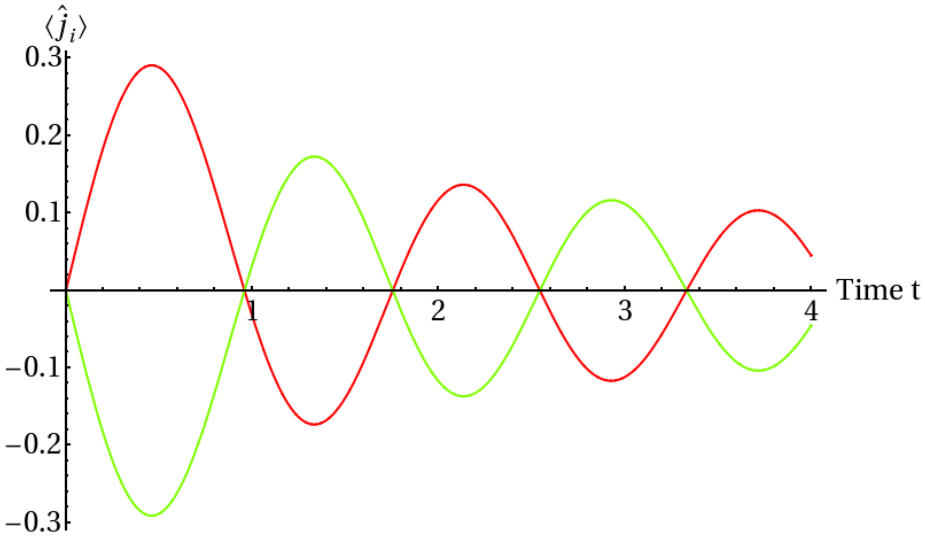

Figure 5.26: Analytical calculation of $\langle \hat{j}_0 \rangle$ (red) and $\langle \hat{j}_1 \rangle$ (green) as a function of physical time in the Bose-Hubbard model. The initial state was $|0, 1, 0, 1, \ldots\rangle$ on a system consisting of 32 sites. The parameters for this calculation are $t = 1$, $U = 0$ and $\mu = 0$.

the expectation values of $\hat{n}_0$ and $\hat{n}_1$ are plotted. One can see that the DMRG fails very quickly in the Bose-Hubbard model. In the Heisenberg picture, the deviation of simulations from the analytical solution is already greater than 0.01 at $t = 0.5$. Even though the simulation in the Schrödinger picture reaches only three time units before having a deviation of 0.01, it still fares substantially better. One can see that in the Schrödinger picture, the simulations have the same error, independent of the bond dimension, until roughly $t = 2$. This is because the error coming from the limitation on the number of bosons per site is dominant there. Also, the computation time in the Heisenberg picture is about sixteen times of that in the Schrödinger picture, while the memory requirement increases by a factor eight. These increases in computation time and memory requirement are due to the extra spin index that arises in MPO multiplications compared to applying an MPO to an MPS. Unfortunately, then, our hope that simulating in the Heisenberg picture would lead to better results for local site operators, seems to be unfounded. Figure 5.29 show the 'entanglement entropy' of the operator as defined in appendix D. A linear increase is clearly visible. Although the entropy appears to increase slower than in the other models, keep in mind that the compression is also more aggressive, due to the larger size of the Trotter MPO.

Some final words need to be said about the computation time required to perform simulations in both the Schrödinger and Heisenberg picture, since especially in simulating the Bose-Hubbard model, the difference in computation time is at its greatest. The simulation in the Schrödinger picture until $t = 0.4$ with $d = 80$ took almost two days. In comparison, the same simulation in the Heisenberg picture too eleven days to simulate just until $t = 0.1$. That's roughly a factor of 220 slower. Likewise in the case of $d = 40$, where the simulation in the Heisenberg got until $t = 1$ after eleven days, while the simulation in the

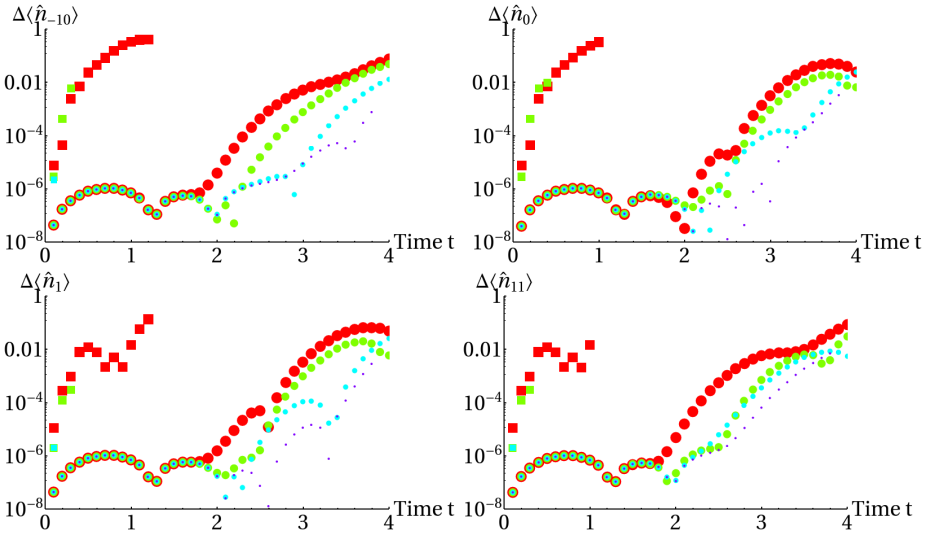

Figure 5.27: Simulation of $\langle \hat{n}_i \rangle$ for $i \in \{-10, 0, 1, 11\}$ compared to the analytical results as a function of physical time in the Bose-Hubbard model. The initial state was $|0, 1, 0, 1, \ldots\rangle$ on a system consisting of 32 sites. The parameters for this calculation are $t = 1$, $U = 0$ and $\mu = 0$ with at most seven bosons allowed per site. The squares belong to the simulation in the Heisenberg picture and the discs to the simulation in the Schrödinger picture. The simulation was run for four bond dimensions, $d = 40$ (red), $d = 60$ (green), $d = 80$ (light blue) and $d = 100$ (purple). The points for $d = 100$ are missing for the simulations in the Heisenberg picture, because the memory requirements for that were larger than was available. Also, note that there is only one point visible for $d = 80$ in the Heisenberg picture. This is due to the fact that that single point already took 11 days to calculate.

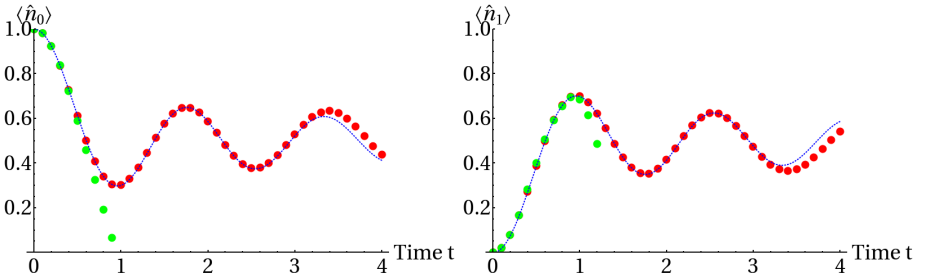

Figure 5.28: Simulation of $\langle \hat{n}_0 \rangle$ (left) and $\langle \hat{n}_1 \rangle$ (right) in the Bose-Hubbard model. The initial state was $|0, 1, 0, 1, \ldots\rangle$ on a system consisting of 32 sites. The parameters for this calculation are $t = 1$, $U = 0$ and $\mu = 0$ with at most seven bosons allowed per site. The bond dimension in both cases was $d = 40$. The simulation in the Schrödinger picture is displayed in red and the simulation in the Heisenberg picture is green. The blue dotted line is the analytical solution.

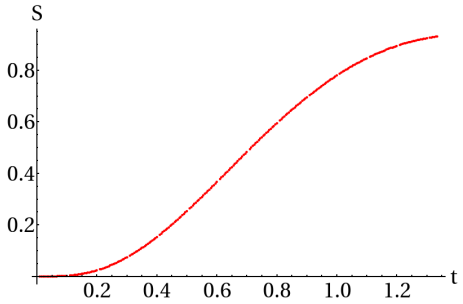

Figure 5.29: The 'entanglement entropy' for the $\hat{n}_0$ operator in the Bose-Hubbard model on a chain of length $N = 16$.

Schrödinger picture performed the same feat in just over seven hours. In that case the Heisenberg picture is a factor of 147 slower. Combined with the deplorable accuracy, the Heisenberg picture seems to be woefully unqualified to simulate the Bose-Hubbard model in.

Chapter 6

Conclusion

Of the four studied models, only the spin-$\frac{1}{2}$ XX chain turned out to be more accurate in the Heisenberg picture. This was solely due to the fact that the time evolving operators, $\hat{S}_i^z(t)$, $\hat{j}_i(t)$ and $S_i^{\hat{z}}S_j^z(t)$, could be described by an MPO with a small bond dimension, d. The compression procedure therefore has little effect in that case. The question if simulations perform better in the Heisenberg picture depends on both the Hamiltonian and on the studied operator. This became clear from the time evolution of the $\hat{S}_i^-$ operator. The time evolving version of that operator could not be described as an MPO with a small bond dimension d. Consequently, the simulation in the Heisenberg picture showed the same degradation as it did for the spin-$\frac{1}{2}$ XXZ chain and the Bose-Hubbard model. The left graph in figure 5.13 shows most strikingly why simulations in the Heisenberg picture break down. The compression procedure causes the operator to deviate exponentially from the truncated operator, because the singular values of the operator that are being thrown away are becoming to big. In the case of the XXZ chain, three quarters of the singular values are thrown away after every compression because the Trotter MPO has an bond dimension of $d = 4$, and therefore multiplying it with the time evolving MPO will increase the size of the latter by a factor four. In the case of the Bose-Hubbard model it is even worse, since with a compressed Trotter MPO of $d = 16$, a fraction of $\frac{15}{16}$ of the singular values are discarded. One therefore needs a very steep exponential decay of the singular values, but unfortunately the time evolution causes this exponential decay to become more shallow over time, as the right graph in figure 5.13 shows. Even though this is also the case in the Schrödinger picture, it appears to have less effect there, possibly due to the fact that the compression occurs twice as often, since the Trotter MPOs have to be applied to both sides of the operator. In the end, we must conclude that using the DMRG in the Heisenberg picture is only preferable if the operator under study can be described as an MPO with a small bond dimension as it evolves in time. In all other cases, the DMRG in the Schrödinger picture is preferable. Not only is it more accurate, DRMG in the Schrödinger picture is also at least three times faster than the DMRG in the Heisenberg picture and uses less memory in the process.

Appendix A

Abbreviations

A full list of abbreviations used throughout this thesis.

- DMRG: Density matrix renormalization group
- MPO: Matrix product operator
- MPS: Matrix product state
- SVD: Singular value decomposition

Appendix B

Decompositions

Calculations with MPOs and MPSs rely heavily on the various matrix decompositions. Therefore, these matrix decompositions are described here in more detail. Especially the SVD needs to be investigated, since the library used, *Lapack++ 3.5.2*, had trouble calculating complex SVDs.

B.1 QR and LQ decomposition

One way to get unitary matrices out of a matrix is to use a QR decomposition or similarly an LQ decomposition. It is preferred over an SVD, since it is generally faster. A QR decomposition of an $M \times N$ matrix with $M \leq N$ is $\mathcal{O}(N)$ compared to $\mathcal{O}(MN^2)$ for an SVD. The downside is that one does not get the singular values. With a QR decomposition, any $M \times N$ matrix A can be decomposed into a unitary $M \times M$ matrix Q and an upper triangular $M \times N$ matrix R.

$$A = QR \tag{B.1}$$

More specifically for $M > N$:

$$A = \begin{bmatrix} Q_1 & Q_2 \end{bmatrix} \begin{bmatrix} R \\ 0 \end{bmatrix} = Q_1 R \tag{B.2}$$

Where Q_1 is an $M \times N$ matrix, Q_2 is an $M \times (M-N)$ matrix and R is an $N \times N$ matrix. The matrix Q_2 therefore does not contain any useful information about A, but is still needed to have a unitary matrix Q. And for $M < N$:

$$\begin{bmatrix} A_1 & A_2 \end{bmatrix} = \begin{bmatrix} Q \end{bmatrix} \begin{bmatrix} R_1 & R_2 \end{bmatrix} \tag{B.3}$$

Here A is divided in an $M \times M$ matrix A_1 and an $M \times (N-M)$ matrix A_2 and likewise for R. One only needs to solve the QR decomposition for A_1 to get Q and R_1. The remaining matrix R_2 can then be calculated from A_2 using the fact that Q is unitary:

$$A_2 = QR_2 \;\Rightarrow\; R_2 = Q^\dagger A_2 \tag{B.4}$$

One can use the QR decomposition to left-canonize MPOs/MPSs, but for right canonization, one can use an LQ decomposition, which splits a matrix A into a

left triangular $M \times N$ matrix L and a unitary $N \times N$ matrix Q. One can obtain the LQ decomposition from a QR decomposition by taking the QR decomposition of $A^\dagger$.

$$A = \left(A^\dagger\right)^\dagger = \left(Q'R'\right)^\dagger = R'^\dagger Q'^\dagger = LQ. \tag{B.5}$$

In the last step $R'^\dagger$ is renamed to L, since the complex conjugate of a right triangular matrix is a lower triangular matrix. The desired unitary matrix Q is simply the complex conjugate of the Q' from the QR decomposition of $A^\dagger$.

B.2 Singular Value Decomposition

In order to perform many of the above calculations, it is crucial to have an accurate way of calculating the SVD. In our simulations *Lapack++ 2.5.3* was used to calculate the SVDs. Unfortunately, the calculation can occasionally fail due to the SVD not converging; most likely because *Lapack++ 2.5.3* uses an iterative method to calculate the SVD and the particular matrix it would fail on, would just need to many iterative steps. When that happens, the simulation would normally fail altogether. Fortunately, there are alternative ways to calculate the SVD, although they are slower and more memory-intensive. Since the SVD from *Lapack++* only fails rarely, this is not a problem. There are two alternative methods of calculating the SVD of a complex M by N matrix A.

- Eigenvalues of $A^\dagger A$ or $AA^\dagger$.
- Real-valued SVD of $\begin{bmatrix} A_R & -A_I \\ A_I & A_R \end{bmatrix}$, where A_R is the real part of A and A_I is the imaginary part of A.

Since the real-valued SVD still relies on the *Lapack++* routine, it can still fail. The best approach is to try the real-valued SVD first and if that fails too, then to use the $A^\dagger A$-method.

B.2.1 Eigenvalues of $A^\dagger A$ or $AA^\dagger$

One method to find the SVD is to find the eigenvalues of $A^\dagger A$ or $AA^\dagger$ when $M > N$ or $M \leq N$ respectively. These eigenvalues will be the squares of the singular values. For $A^\dagger A$:

$$\begin{aligned} A^\dagger A\vec{v}_i &= \left(USV^\dagger\right)^\dagger USV^\dagger\vec{v}_i = VS^\mathsf{T}U^\dagger USV^\dagger\vec{v}_i \\ &= VS^\mathsf{T}SV^\dagger\vec{v}_i = VS^\mathsf{T}S\hat{e}_i = Vs_i^2\hat{e}_i = s_i^2\vec{v}_i \end{aligned} \tag{B.6}$$

The vectors v_i are the columns of V and are also the eigenvectors of $A^\dagger A$. Thus once one has the eigenvalues and eigenvectors, one can already construct S and V. U can be obtained by

$$A = USV^\dagger \;\Rightarrow\; AV = US \Rightarrow U = AVS^{-1} \tag{B.7}$$

The matrix S^{-1} contains $\frac{1}{s_i}$ on the diagonal entries. For $AA^\dagger$, the calculation is similar.

$$\begin{aligned} AA^\dagger\vec{u}_i &= USV^\dagger\left(USV^\dagger\right)^\dagger\vec{u}_i = USV^\dagger VS^\mathsf{T}U^\dagger\vec{u}_i \\ &= USS^\mathsf{T}U^\dagger\vec{u}_i = USS^\mathsf{T}\hat{e}_i = Us_i^2\hat{e}_i = s_i^2\vec{u}_i \end{aligned} \tag{B.8}$$

The vectors u_i are the columns of U and are also the eigenvectors of $AA^\dagger$. Meaning one can again use the eigenvalues and eigenvectors to construct S and U. V can be obtained by

$$A^\dagger = VS^\mathsf{T}U^\dagger \;\Rightarrow\; A^\dagger U = VS^\mathsf{T} \Rightarrow V = A^\dagger U \left(S^\mathsf{T}\right)^{-1} \tag{B.9}$$

The matrix $\left(S^\mathsf{T}\right)^{-1}$ again contains the entries $\frac{1}{s_i}$ on the diagonal. However, in both (B.7) and (B.9), one has to divide by the singular values s_i. Since some of these singular values can be quite small and indeed sometimes 0, this can lead to problems. The accuracy will decrease drastically for the columns of U or V respectively, where s_i is very small. In that case, one must avoid using this method to calculate the columns of U or V respectively that belong to small singular values. Instead, one can use the Gram-Schmidt process to generate vectors that are orthonormal to the columns that belong to the large singular values. Since these generated vectors belong to small singular values, they will not influence the calculated SVD much, but they do introduce an inaccuracy. It's therefore better to try the routine from *Lapack++* first, and if that fails to try the real-valued SVD. Only if the latter method fails too, should this routine be used as a last resort.

B.2.2 Real-valued SVD

Since complex SVD's are generally more difficult to calculate, another approach is to calculate the SVD of the matrix $\begin{bmatrix} A_R & -A_I \\ A_I & A_R \end{bmatrix}$, where A_R is the real part of A and A_I is the imaginary part of A. To see why this is so, consider the definition of an SVD:

$$A\vec{v}_j = s_j\vec{u}_j \tag{B.10}$$

$$A^\dagger\vec{u}_j = s_j\vec{v}_j \tag{B.11}$$

Where the s_j denote the singular values and $\vec{u}_j$, and $\vec{v}_j$ are the left and right singular vectors respectively. If we explicitly write out the real and imaginary part for (B.10), we get the following:

$$A\vec{v}_j = s_j\vec{u}_j \tag{B.12}$$

$$\Rightarrow (A_R + iA_I)\left(\vec{v}_{R_j} + i\vec{v}_{I_j}\right) = s_j\left(\vec{u}_{R_j} + i\vec{u}_{I_j}\right) \tag{B.13}$$

$$\Rightarrow A_R\vec{v}_{R_j} + A_I\vec{v}_{I_j} + i\left(A_I\vec{v}_{R_j} + A_R\vec{v}_{I_j}\right) = s_j\vec{u}_{R_j} + is_j\vec{u}_{I_j} \tag{B.14}$$

One can then split the real and imaginary part, which gives two equations:

$$\begin{cases} A_R\vec{v}_{R_j} - A_I\vec{v}_{I_j} = s_j\vec{u}_{R_j} \\ A_I\vec{v}_{R_j} + A_R\vec{v}_{I_j} = s_j\vec{u}_{I_j} \end{cases} \tag{B.15}$$

These can be written in matrix form:

$$\begin{cases} \begin{bmatrix} A_R & -A_I \end{bmatrix} \begin{bmatrix} \vec{v}_{R_j} \\ \vec{v}_{I_j} \end{bmatrix} = s_j\vec{u}_{R_j} \\ \begin{bmatrix} A_I & A_R \end{bmatrix} \begin{bmatrix} \vec{v}_{R_j} \\ \vec{v}_{I_j} \end{bmatrix} = s_j\vec{u}_{I_j} \end{cases} \tag{B.16}$$

These two equations can be combined into one:

$$\begin{bmatrix} A_R & -A_I \\ A_I & A_R \end{bmatrix} \begin{bmatrix} \vec{v}_{R_j} \\ \vec{v}_{I_j} \end{bmatrix} = s_j \begin{bmatrix} \vec{u}_{R_j} \\ \vec{u}_{I_j} \end{bmatrix} \tag{B.17}$$

And similarly with (B.11):

$$\begin{aligned}
& A^\dagger \vec{u}_j = s_j \vec{v}_j \\
\Rightarrow\ & \left(A_R^\mathsf{T} - iA_I^\mathsf{T}\right)\left(\vec{u}_{R_j} + i\vec{u}_{I_j}\right) = s_j\left(\vec{v}_{R_j} + i\vec{v}_{I_j}\right) \\
\Rightarrow\ & A_R^\mathsf{T}\vec{u}_{R_j} + A_I^\mathsf{T}\vec{u}_{I_j} + i\left(-A_I^\mathsf{T}\vec{u}_{R_j} + A_R^\mathsf{T}\vec{u}_{I_j}\right) = s_j\vec{v}_{R_j} + is_j\vec{v}_{I_j} \\
\Rightarrow\ & \begin{cases} A_R^\mathsf{T}\vec{u}_{R_j} + A_I^\mathsf{T}\vec{u}_{I_j} = s_j\vec{v}_{R_j} \\ -A_I^\mathsf{T}\vec{u}_{R_j} + A_R^\mathsf{T}\vec{u}_{I_j} = s_j\vec{v}_{I_j} \end{cases} \\
\Rightarrow\ & \begin{cases} \begin{bmatrix} A_R^\mathsf{T} & A_I^\mathsf{T} \end{bmatrix} \begin{bmatrix} \vec{u}_{R_j} \\ \vec{u}_{I_j} \end{bmatrix} = s_j\vec{v}_{R_j} \\ \begin{bmatrix} -A_I^\mathsf{T} & A_R^\mathsf{T} \end{bmatrix} \begin{bmatrix} \vec{u}_{R_j} \\ \vec{u}_{I_j} \end{bmatrix} = s_j\vec{v}_{I_j} \end{cases} \\
\Rightarrow\ & \begin{bmatrix} A_R^\mathsf{T} & A_I^\mathsf{T} \\ -A_I^\mathsf{T} & A_R^\mathsf{T} \end{bmatrix} \begin{bmatrix} \vec{u}_{R_j} \\ \vec{u}_{I_j} \end{bmatrix} = s_j \begin{bmatrix} \vec{v}_{R_j} \\ \vec{v}_{I_j} \end{bmatrix} \\
\Rightarrow\ & \begin{bmatrix} A_R & -A_I \\ A_I & A_R \end{bmatrix}^\mathsf{T} \begin{bmatrix} \vec{u}_{R_j} \\ \vec{u}_{I_j} \end{bmatrix} = s_j \begin{bmatrix} \vec{v}_{R_j} \\ \vec{v}_{I_j} \end{bmatrix}
\end{aligned} \tag{B.18}$$

This shows that the SVD of $\begin{bmatrix} A_R & -A_I \\ A_I & A_R \end{bmatrix}$ will also give us the singular values and matrices U and V. The singular values from the original matrix A will now appear twice in the larger matrix because of the following identity

$$\begin{bmatrix} A_R & -A_I \\ A_I & A_R \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \begin{bmatrix} A_R & -A_I \\ A_I & A_R \end{bmatrix} \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \tag{B.19}$$

Inserting this in (B.17) gives us another solution with the same singular values as (B.17):

$$\begin{aligned}
& \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \begin{bmatrix} A_R & -A_I \\ A_I & A_R \end{bmatrix} \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} \vec{v}_{R_j} \\ \vec{v}_{I_j} \end{bmatrix} = s_j \begin{bmatrix} \vec{u}_{R_j} \\ \vec{u}_{I_j} \end{bmatrix} \\
\Rightarrow\ & \begin{bmatrix} A_R & -A_I \\ A_I & A_R \end{bmatrix} \begin{bmatrix} -\vec{v}_{I_j} \\ \vec{v}_{R_j} \end{bmatrix} = s_j \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} \vec{u}_{R_j} \\ \vec{u}_{I_j} \end{bmatrix} \\
\Rightarrow\ & \begin{bmatrix} A_R & -A_I \\ A_I & A_R \end{bmatrix} \begin{bmatrix} -\vec{v}_{I_j} \\ \vec{v}_{R_j} \end{bmatrix} = s_j \begin{bmatrix} -\vec{u}_{I_j} \\ \vec{u}_{R_j} \end{bmatrix}
\end{aligned} \tag{B.20}$$

This means that every singular value will appear twice with two different vectors. Sorting the singular values appropriately, we can write the SVD of the large matrix as follows:

$$\begin{bmatrix} A_R & -A_I \\ A_I & A_R \end{bmatrix} = \begin{bmatrix} U_R & -U_I \\ U_I & U_R \end{bmatrix} \begin{bmatrix} S & 0 \\ 0 & S \end{bmatrix} \begin{bmatrix} V_R & -V_I \\ V_I & V_R \end{bmatrix}^\mathsf{T} \tag{B.21}$$

Note that in (B.20) an arbitrary sign can be introduced. This means that the second column of the 'U' and 'V' of the large matrix can have a different sign. Normally, one would need to take care with degenerate singular values, because the accompanying vectors may not be orthonormal. Luckily, *Lapack++* takes care of this for us and it should always provide us with unitary matrices.

Appendix C

Minimal bond dimension for exact solution

In the case of the spin-$\frac{1}{2}$ XX chain the time evolution of the $\hat{S}_z$ operator can be described by an MPO with a bond dimension of only $d = 4$ [17]. As a prelude for the $\hat{S}^+$ operator, we will show the case of the $\hat{S}^z$ operator again. First, the Jordan-Wigner transformation of (5.14) has to be performed again:

$$\hat{S}_n^z = \hat{c}_n^\dagger \hat{c}_n - \frac{1}{2} \tag{C.1}$$

This can be written in the diagonalized operators from (5.22) and we can drop the $\frac{1}{2}$ for simplicity:

$$\hat{c}_n^\dagger \hat{c}_n = \sum_{k=1}^{N} \hat{d}_k^\dagger \left(U^\dagger\right)_{k,n} \sum_{\ell=1} U_{n,\ell} \hat{d}_\ell \tag{C.2}$$

We can insert the time evolution from (5.27) and convert back to the previous fermionic operators:

$$\begin{aligned} &= \sum_{k=1}^{N} \hat{d}_k^\dagger(0) e^{i\epsilon_k t} \left(U^\dagger\right)_{k,n} \sum_{\ell=1}^{N} U_{n,\ell} \hat{d}_\ell(0) e^{-i\epsilon_\ell t} \\ &= \sum_{k=1}^{N} \sum_{i=-N/2+1}^{N/2} \hat{c}_i^\dagger U_{i,k} e^{i\epsilon_k t} \left(U^\dagger\right)_{k,n} \sum_{\ell=1}^{N} \sum_{j=-N/2+1}^{N/2} U_{n,\ell} e^{-i\epsilon_\ell t} \left(U^\dagger\right)_{\ell,j} \hat{c}_j \end{aligned} \tag{C.3}$$

Finally, we can reverse the Jordan-Wigner transformation to arrive at an operator we can write in MPO form. To shorten the notation, we can write $U_{i,k} e^{i\epsilon_k t} \left(U^\dagger\right)_{k,n} = \left(e^{iTt}\right)_{i,n}$, where T is the Hermitian matrix from (5.20):

$$= \sum_{i=-N/2+1}^{N/2} \hat{S}_i^+ (-1)^{\phi_i} \left(e^{iTt}\right)_{i,n} \sum_{j=-N/2+1}^{N/2} \left(e^{-iTt}\right)_{n,j} (-1)^{\phi_j} \hat{S}_j^- \tag{C.4}$$

This is essentially a product of two MPOs, which can be seen by looking at the first part and plugging in the definition of ϕ_i from the Jordan-Wigner transfor-

mation:

$$\sum_{i=-N/2+1}^{N/2} (-1)^{\sum_{m=-N/2+1}^{i-1}\left(\hat{S}_m^z+\frac{1}{2}\right)} \hat{S}_i^+ \left(e^{iTt}\right)_{i,n} \tag{C.5}$$

$$= \sum_{i=-N/2+1}^{N/2} \left(\prod_{m=-N/2+1}^{i-1} (-1)^{\hat{S}_m^z+\frac{1}{2}} \right) \hat{S}_i^+ \left(e^{iTt}\right)_{i,n} \tag{C.6}$$

This complicated looking expression is actually only an MPO with a bond dimension of $d = 2$:

$$B^{[-N/2+1]} = \begin{bmatrix} \hat{S}^+ & (-1)^{\hat{S}^z+\frac{1}{2}} \end{bmatrix} \tag{C.7}$$

$$B^{[i]} = \begin{bmatrix} 1 & 0 \\ \hat{S}^+ & (-1)^{\hat{S}^z+\frac{1}{2}} \end{bmatrix} \tag{C.8}$$

$$B^{[N/2]} = \begin{bmatrix} 1 \\ \hat{S}^+ \end{bmatrix} \tag{C.9}$$

The same is true for the second part of (C.4), which can also be written as an MPO with a bond dimension of $d = 2$. Multiplying both MPOs gives an MPO with a bond dimension of $d = 4$.

Now that it is clear how it works for the $\hat{S}^z$ operator, let us look at the $\hat{S}^+$ operator. We again apply the Jordan-Wigner transformation:

$$\hat{S}_n^+ = \hat{c}_n^\dagger (-1)^{\phi_n} = \hat{c}_n^\dagger \prod_{m=-N/2+1}^{n} (-1)^{\hat{c}_m^\dagger \hat{c}_m} \quad , \tag{C.10}$$

and again plug in the diagonal operators with their time evolution:

$$= \sum_{k=1}^{N} \hat{d}_k^\dagger(0) \left(U^\dagger\right)_{k,n} e^{i\epsilon_k t} \quad \times$$
$$\prod_{m=-N/2+1}^{n} (-1)^{\sum_{i=1}^{N} \hat{d}_i^\dagger(0)\left(U^\dagger\right)_{i,m} e^{i\epsilon_i t} \sum_{j=1} U_{m,j} \hat{d}_j(0) e^{-i\epsilon_j t}} \tag{C.11}$$

Reverting back to the fermionic operators again and undoing the Jordan-Wigner

transformation, we have,

$$
\begin{aligned}
&= \sum_{k=1}^{N} \sum_{\ell=-N/2+1}^{N/2} \hat{c}_\ell^\dagger \, U_{\ell,k} \, e^{i\epsilon_k t} \left(U^\dagger\right)_{k,n} \quad \times \\
&\qquad \prod_{m=-N/2+1}^{n} (-1)^{\sum_{i=1}^{N} \sum_{r=-N/2+1}^{N/2} \hat{c}_r^\dagger U_{r,i} \left(U^\dagger\right)_{i,m} e^{i\epsilon_i t} \times} \\
&\qquad\qquad\qquad\qquad \blacksquare^{\times \sum_{j=1}^{N} \sum_{s=-N/2+1}^{N/2} U_{m,j} \hat{c}_s \left(U^\dagger\right)_{j,s} e^{-i\epsilon_j t}} \\
&= \sum_{\ell=-N/2+1}^{N/2} \hat{S}_\ell^+ (-1)^{\phi_\ell} \left(e^{iTt}\right)_{\ell,n} \quad \times \\
&\qquad \prod_{m=-N/2+1}^{n} (-1)^{\sum_{r=-N/2+1}^{N/2} \hat{S}_r^+ (-1)^{\phi_r} \left(e^{iTt}\right)_{r,m} \sum_{s=-N/2+1}^{N/2} \left(e^{-iTt}\right)_{m,s} (-1)^{\phi_s} \hat{S}_s^-},
\end{aligned}
\tag{C.12}
$$

where the black square only indicates that the exponent continues from the previous line. The first part of this expression can again be written as an MPO with a bond dimension of $d = 2$, but the product of the $(-1)^{\cdots}$'s can not easily be written as an MPO, especially since nothing in the exponent commutes. This is why the $\hat{S}^+$ operator of section 5.3 does not reach an exact solution in the Heisenberg picture, as the $\hat{S}^z$ operator did in section 5.1, even though both use the time evolution of the spin-$\frac{1}{2}$ XX chain.

Appendix D

'Entanglement entropy' for MPOs

It is well known that one can use the Von Neumann entropy to measure the level of entanglement in states. Such a thing makes no sense for operators of course, but one could use the same method to calculate such a quantity from MPOs in exactly the same way one would for MPSs. To make this clear, we shall briefly show how one can obtain the entanglement entropy from an MPS.

We start with a system of N sites and divide this into two parts. For the sake of convenience, we will just split it right through the middle. This means we will focus on the entanglement that the left part, A, of the system has with the right part, B. One can then write the state of the system as a Schmidt decomposition:

$$|\Psi\rangle = \sum_i \lambda_i |\psi_i\rangle_A |\phi_i\rangle_B \tag{D.1}$$

The states $|\psi_i\rangle_A$ and $|\phi_i\rangle_B$ are the basis states of the systems A and B respectively. The real number λ_i is known as the Schmidt coefficient. The Von Neumann entropy for a reduced density matrix, ρ_A is defined as:

$$S(\rho_A) = -\operatorname{Tr}_A(\rho_A \ln \rho_A) \tag{D.2}$$

We therefore need to rewrite the state $|\Psi\rangle$ in the density matrix formalism:

$$\rho = |\Psi\rangle\langle\Psi| = \sum_i \lambda_i |\psi_i\rangle_A |\phi_i\rangle_B \sum_j \lambda_j \langle\psi_j|_A \langle\phi_j|_B \tag{D.3}$$

The reduced density matrix, ρ_A, is obtained by tracing out system B:

$$\begin{aligned}
\rho_A &= \operatorname{Tr}_B(\rho) \\
&= \sum_m \langle\phi_m|_B \sum_i \lambda_i |\psi_i\rangle_A |\phi_i\rangle_B \sum_j \lambda_j \langle\psi_j|_A \langle\phi_j|_B |\phi_m\rangle_B \\
&= \sum_m \sum_i \lambda_i |\psi_i\rangle_A \delta_{mi} \sum_j \lambda_j \langle\psi_j|_A \delta_{jm} \\
&= \sum_m \lambda_m^2 |\psi_m\rangle\langle\psi_m|_A
\end{aligned} \tag{D.4}$$

Plugging this into (D.2) gives us a formula for the entropy:

$$S = -\sum_m \lambda_m^2 \ln\left(\lambda_m^2\right) \tag{D.5}$$

To actually calculate this quantity, we need to know what the Schmidt coefficients are. This is especially easy in the case of an MPS, since the form is already very similar to the Schmidt decomposition. Recall from (3.2) the definition of an MPS:

$$|\Psi\rangle = p \sum_{\boldsymbol{\sigma}} \sum_{a_1,\ldots} A^{\sigma_1}_{a_0,a_1} A^{\sigma_2}_{a_1,a_2} A^{\sigma_3}_{a_2,a_3} \ldots A^{\sigma_N}_{a_{N-1},a_N} |\boldsymbol{\sigma}\rangle \tag{D.6}$$

We need to split this into two parts, which can be accomplished by performing a suitable SVD:

$$\begin{aligned}
|\Psi\rangle &= p \sum_{\boldsymbol{\sigma}} \sum_{a_1,\ldots} \ldots A^{\sigma_{N/2-1}}_{a_{N/2-1},a_{N/2}} A^{\sigma_{N/2}}_{a_{N/2},a_{N/2+1}} A^{\sigma_{N/2+1}}_{a_{N/2+1},a_{N/2+2}} \ldots |\boldsymbol{\sigma}\rangle \\
&= p \sum_{\boldsymbol{\sigma}} \sum_{a_1,\ldots} \ldots A^{\sigma_{N/2-1}}_{a_{N/2-1},a_{N/2}} A_{\left(\sigma_{N/2},a_{N/2}\right),a_{N/2+1}} A^{\sigma_{N/2+1}}_{a_{N/2+1},a_{N/2+2}} \ldots |\boldsymbol{\sigma}\rangle \\
&= p \sum_{s_{N/2}} \sum_{\boldsymbol{\sigma}} \sum_{a_1,\ldots} \ldots A^{\sigma_{N/2-1}}_{a_{N/2-1},a_{N/2}} U_{\left(\sigma_{N/2},a_{N/2}\right),s_{N/2}} S_{s_{N/2},s_{N/2}} \quad \times \\
&\qquad (V^\dagger)_{s_{N/2},a_{N/2+1}} A^{\sigma_{N/2+1}}_{a_{N/2+1},a_{N/2+2}} \ldots |\boldsymbol{\sigma}\rangle \\
&= p \sum_{s_{N/2}} \sum_{\boldsymbol{\sigma}} \sum_{a_1,\ldots} \ldots A^{\sigma_{N/2-1}}_{a_{N/2-1},a_{N/2}} U^{\sigma_{N/2}}_{a_{N/2},s_{N/2}} S_{s_{N/2},s_{N/2}} \quad \times \\
&\qquad (V^\dagger)_{s_{N/2},a_{N/2+1}} A^{\sigma_{N/2+1}}_{a_{N/2+1},a_{N/2+2}} \ldots |\boldsymbol{\sigma}\rangle
\end{aligned} \tag{D.7}$$

This is exactly the form of (D.1), when we make the following observations:

$$i = s_{N/2} \tag{D.8}$$

$$\lambda_i = S_{s_{N/2},s_{N/2}} \tag{D.9}$$

$$|\psi_i\rangle_A = p \sum_{\sigma_1,\ldots,\sigma_{N/2}} \sum_{a_1,\ldots,a_{N/2}} \ldots A^{\sigma_{N/2-1}}_{a_{N/2-1},a_{N/2}} U^{\sigma_{N/2}}_{a_{N/2},i} |\sigma_1 \ldots \sigma_{N/2}\rangle \tag{D.10}$$

$$\begin{aligned}
|\phi_i\rangle_B = &\sum_{\sigma_{N/2+1},\ldots,\sigma_N} \sum_{a_{N/2+1},\ldots,a_{N-1}} (V^\dagger)_{i,a_{N/2+1}} \quad \times \\
&A^{\sigma_{N/2+1}}_{a_{N/2+1},a_{N/2+2}} \ldots |\sigma_{N/2+1} \ldots \sigma_N\rangle
\end{aligned} \tag{D.11}$$

One can see that the Schmidt coefficients are the same as the singular values. Better yet, these singular values are already calculated during the compression algorithm in section 3.7.

There is nothing stopping us from doing the same thing for an MPO. The only difference is that the calculated quantity does not have a physical meaning, as it did for an MPS. Nevertheless, it behaves in the same way and, as such, gives us more insight into the behaviour of simulations in the Heisenberg picture.

We obtain the singular values in the same way as we did in (D.7):

$$
\begin{aligned}
|\Psi\rangle &= q \sum_{\boldsymbol{\sigma},\boldsymbol{\sigma'}} \sum_{b_1,\dots} \dots B^{\sigma_{N/2-1},\sigma'_{N/2-1}}_{b_{N/2-1},b_{N/2}} B^{\sigma_{N/2},\sigma'_{N/2}}_{b_{N/2},b_{N/2+1}} B^{\sigma_{N/2+1},\sigma'_{N/2+1}}_{b_{N/2+1},b_{N/2+2}} \dots |\boldsymbol{\sigma}\rangle\langle\boldsymbol{\sigma'}| \\
&= q \sum_{\boldsymbol{\sigma},\boldsymbol{\sigma'}} \sum_{b_1,\dots} \dots B^{\sigma_{N/2-1},\sigma'_{N/2-1}}_{b_{N/2-1},b_{N/2}} B_{\left(\sigma_{N/2},\sigma'_{N/2},b_{N/2}\right),b_{N/2+1}} \quad \times \\
&\qquad B^{\sigma_{N/2+1},\sigma'_{N/2+1}}_{b_{N/2+1},b_{N/2+2}} \dots |\boldsymbol{\sigma}\rangle\langle\boldsymbol{\sigma'}| \\
&= q \sum_{s_{N/2}} \sum_{\boldsymbol{\sigma},\boldsymbol{\sigma'}} \sum_{b_1,\dots} \dots B^{\sigma_{N/2-1},\sigma'_{N/2-1}}_{b_{N/2-1},b_{N/2}} U_{\left(\sigma_{N/2},\sigma'_{N/2},b_{N/2}\right),s_{N/2}} \quad \times \\
&\qquad S_{s_{N/2},s_{N/2}} (V^\dagger)_{s_{N/2},b_{N/2+1}} B^{\sigma_{N/2+1},\sigma'_{N/2+1}}_{b_{N/2+1},b_{N/2+2}} \dots |\boldsymbol{\sigma}\rangle\langle\boldsymbol{\sigma'}| \\
&= q \sum_{s_{N/2}} \sum_{\boldsymbol{\sigma},\boldsymbol{\sigma'}} \sum_{b_1,\dots} \dots B^{\sigma_{N/2-1},\sigma'_{N/2-1}}_{b_{N/2-1},b_{N/2}} U^{\sigma_{N/2},\sigma'_{N/2}}_{b_{N/2},s_{N/2}} S_{s_{N/2},s_{N/2}} \quad \times \\
&\qquad (V^\dagger)_{s_{N/2},b_{N/2+1}} B^{\sigma_{N/2+1},\sigma'_{N/2+1}}_{b_{N/2+1},b_{N/2+2}} \dots |\boldsymbol{\sigma}\rangle\langle\boldsymbol{\sigma'}| \qquad \text{(D.12)}
\end{aligned}
$$

The singular values, $S_{s_{N/2},s_{N/2}}$, can be used as the λ_m in (D.2), providing us with a sort of 'entanglement entropy' for MPOs.

A few more words need to be said about the maximum entropy, since this is dependent on the number of singular values, d, and therefore influenced by the compression algorithm. To calculate the maximum entropy, we can simply take the derivative with respect to every variable with the constraint that they add up to 1 (in the case of canonized MPOs). If we set $x_m = \lambda_m^2$ in (D.5) for ease of calculation, we need to minimize

$$
S = -\sum_{m=1}^{d} x_m \ln x_m \quad . \qquad \text{(D.13)}
$$

Since we have the constraint $\sum_{m=1}^{d} x_m = 1$, we can substitute $x_d = 1 - \sum_{m=1}^{d-1} x_m$ in the above equation.

$$
S = -\sum_{m=1}^{d-1} x_m \ln x_m - \left(1 - \sum_{m=1}^{d-1} x_m\right) \ln \left(1 - \sum_{m=1}^{d-1} x_m\right) \qquad \text{(D.14)}
$$

Taking the derivative with respect to an x_i yields:

$$
\begin{aligned}
\frac{\partial}{\partial x_i} S &= -\ln x_i - \frac{x_i}{x_i} + \ln \left(1 - \sum_{m=1}^{d-1} x_m\right) + \frac{\left(1 - \sum_{m=1}^{d-1} x_m\right)}{\left(1 - \sum_{m=1}^{d-1} x_m\right)} \\
&= -\ln x_i + \ln \left(1 - \sum_{m=1}^{d-1} x_m\right) \qquad \text{(D.15)}
\end{aligned}
$$

Setting this to zero gives $d-1$ equations, which are all the same:

$$\begin{aligned} -\ln x_i + \ln\left(1 - \sum_{m=1}^{d-1} x_m\right) = 0 \quad &\Rightarrow \\ x_i = 1 - \sum_{m=1}^{d-1} x_m \quad &\Rightarrow \\ 2x_i = 1 - \sum_{\substack{m=1 \\ m\neq i}}^{d-1} x_m \end{aligned} \tag{D.16}$$

Since all of these equations are the same, and it is a system of $d-1$ equations with $d-1$ variables, this means that every x_i is necessarily the same, i.e. $x = x_i \; \forall i$. Plugging this in the equation above gives

$$2x = 1 - (d-2)x \quad \Rightarrow \quad d\,x = 1 \quad \Rightarrow \quad x = \frac{1}{d} \quad . \tag{D.17}$$

By using the constraint, one can see that also x_d is equal to $\frac{1}{d}$. Plugging this result back into the formula for the entropy, (D.13), one ends up with the maximum possible entropy:

$$S = -\sum_{m=1}^{d} x_m \ln x_m \tag{D.18}$$

$$= -\sum_{m=1}^{d} \frac{1}{d} \ln \frac{1}{d} \tag{D.19}$$

$$= \ln d \tag{D.20}$$

Since d is the maximum allowed bond dimension, one can see that the compression algorithm, which always truncates the matrices in an MPO to at most d singular values, will create an upper limit of the 'entanglement entropy' for an MPO to $\ln d$.

Bibliography

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

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

[3] K. G. Wilson, "The renormalization group: Critical phenomena and the Kondo problem," *Rev. Mod. Phys.*, vol. 47, pp. 773–840, Oct 1975.

[4] R. Žitko and T. Pruschke, "Energy resolution and discretization artifacts in the numerical renormalization group," *Phys. Rev. B*, vol. 79, p. 085106, Feb 2009.

[5] E. S. Sørensen and I. Affleck, "Large-scale numerical evidence for Bose condensation in the $S = 1$ antiferromagnetic chain in a strong field," *Phys. Rev. Lett.*, vol. 71, pp. 1633–1636, Sep 1993.

[6] S. Rapsch, U. Schollwöck, and W. Zwerger, "Density matrix renormalization group for disordered bosons in one dimension," *EPL (Europhysics Letters)*, vol. 46, no. 5, p. 559, 1999.

[7] T. Senthil, J. B. Marston, and M. P. A. Fisher, "Spin quantum Hall effect in unconventional superconductors," *Phys. Rev. B*, vol. 60, pp. 4245–4254, Aug 1999.

[8] S. Östlund and S. Rommer, "Thermodynamic Limit of Density Matrix Renormalization," *Phys. Rev. Lett.*, vol. 75, pp. 3537–3540, Nov 1995.

[9] J. Dukelsky, M. A. Martín-Delgado, T. Nishino, and G. Sierra, ""Equivalence of the variational matrix product method and the density matrix renormalization group applied to spin chains"," *EPL (Europhysics Letters)*, vol. 43, no. 4, p. 457, 1998.

[10] J. M. Román, G. Sierra, J. Dukelsky, and M. A. Martín-Delgado, "The matrix product approach to quantum spin ladders," *Journal of Physics A: Mathematical and General*, vol. 31, no. 48, p. 9729, 1998.

[11] S. R. White and A. E. Feiguin, "Real-Time Evolution Using the Density Matrix Renormalization Group," *Phys. Rev. Lett.*, vol. 93, p. 076401, Aug 2004.

[12] A. J. Daley, C. Kollath, U. Schollwöck, and G. Vidal, "Time-dependent density-matrix renormalization-group using adaptive effective Hilbert spaces," *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2004, no. 04, p. P04005, 2004.

[13] G. Vidal, "Efficient Simulation of One-Dimensional Quantum Many-Body Systems," *Phys. Rev. Lett.*, vol. 93, p. 040502, Jul 2004.

[14] I. Pižorn and T. c. v. Prosen, "Operator space entanglement entropy in XY spin chains," *Phys. Rev. B*, vol. 79, p. 184416, May 2009.

[15] D. Muth, R. Unanyan, and M. Fleischhauer, "Dynamical simulation of integrable and non-integrable models in the Heisenberg picture," *ArXiv e-prints*, Sept. 2010.

[16] T. c. v. Prosen and M. Žnidarič, "Is the efficiency of classical simulations of quantum dynamics related to integrability?," *Phys. Rev. E*, vol. 75, p. 015202, Jan 2007.

[17] M. J. Hartmann, J. Prior, S. R. Clark, and M. B. Plenio, "Density Matrix Renormalization Group in the Heisenberg Picture," *Physical Review Letters*, vol. 102, p. 057202, Feb. 2009.

[18] U. Schollwöck, "The density-matrix renormalization group in the age of matrix product states," *Annals of Physics*, vol. 326, no. 1, pp. 96 – 192, 2011. January 2011 Special Issue.

[19] I. P. McCulloch, "From density-matrix renormalization group to matrix product states," *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2007, no. 10, p. P10014, 2007.

[20] J. J. García-Ripoll, "Time evolution of Matrix Product States," *New Journal of Physics*, vol. 8, no. 12, p. 305, 2006.

[21] Jordan, P. and Wigner, E., "Über das Paulische Äquivalenzverbot," *Zeitschrift für Physik A Hadrons and Nuclei*, vol. 47, no. 9, pp. 631–651, 1928.

[22] T. Antal, Z. Rácz, A. Rákos, and G. M. Schütz, "Transport in the xx chain at zero temperature: Emergence of flat magnetization profiles," *Phys. Rev. E*, vol. 59, pp. 4912–4918, May 1999.

[23] U. Schollwöck, "The density-matrix renormalization group," *Rev. Mod. Phys.*, vol. 77, pp. 259–315, Apr 2005.

Acknowledgements

I would like to thank, in alphabetic order, Dr. Fabian Heidrich-Meisner, Andreas Hösl, Dr. Christian Kasztelan, Stephan Langer, Prof. Dr. Ulrich Schöllwock, Sebastian Smerat and Anton Wöllert for their helpful discussions and helping hand in making sense of the data. I would especially like to thank Anton Wöllert and Ulrich Schollwöck for proofreading my thesis several times. And last but not least my soon to be wife, Emily Campling, for making sure this thesis makes sense and the language used actually resembles English.

Declaration for the Master's Thesis

I warrant, that the thesis is my original work and that I have not received outside assistance. Only the sources cited have been used in this draft. Parts that are direct quotes or paraphrases are identified as such.

München, (date) _____ (signature) _____