FACULTY OF ENGINEERING
AND ARCHITECTURE

Cluster Expansion of Thermal States using Tensor Networks

David Devoogdt
Student number: 01608249

Supervisors: Prof. dr. Jutho Haegeman, Prof. Frank Verstraete
Counsellors: Laurens Lootens, Robijn Vanhove, Bram Vanhecke

Master's dissertation submitted in order to obtain the academic degree of
Master of Science in Engineering Physics

Academic year 2020-2021

GHENT
UNIVERSITY

Todo list

Foreword

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula.

Fusce mauris. Vestibulum luctus nibh at lectus. Sed bibendum, nulla a faucibus semper, leo velit ultricies tellus, ac venenatis arcu wisi vel nisl. Vestibulum diam. Aliquam pellentesque, augue quis sagittis posuere, turpis lacus congue quam, in hendrerit risus eros eget felis. Maecenas eget erat in sapien mattis porttitor. Vestibulum porttitor. Nulla facilisi. Sed a turpis eu lacus commodo facilisis. Morbi fringilla, wisi in dignissim interdum, justo lectus sagittis dui, et vehicula libero dui cursus dui. Mauris tempor ligula sed lacus. Duis cursus enim ut augue. Cras ac magna. Cras nulla. Nulla egestas. Curabitur a leo. Quisque egestas wisi eget nunc. Nam feugiat lacus vel est. Curabitur consectetuer.

Suspendisse vel felis. Ut lorem lorem, interdum eu, tincidunt sit amet, laoreet vitae, arcu. Aenean faucibus pede eu ante. Praesent enim elit, rutrum at, molestie non, nonummy vel, nisl. Ut lectus eros, malesuada sit amet, fermentum eu, sodales cursus, magna. Donec eu purus. Quisque vehicula, urna sed ultricies auctor, pede lorem egestas dui, et convallis elit erat sed nulla. Donec luctus. Curabitur et nunc. Aliquam dolor odio, commodo pretium, ultricies non, pharetra in, velit. Integer arcu est, nonummy in, fermentum faucibus, egestas vel, odio.

Sed commodo posuere pede. Mauris ut est. Ut quis purus. Sed ac odio. Sed vehicula hendrerit sem. Duis non odio. Morbi ut dui. Sed accumsan risus eget odio. In hac habitasse platea dictumst. Pellentesque non elit. Fusce sed justo eu urna porta tincidunt. Mauris felis odio, sollicitudin sed, volutpat a, ornare ac, erat. Morbi quis dolor. Donec pellentesque, erat ac sagittis semper, nunc dui lobortis purus, quis congue purus metus ultricies tellus. Proin et quam. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos hymenaeos. Praesent sapien turpis, fermentum vel, eleifend faucibus, vehicula eu, lacus.20

Permission of use on loan

The author(s) gives (give) permission to make this master dissertation available for consultation and to copy parts of this master dissertation for personal use. In all cases of other use, the copyright terms have to be respected, in particular with regard to the obligation to state explicitly the source when quoting results from this master dissertation.

May 24, 2021

date

Abstract

Het abstract is maximum één bladzijde en bevat minstens: a) De informatie die werd vermeld op het titelblad (eigen vorm); b) Een summiere beschrijving van het werk (vijftien à twintig regels); c) Eventueel: drie tot vijf goed gekozen trefwoorden die het onderwerp best omschrijven.

Extended Abstract

De extended abstract heeft een standaardlengte van minimaal twee bladzijden, met een maximum van zes bladzijden.

Contents

Chapter 1

Introduction

> There is nothing new to be discovered in physics now. All that remains is more and more precise measurement.
>
> Lord Kelvin, 1900

1.1 Introduction

In 2015, there were about 5.6 million known physics papers in literature. At the current rate, this number doubles every 18.7 years [1]. Despite this enormous body of literature, there are a lot of things which are not completely understood. Some examples include a self-consistent theory of quantum gravity, the need for dark energy and matter in cosmology, the arrow of time, the matter-antimatter asymmetry. There even is no interpretation of quantum mechanics where everyone agrees upon.

But certainly not all open problems have to do with 'new' physics. In many areas of physics, computing the implications of relatively simple laws becomes exceedingly difficult for many particles. Of historical importance is the three-body problem, describing the trajectory of 3 gravitational bodies such as the earth, moon and sun. The general case is not solved, despite developments over the last 300 years.

In reality, the real challenge is to model the macroscopic properties of quantum many-body system with around 10^{23} particles. Needless to say, this not an easy task at all. Finding good and computable approximations is of primary importance in the fields of quantum chemistry, condensed matter physics, and materials science.

In computational chemistry, the many-body problem is tackled with methods which fall in one of the following categories: (post-) Hartree-Fock methods, density functional theory (DFT) and force-field methods. While they have many vind bron en voorbeelden

applications , these methods are not fully able to capture all the properties of the so called strongly correlated matter.

Examples of phase of strongly correlated matter which are not yet understood include high-T superconductors, topological ordered phases, quantum spin liquids [2]. There exist different methods to investigate these exciting materials. A very limited number of models is quantum integrable, meaning they can be solved in a non pertubative way. Also, some properties of models near criticality can be determined exactly with conformal field theory (CFT). But for some systems, we can only simulate the behaviour with numerical techniques. To make progress, new fast and accurate numerical methods are needed, because exact diagonalisation becomes unfeasible for large systems.

Some examples of such numerical techniques, which will not be discussed, are: Dynamical Mean Field (DMFT) / Dynamical Cluster Approximation (DCA), Series expansion, Density Matrix Embedding Theory (DMET), Fixed-node Monte Carlo, Diagrammatic Monte Carlo, Variational Monte Carlo, Functional renormalization group (FRG) and Coupled-cluster methods. [3]

In this thesis, a technique is proposed that builds on the broad field of tensor networks.

1.2 Tensor networks

This is often referred to as the curse of dimensionality. The size of the Hilbert space of quantum states grows exponentially fast. This prevents an effecient description of all possible quantum states.

area law+picture

sign problem monte carlo, ...

write about tensor networks

Chapter 2

Tensor networks

2.1 Introduction

Tensor networks form a vast subject. This chapter gives a very brief introduction into the field of tensornetworks. First, the graphical notation used to represent these tensors is explained. The most relevant kinds of tensor networks for this dissertation are introduced

A practical introduction is given how to manipulate these networks. Next, a selected number of MPS algorithms is given. The focus will be to understand the intuition behind them, but not to provide a mathematical rigorous treatment.

The next section focusses on the contraction of infinite dimensional 2D networks. This is chosen both to improve the understanding of tensor networks through example, and because one of the algorithms, VUMPS, will be widely used in to calculate 2D phase transition later on.

2.2 Tensor networks

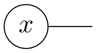
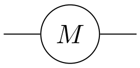
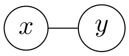
2.2.1 Graphical notation

Before explaing tensor networks, some graphical notation should be introduced. This really is a way to conveniently write vectors, matrices an in general tensors without the need to introduce many labels. A tensor T is represented by a circle with a number of external legs, according to the number of external indices. Connected legs are summed. Some examples are shown in table 2.1. Every leg which is connected to multiple tensors, is contracted.

2.2.2 Representing a quantum state

Tensor network come in many shapes and forms. Tensor networks are really used to represent a tensor with many legs. A general quantum state with N

Table 2.1: Caption

conventional	Einstein	tensor notation
$\vec{x}$	x_α	
M	$M_{\alpha\beta}$	
$\vec{x}\cdot\vec{y}$	$x_\alpha y_\alpha$	

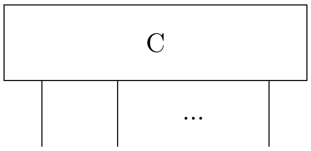


Figure 2.1: Caption

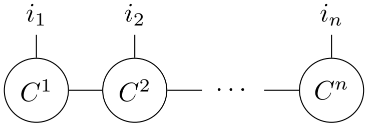
sites can be described in a given basis $|i\rangle$ in the following way:

$$|\Psi\rangle = \sum_{i_1 i_2 \cdots i_n} C^{i_1 i_2 \cdots i_n} |i_1\rangle \otimes |i_2\rangle \otimes \cdots \otimes |i_n\rangle \tag{2.1}$$

Here the tensor C holds all the information of the quantum state. The graphical representation can be seen in fig. 2.1.

This requires an exponential number d^n of coefficients C where d is the dimensions of basis $|i\rangle$. In order to make the problem tractable, the following form is proposed as wave function:

$$C^{i_1 i_2 \cdots i_n} = {C^1}^{i_1}_{\alpha_1} {C^2}^{i_2}_{\alpha_1 \alpha_2} \cdots {C^n}^{i_n}_{\alpha_{n-1}} \tag{2.2}$$

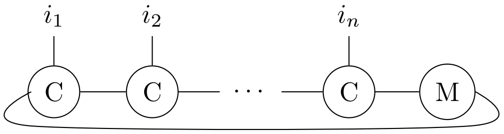


(2.3)

Where summation over shared indices is implied. It is always possible to find such a representation by means of matrix decomposition (see section 2.3). The summation over α_i are called a virtual bond and their dimension is denoted by χ. At this point, this is not yet an improvement because the bond dimension needs to be exponentially large in order to represent the tensor C exactly.

Explicit translational invariance is given by tensor $C^i_{\alpha\beta}$ that don't depend on the location. The chain is closed by a matrix M which contains the boundary condiditions. Setting $\alpha_n = \alpha_0$. We can now write this as a Trace over matrix products:

$$|\Psi\rangle = \mathrm{Tr}\big(C^{i_1}C^{i_2}\cdots C^{i_n}M\big)\,|i_1\rangle\otimes|i_2\rangle\otimes\cdots\otimes|i_n\rangle \tag{2.4}$$



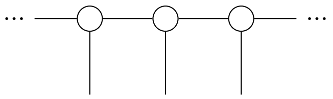
(2.5)

2.2.3 Classification

Tensor networks come in many shapes and many forms. During this thesis, we will mainly encounter the types explained here.

2.2.3.1 MPS

A general MPS is of the form eq. (2.5). For an infinite extended chain with translation invariance, it is logical to assume an unit cell:



(2.6)

This will be the form we will work with in this thesis. Of course, larger unit cells are also possible.

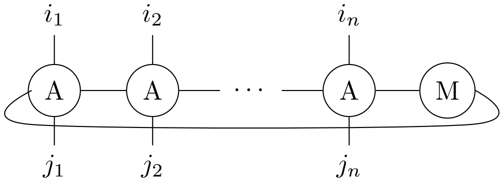
MPS are dense. This means that (with a bond dimension exponential in the system size) the MPS can represent every state in the full Hilbert space. [2]. However, the low energy states can typically be represented by a much lower bond dimension.

This can be explain in terms of the so called 'area law': the entropy low energy states often scale with the boundary area of volume. MPS exactly follow this area law.

2.2.3.2 MPO

Similarly to an MPS, a Matrix Product Operator (MPO) is of the following form:

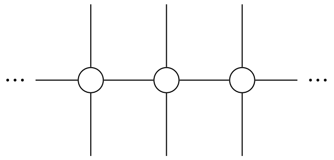
$$\begin{aligned}\hat{O} = \sum & Tr(A^{i_1 j_1}A^{i_2 j_2}\cdots A^{i_n j_n}M)\\ & \times |i_1\rangle\langle j_1|\otimes|i_2\rangle\langle j_2|\otimes\cdots\otimes|i_n\rangle\langle j_n|\end{aligned}$$



(2.7)

The matrix M contains the boundary conditions of the operator. Its clear that this can again be made translation invariant. Acting with an MPS on a MPO $\hat{O}\left|\Psi\right\rangle$ again results in a MPS, as expected.

An uniform MPO is denoted by:



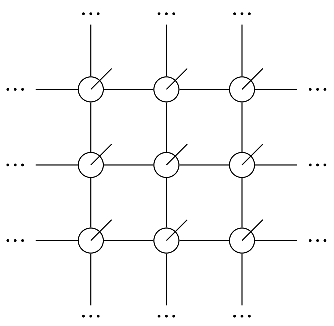
(2.8)

This can also be reinterpreted as an MPS with physical bond dimension D^2, by grouping the i and j indices together per site.

2.2.3.2.1 Examples of MPO hamiltonians ??

2.2.3.3 PEPS

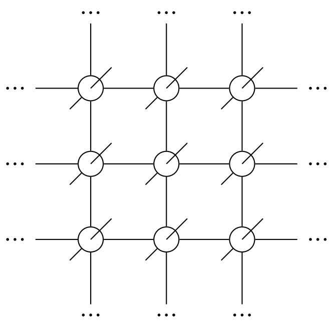
Projected entangled pair states (PEPS) are the 2D equivalent of MPS. In the visualisation, the physical indices come out of the plane.



(2.9)

2.2.3.4 PEPO

The operator version of PEPS is called PEPO. The depiction looks as follows:



(2.10)

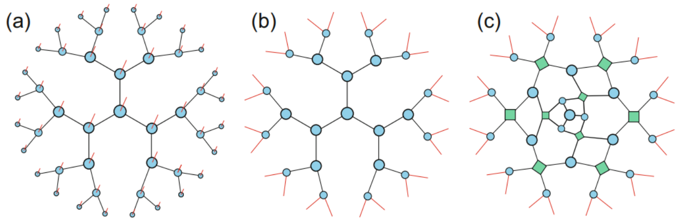


Figure 2.2: (a) and (b) two different TTNSs and (c) MERA. Figure taken from [4].

2.2.3.5 Others

To show there exist many more tensor network beside the ones treated in this thesis, 2 examples are shown in fig. 2.2.

Two Tree tensor networks (TTNs) are shown. The red indices are physical. MERA (c) is also shown. These tensor network are specific because they can be contracted exactly, as opposed to for instance PEPS.

2.3 Tensor network manipulations

This section serves as an introduction of tensor network manipulations. The overview mainly focusses on MPS/MPO networks, but most of the oprations translate to the 2D case.

The MPS's are processed by transforming the tensor into a matrix, performing some matrix calculations and casting it back into its original form. In this way, the standard methods from linear algebra can be used. This section gives some examples how this is done in practice:

2.3.1 Basics

2.3.1.1 Grouping legs

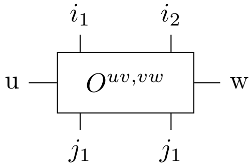
One of the most basic manipulations is to group some legs of a tensor into one leg:

$$\begin{aligned} T^{i_1 i_2 j_1 j_2} &= \begin{array}{c} i_1 \qquad i_2 \\ | \qquad\ \ | \\ \boxed{\qquad T \qquad} \\ | \qquad\ \ | \\ j_1 \qquad j_1 \end{array} \\ &\cong \ (i_1 j_1) - \boxed{\qquad T \qquad} - (i_2 j_2) \\ &= T^{(i_1 j_1)(i_2 j_2)} \end{aligned} \tag{2.11}$$

The dimension of the new leg is the product of the dimension of the individual legs. Contracting 2 merged legs with 2 merged legs is exactly the same as contracting them separately. The both The 4 leg tensor and matrix contain exactly the same information. Manipulating this in memory requires both permute and reshape commands. This requires some time, the internal representation of the matrix changes.

2.3.1.2 Decomposition

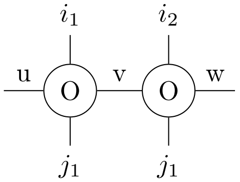
The grouping above can be applied to decompose a tensor into 2 tensor with matrix techniques. An example, which will be needed later on, is give here.



$$= O^{i_1 i_2 j_1 j_2}_{\alpha_u \gamma_w}$$

$$\begin{aligned} &\cong O^{uw}_{(\alpha_u i_1 j_1)(\gamma_w i_2 j_2)} \\ &= O^{uv}_{(\alpha_u i_1 j_1)\alpha_v} O^{vw}_{\alpha_v(\alpha_w i_2 j_2)} \end{aligned} \tag{2.12}$$

$$\cong$$



The indices U,V and W represent blocks indices. Step 2 reshapes and groups the indices on to one index on the left and one on the right. The dimension of this index is the product of the separate dimensions. Step 3 decomposes the matrix into a product of 2 matrices. The last step transforms the indices back to separate legs.

For an exact representation, the bond dimension of virtual level v is:

$$\dim v = \min(\dim u, \dim v) + 2 \dim i \tag{2.13}$$

Many matrix decompositions exist. Some useful examples here are SVD decomposition, eigenvalue decomposition, QR, $\cdots$.

2.3.1.3 Truncation

The above procedure also gives a way to truncate the bond dimension: first take together 2 neighbouring sites, then perform an SVD $T = U \Sigma V^\dagger$. Split as follows:

$$T = \begin{bmatrix} U_1 & U_2 \end{bmatrix} \begin{bmatrix} \Sigma_1 & 0 \\ 0 & \Sigma_2 \end{bmatrix} \begin{bmatrix} V_1 & V_2 \end{bmatrix}^\dagger \tag{2.14}$$

Where Σ_1 contains the n largest singular values. Then $\hat{T} = U_1\Sigma_1V_1^\dagger$ is the best rank n approximation to the original tensor.

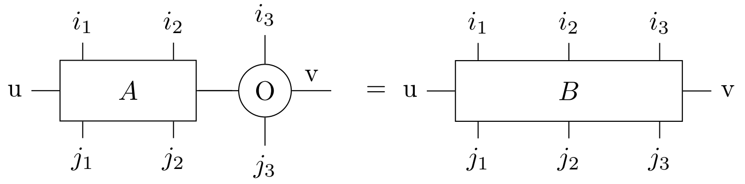
$$min_{rank(A)\leq n}\|T - A\| = \|T - \hat{T}\| \tag{2.15}$$

2.3.1.4 Virtual levels

In the previous example, the levels were indicate with a block index or virtual level. The idea is to create seperate the contraction into blocks. This is completely analogous to matrix block multipliction. This wil be a more natural form to represent the algorithm. Of course, one can easily switch between block representation and the full one.

2.3.1.5 Inversion

Suppose we want to find a MPO O for given tensors A and B such that the following holds:



(2.16)

Again, the indices can be taken toghether in the following way: $\alpha = (ui_1j_1i_2j_2)$ and $\beta = (i_3j_3v)$:

$$A_{\alpha\gamma}O_{\gamma\beta} = B_{\alpha\beta} \tag{2.17}$$

This a a standard matrix equation and can hence be solved with linear algebra packages. Note that it is not necessary to calculate A^{-1} to obtain the solution. Linear solver are generally much faster. As this is one of the core problems to solve both in 1D and 2D, this will be discussed in detail in section 5.2.

2.3.1.6 Contraction order

Tensor network diagram determine unambigously the result of a contraction. This does not mean all contraction orders are equivalent. The number of operations needed to contract 2 vectors is D. Each additional leg multiplies the number of operations by the bond dimension. Two differen contraction orders are shown in fig. 2.3.

2.4 MPS algorithms

This section, and the following one, will introduce some different tensor network algorithms. The gaol is to provide an intuitive expanation how these algorithm

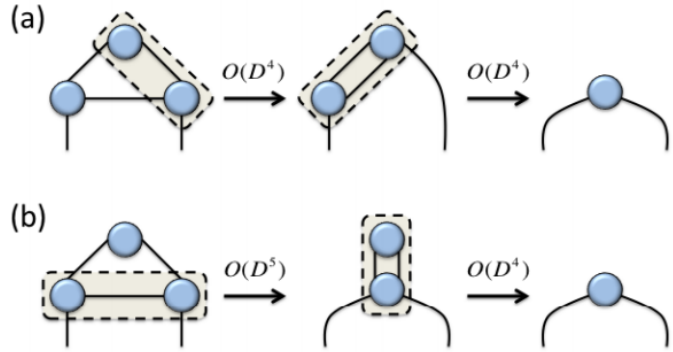


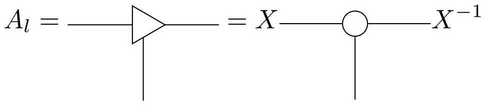
Figure 2.3: (a) Contraction of 3 tensors in $O(D^4)$ time (b) Contraction of same tensors in $O(D^5)$ time. Figure taken from [2].

work. For rigorous derivations and mathematical details, other sources can be read.

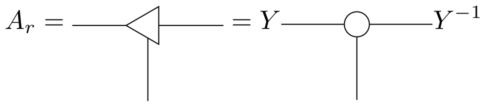
2.4.1 Canonical form

A translation invariant MPS has the following form shown in eq. (2.6)

It can be easily seen that inserting XX^{-1} on each bond doesn't change the contracted tensor. This is called a guage transformation.

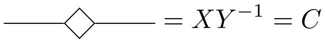


(2.18)



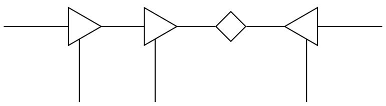
(2.19)

and



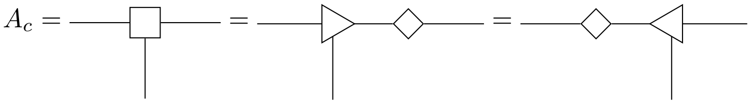
(2.20)

Then eq. (2.6) can be written as follows:



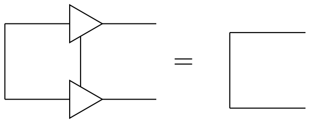
(2.21)

Introducing one more tensor:

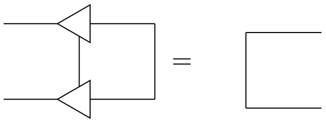


(2.22)

At the moment the matrices X and Y are not yet defined. To bring an MPS A in its canonical form, the following choice is made



$= I$ (2.23a)



$= I$ (2.23b)

The tensors below are the conjugated version of the tensors above. There is still some freedom: an unitary gauge transformation can still be applied. With this choice, it is possible to bring C in diagonal form.

2.4.1.1 Entropy

The canonical form above is very convenient to calculate the entropy, which is given by

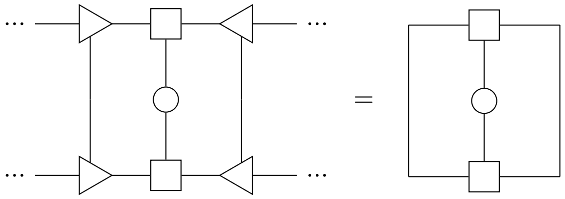
$$S = -\sum_i \alpha_i^2 \log\left(\alpha_i^2\right) \tag{2.24}$$

where α_i are the singular values of C.

2.4.1.2 expectation value

??

One properties of MPS is that expectation values can be calculated exactly. suppose we have an operator $\hat{O}$ acting on a single site of the MPS: $\left\langle \Psi \middle| \hat{O} \middle| \Psi \right\rangle$



(2.25)

where we made use of eq. (2.23). Similar techniques allow to calculate correlatetion functions, energies, ...

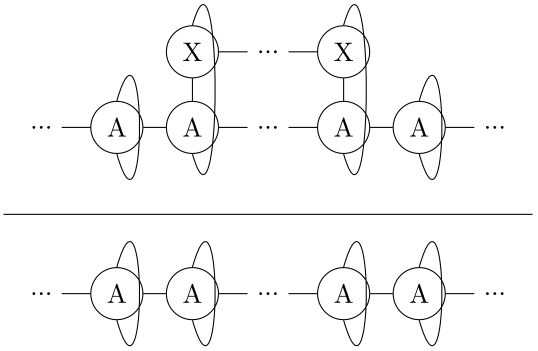
2.5 tensor network contraction

The contraction in this section is somewhat different. The MPO with the same $S_n = (i_1 i_2 \cdots i_n) = (j_1 j_2 \cdots j_n)$ represents the probability of finding the system in state S_n. This will be needed in section 3.4.1. On a patch, an operator X is applied. This could be a measurement.

2.5.1 1D problem

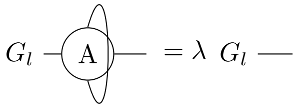
Suppose that the there is an MPO representation A that gives the probability of a state, and an operator X which performs a measurement over a number of sites. The expecatation value $\langle X \rangle$ is given by:

$\langle X \rangle =$

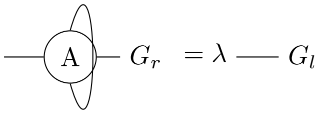


(2.26)

In the thermodynamic limit there are an infinity number of A to the left and the right. This can be simulated by taking the left and right fixed points of the traced MPO A corresponding to the largest eigenvector λ.



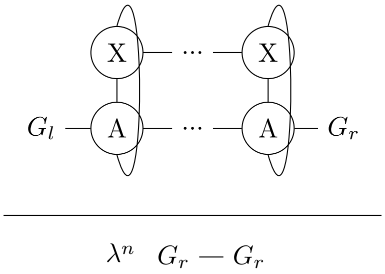
(2.27)



(2.28)

Equation eq. (2.26) can now be easily caclulated:

$\langle X \rangle =$

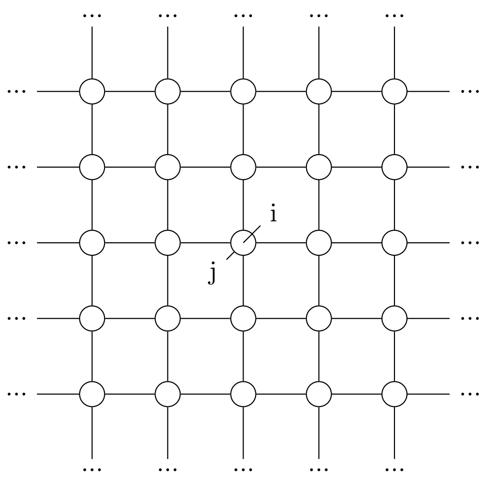


(2.29)

2.5.2 2D problem

PEPS contraction concerns a similar problem, but in 2D. Here, instead of already applying operator X, the (one site) reduced density matrix $\rho_{i,j}$ is computed:

$\rho_{i,j} =$



(2.30)

This problem is much harder than in 1D. For instance, the related problem of calculating the expecatation value of on operator, similar to **??** in 1D, is #P-Hard [2].

2.5.3 Overview methods

As this is a difficult task, and often the bottleneck in simulations, many algorithms exist to perform this. The classification is taken from [5]

2.5.3.1 Real-space renormalization-group methods

The general idea is to course grain the tensor network. Of the many methods, historically the first one was Tensor renormalization group (TRG) [6], shown in fig. 2.4. In the first step i, the tensor are split with a SVD in 2 different ways, depending on the position on the lattice. ii recombines 4 halves of the decomposition into 1 new tensor. The result is a rotated grid, with side length $\sqrt{2}$. The bond dimension is truncated during the SVD step.

Many other variants exist, such as Tensor Network Renormalization (TNR), which wont be discussed here.

2.5.3.2 Corner transfer matrix methods

Another method goes by the name Corner transfer matrix renormalisation group (CTMRG), as descirbed in [7]. The full network is approximated by 4 corner matrices and 4 C and 4 half row transfer matrices T as shown in fig. 2.5a. The idea is to find matrices such that inserting a row and truncating it again results in the same tensors. This is shown in fig. 2.5b. First, a new row is inserted.

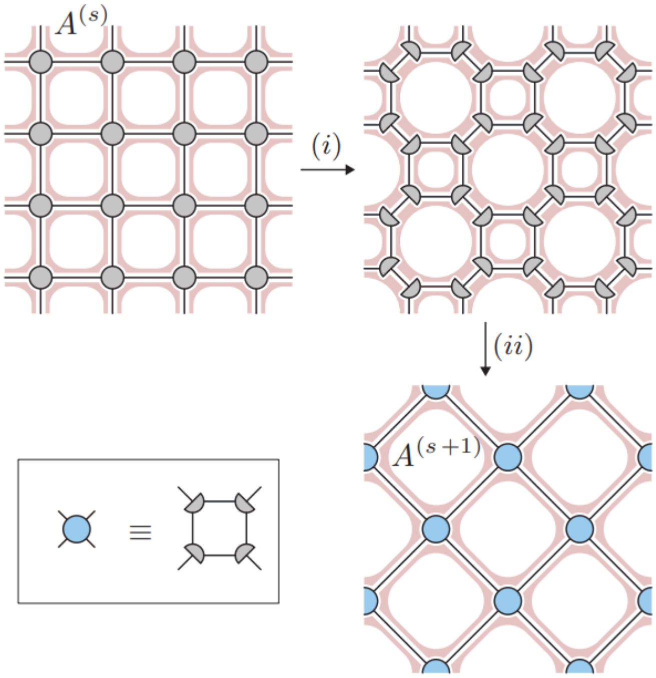
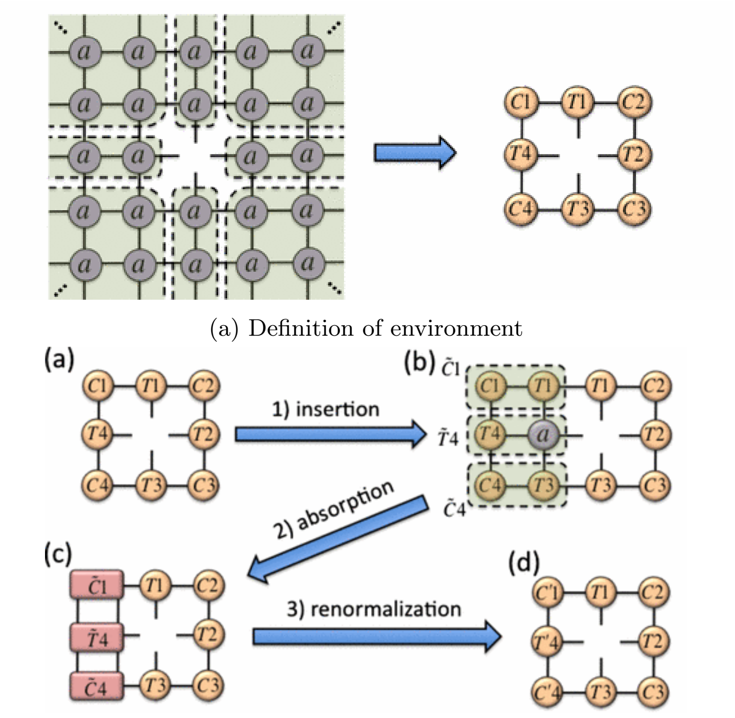


Figure 2.4: Steps in TRG procedure. Figure taken from [6].



(a) Definition of environment

(b) 3 steps of CTMRG

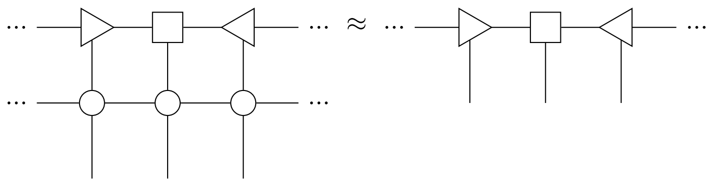
Figure 2.5: Figures adapted from [7]

From fig. 2.5a it can be seen that this should not change the environment. Some of the tensors are taken together in step 2. As a final step, the tensor are suitably truncated. Once this cycle converges, the environment is known.

Note that many important details are not written down here, and this is merely to give some intuition about tensor network contraction algorithms.

2.5.3.3 Boundary methods

The goal of these methods is to find an MPS fixed point for the infinite lattice:



(2.31)

2.5.3.3.1 Time-evolving block decimation Time-evolving block decimation was introduced as a method to calculate thermal states (this will also be

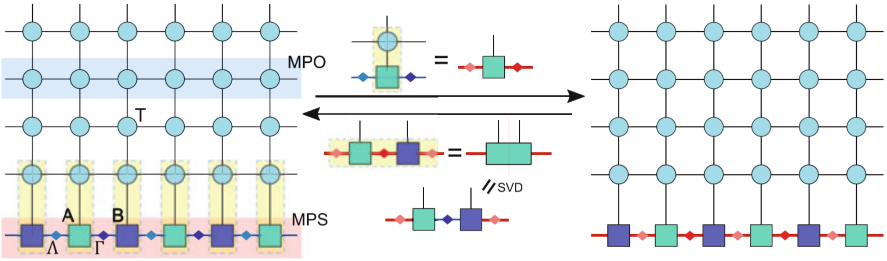


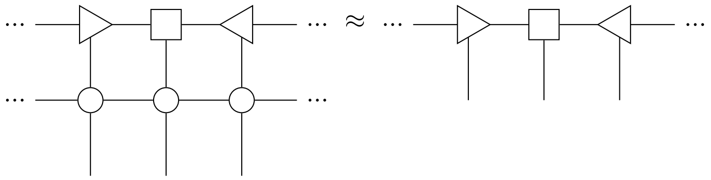
Figure 2.6: Figure taken from [4].

encountered later on in fig. 2.6). It can also be used as a power method for contraction. The idea is shown in fig. 2.6. The boundary MPS is applied, and afterwards truncated. When this procedure is applied enough times, a fixed point is reached.

2.5.4 Vumps

The purpuse of this section is to give some intuition on the variational uniform Matrix Product State (VUMPS) algorithm, which will be used later on in this thesis.

The goal is to find a MPS layer for the MPO such that:

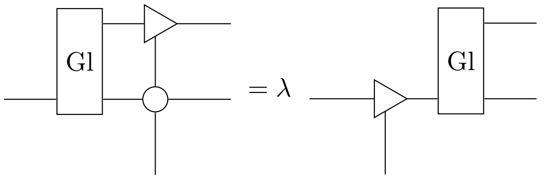


(2.32)

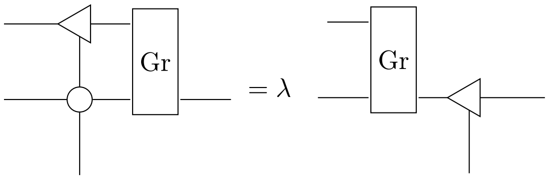
This is very similar to eq. (2.27). The expression holds approximately, because the MPS on the left hand side has a larger bond dimension than on the right hand side.

2.5.4.1 The equations

Suppose there are tensor which fullfil the conditions stated below:

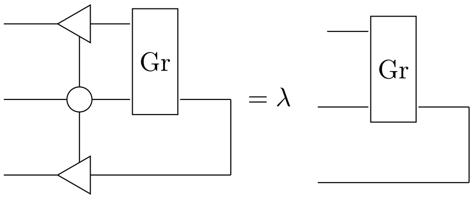


(2.33a)



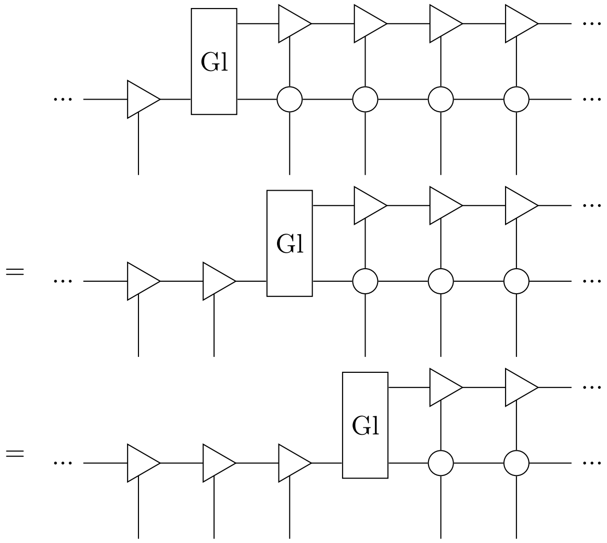
(2.33b)

These eigentensor equations are solved in pratice in a slightly different manner:



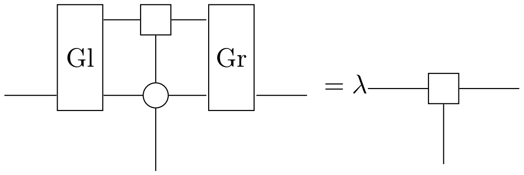
(2.34)

Wher the Ar tensor was connected from below on both sides and eq. (2.23) was used on the right hand side. The blocks in eq. (2.33) form a zipper from the left and right respectively. Each application brings down one of the MPS tensors in the following way.



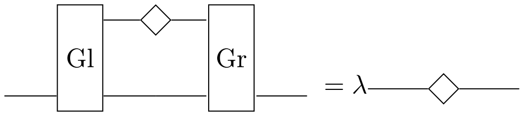
(2.35)

The left and right zipper can now move towards each other, until they meet at the center. In order to become the same MPS as before the application to the MPO (eq. (2.32)), one more condition is needed:



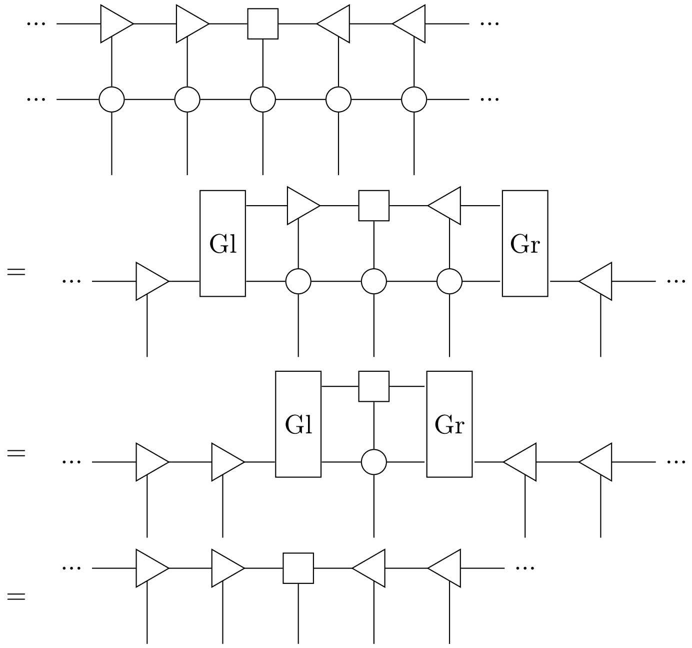
(2.36)

This completely determines the problem, but one more equation is used to solve the problem. Combining on of the equations eq. (2.33), the definitipn of A_c eq. (2.36) and one of eq. (2.23) gives C:



(2.37)

Then eq. (2.32) is solved:



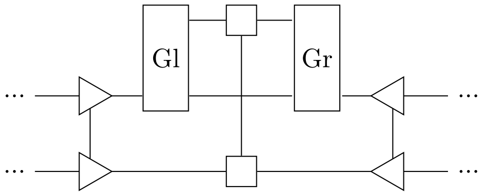
(2.38)

check lambdas in literature

Contracting a 2D tensor network is thus reduced to solving the eq. (2.22), eq. (2.33), eq. (2.36) and eq. (2.37) simultaneously. Inspection of the equations show that the following cycle needs to be solved:

- $A_c, C \to A_l, A_r$ eq. (2.22)
- $A_l, A_r \to G_l, G_r$ eq. (2.33)
- $G_l, G_r \to Ac, C$ eq. (2.36) and eq. (2.37)

The calculated environment can now be used to solve the original problem. Due to symmetry, the same MPS can be applied from below. eq. (2.37) now becomes:



(2.39)

2.5.4.2 The derivation

While the above derivation is reasonable, it is not very rigorous. The algorithm finds its origins in tangent space methods, such as explained in [8].

Not every state in the many body Hillbert space can be represented by an MPS. For a

By carefully constructing the tangent space and making use of the available gauge freedom, a compact expression can be found for the tangent space projector $\mathcal{P}_A$. This projects a state from the many body Hillbert space to the tangent space of an MPS A. For an optimal MPS approximation A, the error made in the approximation should be orthogonal to the tangent space. For eq. (2.32) this means that the application of the projection of the error on the tangent space should be zero, i.e. the MPS is at a variational minimum. [5]

Applying the projector $\mathcal{P}_A$ to eq. (2.32) exactly gives rise to the equations stated earlier.

2.5.4.3 Multisite

2.5.4.4 Calculating the MPS from below

2.6 Conclusion

Tensor networks were introduced from a practical point of view: what different kinds exist, how to reason and manipulate tensornetwork. Some useful algorithms are discussed. The canonical form of an MPS allows to contract 1D networks fast

succeses MPS

The diffuculty of contracting an 2D tensor network was touched upon, and some powerfull algorithms are introduced, in particular VUMPS.

finish this

Chapter 3

Strongly correlated matter

3.1 Introduction

This chapter talks about strongly correlated matter. First, the focus will be on the different phases of matter, and phase transition between them. It turns out that there are a lot of interesting aspects related to phase transitions, such as universality. This principle says that the phsyics of many models at criticality can be captured by a limited numbers of classes, each charactered by a set of critical exponents. From a simulation point of view, the finite size scaling method is introduced to capture these properties while using a relative small grid.

In a second sections, some models are introduced, together with some known properties of these models. These models will be used to test the cluster expansions introduced later.

In a last section, an overview of operator exponentials is given. These are used both in statistical mechanics as in time evolution of a quantum state. The current methods for approximating these exponentials within the tensor network framework are discussed. The cluster expansions from this thesis, will also be a tensor network method to simulate the operator exponentials.

3.2 Phases and Criticality

3.2.1 Phases of matter

An important area of research is the study of the different phases of (quantum) matter. A phase is a state of matter in which the macroscopic physical properties of the substance are uniform on a macroscopic length scale. These phase can be measured by thermodynamic function, i.e. by function of a few macroscopic parameters. [9]. More precisely, for a given phase the properties vary as an analytic function of the macroscopic variables.

Interesting physics happens at the boundary between 2 or more distinc phases. The phase transitions were classified by Ehrenfest [10], who looked at the free energy across the phase boundary. If the free energy shows a discontinuity, it is called first order (or discontinuous) phase transition. Similarly, if the derivitive shows a discontinuity, it is called second order (or continuous). Higher order phase transitions are possible, and there are even examples of infinite order transitions, such as the BKT transition.

3.2.2 Symmetry breaking

Sometimes, but not always, a phase transition is related to spontaneous symmetry breaking. A state $|\Psi\rangle$ is said to be symmetric under a unitary transformation U if the state only changes by a phase factor: $\hat{U}\,|\Psi\rangle = e^{i\phi}\,|\Psi\rangle$. A hamiltonian posesses a symmetry if it commutes with U: $[H, U] = 0$ [11]. A remarkable fact is that many ground states are not invariant under a symmetry U of the hamiltonian.

For phase transitions associated with a broken symmetry, one can define an order parameter. This parameter evalutes to 0 for the symmetric phase, but not for the sponteous broken phase.

In continuous or second-order phase transitions the order parameter increases continuously from zero as the critical temperature is traversed. The entropy also changes continuously. On the other hand, the correlation length and related energy scales diverge at the critical temperature. In fact, at the critical temperature of a second-order phase transition, scale invariance systems become scale-invariant, in the sense that physical properties no longer depend on the length (or energy) scale at which they are probed. Many symmetry-breaking phase transitions are second-order, with the onsets of superfluidity, (anti)ferromagnetism and many phases of liquid crystals as famous examples.[11]

3.2.3 Universality

Universality looks at the behaviour of the system near a continuous phase transition. These can be discribed well by so called power laws. For classical phase transitions (driven by temperature) near critical temperature T_c, observables a_i depend in the following way on the reduced temperature $t = \frac{T-T_c}{T_c}$: $a_i(t) \sim t^{\alpha_i}$. One would expect that the set of critical exponents α_i depends on the precise form of the hamiltonian of the system, but it turns out these exponents can be captured by a limited numer of universality classes. This means that the physics near criticality is completely understood once it is understood for one member of the class.

3.2.4 Critical exponents for spin systems

The following table defines some of the critical exponents for the Ising system.

The 2 point correlation function is defined as $f(x, y) = \langle m(x)m(y)\rangle - \langle m(x)\rangle \langle m(y)\rangle$. At larger distances this decays exponentially fast (see section 3.2.6) $f(x, y) = e^{-\frac{|x-y|}{\xi}}$, where ξ is the correlation length.

Symbol	name
m	magnetisation
ξ	correlation length
g	external field
t	reduced temperature
d	dimension

for the ordered phase, the following relations hold: $m\ |t|^{\beta}$, $\xi(t) \approx |t|^{-\nu}$. At the critical temperature near a quantum phase transition $m \approx |g - g_c|^{\frac{1}{\delta}}$.

3.2.5 Finite size scaling

Phase transitions only occur for systems with an infinite number degrees of freedom. This poses a problem, as in for instance Monte Carlo simulations only finite grids can be simulated. One computational expensive way to extract the properties in the thermodynamic limit is by making the grid increasingly bigger untill the properties have converged. Fisher's insight was that the properties could also be extrapolated from different finite size calculations by making the following assumption: near a critical point, every thermodynamic properties scales as an universal function of L/ξ, with L the size of the system and ξ the correlation length.

Define $t = \frac{T-T_c}{T_c}$. The mathematical formulation is as follows:

$$A(T, L) = L^{\kappa/\nu} f_A(tL^{1/\nu}) \tag{3.1}$$

This holds for t small (near critical point) and L sufficient large compared to the lattice spacing. The exponents can be fitted by plotting $A(T, L)L^{-\kappa/\nu}$ as a function of $tL^{1/\nu}$ for different sizes L and temperature t. For the correct critical exponents and critical temperature, all the points should collapse to one single graph.

From the ansatz eq. (3.1), other ways can be derived to determine certain coefficients.

3.2.5.1 Finite size scaling for MPS

The finite size scaling for MPS is somewhat different. In [12], it is argued that δ can take the place of $1/L$.

Suppose λ_i are the eigenvalues of eq. (2.34) order from largest real part to smallest. Then

$$\epsilon_i = -\log(|\lambda_i|) \tag{3.2}$$

Then

$$\delta = -\sum_i c_i \epsilon_i \quad \sum_i c_i = 0 \tag{3.3}$$

The intuition behind this is as follows: for a MPS approximation, the gaps in the transfer spectrum go always to zero for sufficient large bond dimension. The distance between these gaps is thus a measure for the system size.

3.2.5.2 subleading corrections

In [13], it is argued that the form proposed in eq. (3.1) does also not fully take into account the finite size effects. Subleading correction could be introduced as follows:

$$A(T, L) = L^{\kappa/\nu}(1 + cL^{-\omega})f_A(tL^{1/\nu} - dL^{-\phi/\nu}) \tag{3.4}$$

Indeed, this reduces to eq. (3.1) for sufficient large L. Because there is always data need around the critical point, the original procedure could be biased and result in wrong parameters. On the other hand, introducing extra parameters can lead to overfitting, again not improving the result.

3.2.6 CFT

One of the tools to derive some properties of phase transitions is Conformal Field Theory (CFT). CFT is a quantum field theory, which obeys an additional rule: the physics remains invariant under a conformal transformation. The exact form of these transformation is:

$$g'_{\mu\nu}(x') = \Lambda(x)g_{\mu\nu}(x) \tag{3.5}$$

In 2D, the shape of the correlation functions can be determined exactly, and indeed correspond to the form from previous section.

One of the properties characterising a conformal field theory is the central charge c, which can only take a discrete number of values. In the case of unitary systems with $c \leq 1$, this has turned out to give a complete classification of possible two dimensional critical behavior. [14]

For the 2D Ising model, the central charge is $c = 1/2$. A scaling relation used in the results chapter is [15]

$$Le^{6S(T,L)/c} \sim \xi \tag{3.6}$$

3.2.7 Quantum phase transitions

A traditional 2nd order phase transition is driven by a change in temperature. Quantum phase transitions on the other hand happen at zero temperature under influence of another parameter g of the model. At finite temperature, 2 things can happen: either there is a line connecting a classical 2nd order phase transition to the quantum phase transition, or the phase transition disappears at finite temperature [16].

3.3 Models

The goal of numerical techniques is to simulate the physics of real world systems. These are, to some extend, captured by different models. Models are a simplified mathematical description that captures some relevant physics of more complicated systems. This section introduces some specific models, their

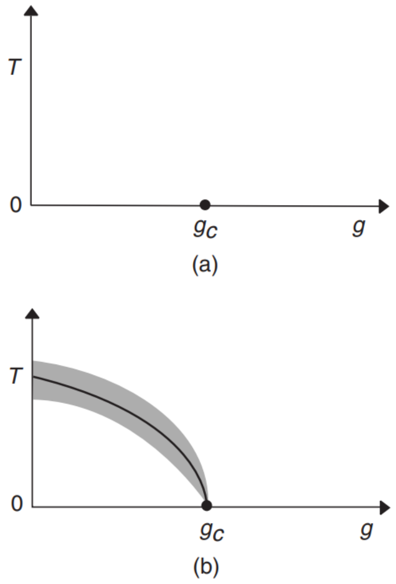


Figure 3.1: Two possible phase diagrams of a system near a quantum phase transition. In both cases there is a quantum critical point at $g = g_c$ and $T = 0$. In (b), there is a line of $T > 0$ second-order phase transitions terminating at the quantum critical point. The theory of phase transitions in classical systems driven by thermal fluctuations can be applied within the shaded region of (b). Figure and caption taken from [16].

relevance and some properties. These models will be used later te benchmark the developed tensor network expansion.

3.3.1 Ising model

The prototypical example of a model in the field of strongly correlated matter is the Ising model. It was first introduced 1925 by Ernest Ising, as a model to capture ferromagnetism. He proved that for a linear chain, there is no phase transition at finite temperature. He wrongly concluded that this would also be the case in higher dimensions, but it turned out t be one of the deepest and far-reaching problems in 20th century [17].

The Ising model, in essence, assigns an energy contribution to neighbouring spins. These spins sit on a fixed position on a chain (1D) or lattice (2D/3D/...). In classical Ising, the operators in the Hamiltonian all commute with each other. An energy is assign between neighbouring spins and possible and energy for alignment with an external magnitic field in the same direction. In quantum Ising model, a transversal field is added. Often, the particles on the grid are spin 1/2 particles, but of course other particles are possible.

q potts,...

Many generalizations exist for the Ising model.

3.3.1.1 Classical Ising

The classical ising model is given by the following hamiltonian:

$$H = -J\left(\sum_{<ij>} \sigma_i \sigma_j + h \sum_i \sigma_i\right) \tag{3.7}$$

where $< ij >$ runs over all neighbouring lattice sites. The possible values of σ depends on the spin dimension. For spin 1/2 lattices $\sigma \in -1, +1$. g encodes the interaction strength of the external magnetic field.

The sign J determines the low temperature ground state. A positive J will tend to allign all neighbouring spins at low temperature. This is often called ferromagnetic, because all the aligned spins cause a macroscopic magnitisation. On the other hand, a negative J causes neighbouring spins to have an opposite sign.

Depending on the sign of the longitudinal field h, the spins tend to align or anti align with this external field. This lifts the degeneracy of the groundstate.

blabla

3.3.1.1.1 1D The classical 1D model was solved analytically by Lens.

3.3.1.1.2 2D In 2D, it becomes important to define the lattice. Here, and in the simulations, we will consider a square lattice. This model was famously solved by Lars Onsager in 1944, by using the transfer matrix method. In 2 dimensions, the Ising model has a phase transition at finite temperature. The critical temperature is $T_c = \frac{2J}{Tln(\sqrt{2}+1)}$.

Only the $h = 0$ case the is solved analytically. For higher dimensions, no analytical solution is known. For these cases, we need to use numerical techniques if we want to understand the behaviour of these models.

On different lattices, interesting thins can happen. For instance, the groundstate of an antiferromagnet on a triangular lattice is not obvious to determine. The spins tend to antiallign, but at least 2 of 3 spins on the corner of a triangle have to align. Remarkably, as will be explained in section 3.2, the physics at the phase transition does remain invariant when the lattice is changed.

3.3.1.2 Quantum Ising

As we all know, the real world behaves, certainly at small length and time scales, quantum mechanically. Therefore, it is important to understand how the quantum Ising model differs from the classical model. In the quantum Ising model, the operators no longer commute with each other. An example is the transversal Ising model given by the following hamiltonian:

$$\hat{H} = -J\left(\sum_{<ij>}\sigma_i^x\sigma_j^x + g\sum_i\sigma_i^z\right) \tag{3.8}$$

In the case that $g = 0$, this is the classical Ising model (in the $h = 0$ case).

3.3.1.2.1 1D Different to the classical case, the 1D model already contains a phase transition.

3.3.1.2.2 2D

3.3.2 Heisenberg

The heisenberg model is given by:

$$\hat{H} = -\left(\sum_{<ij>} J_x\sigma_i^z\sigma_j^z + J_y\sigma_i^y\sigma_j^y + J_z\sigma_i^z\sigma_j^z + h\sum_i\sigma_i^z\right) \tag{3.9}$$

These models have different names depending on the values of J_α with $\alpha = x, y, z$. $J_x = J_y \neq J_z = \Delta$ is called the XXZ model.

3.3.3 Random

It's also possible to construct random hamiltonians.

in basis: hermitian H

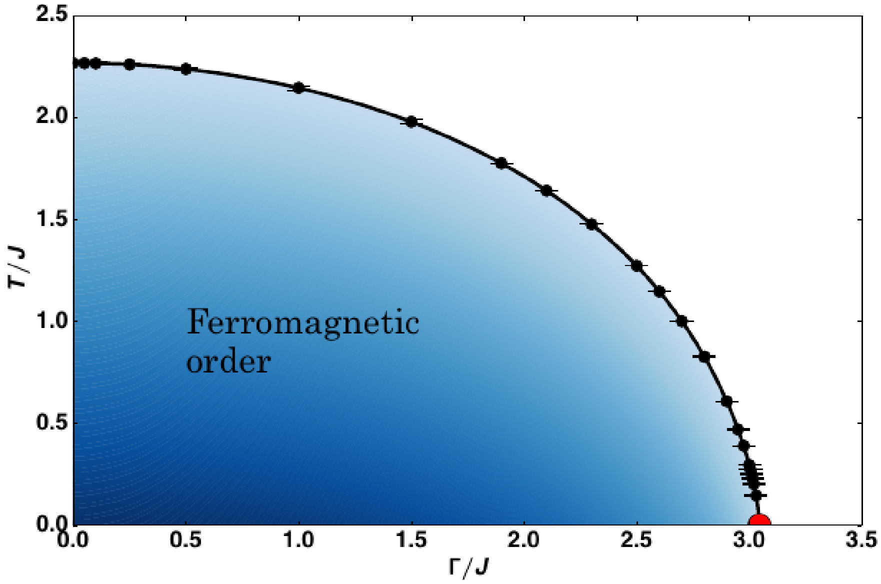


Figure 3.2: Phase diagram for 2D transversal Ising model. Figure taken from [18]

3.3.4 Quantum to classical mapping

In a certain sense, a quantum model in d dimensions can be mapped to a classical model in d+1 dimension.

Due to the Lie product formula

$$e^{A+B} = \lim_{M\to\infty} (e^{A/M}e^{B/M})^M \tag{3.10}$$

we can rewrite

$$\begin{aligned} Z &= \sum_n e^{-\beta E_n} \\ &= \sum_n \left\langle n \middle| e^{-\beta \hat{H}} \middle| n \right\rangle \\ &= \sum_{n_1 \cdots n_M} \left\langle n_1 \middle| e^{-\beta \hat{H}/M} \middle| n_2 \right\rangle \left\langle n_2 \middle| e^{-\beta \hat{H}/M} \middle| n_3 \right\rangle \cdots \left\langle n_M \middle| e^{-\beta \hat{H}/M} \middle| n_1 \right\rangle \\ &= \sum_{n_1 \cdots n_M} \left\langle n_1 \middle| e^{-\beta \hat{H}/M} \middle| n_2 \right\rangle \left\langle n_2 \middle| e^{-\beta \hat{H}/M} \middle| n_3 \right\rangle \cdots \left\langle n_M \middle| e^{-\beta \hat{H}/M} \middle| n_1 \right\rangle \end{aligned} \tag{3.11}$$

Where completeness relations $I = \sum_{n_i} |n_i\rangle \langle n_i|$ inserted M times in line 3. This is similar to a path inegral.

The quantum imaginary time has become a spatial dimension. The quantum model of d dimensions now has the structure of a classical model in d+1 dimensions.

In this way, the 2D transversal field ising model can be mapped to the 3D classical Ising model. [19]

3.4 Operator exponentials

While it is often possible to find exact MPO representation to represent a wide class of hamiltonians (see **??**), it is much harder to do the same for exponentiated operators. These operators play an important role: they act as time evolution operators for quantum systems $|\Psi(t)\rangle = \exp\left(-i\hat{H}t\right)|\Psi(0)\rangle$. A very similar operator governs the partition function in statistical mechanics: the probability of finding a system at inverse temperature $\beta = \frac{1}{T}$ in a microstate i is given by $p_i = \exp\left(-\beta\hat{H}_i\right)$. This is often called "imaginary" time, due to the substitution $\beta = it$. The ability to calculate these operators is essential for understanding the dynamics of a given quantum model, and making contact with real world obsevations of these systems at finite temperature.

3.4.1 Statistical mechanics

The physics of a system in thermodynamical equilibrium can be derived from it's partition function Z. The classical formula generalises to a density matrix ρ as follows:

$$\begin{aligned} Z &= \sum e^{-\beta E_n} \\ &= \sum_n \left\langle n \middle| e^{-\beta\hat{H}} \middle| n \right\rangle \\ &= \mathrm{Tr}\left(e^{-\beta\hat{H}}\right) \end{aligned} \tag{3.12}$$

The first line is the partition function for clasical discrete systems. The index n runs of all possible microstates. It is known that the propability to find the system in a given microstates is given by:

$$p_i = \frac{\sum e^{-\beta E_i}}{Z} \tag{3.13}$$

An useful quantity is the density matrix ρ.

$$\begin{aligned} \rho &= \sum_j p_i \left|\Psi_j\right\rangle \left\langle\Psi_j\right| \\ &= \sum_j \frac{e^{-\beta\hat{H}}}{Z} \left|\Psi_j\right\rangle \left\langle\Psi_j\right| \end{aligned} \tag{3.14}$$

Whith this notation, the partition function Z and ensemble average of an operator $\hat{X}$ are given by:

$$\begin{aligned} Z &= \mathrm{Tr}(\rho) \\ \langle X \rangle &= \mathrm{Tr}\left(\rho \hat{X}\right) \end{aligned} \tag{3.15}$$

3.4.2 Applications

3.4.2.1 Temporal correlation functions

The dynamical behaviour of a system can be captured by its dynamic correlation function:

$$\begin{aligned} C(r,t) &= \left\langle \hat{X}(0,0)\hat{X}(r,t) \right\rangle \\ &= \left\langle \hat{X}(0)e^{i\hat{H}t}\hat{X}(r)e^{-i\hat{H}t} \right\rangle \end{aligned} \tag{3.16}$$

This requires time evolution operators $e^{-i\hat{H}t}$.

3.4.2.2 subsubsection

ground state One practical way of finding the ground state $|E_0\rangle$ is to cool down a given random state $|\Psi(0)\rangle$ to very small T (large β) [2]:

$$|E_0\rangle = \lim_{\beta \leftarrow \infty} \frac{e^{-\beta\hat{H}} |\Psi(0)\rangle}{\langle \Psi(\beta) \mid \Psi(\beta) \rangle} \qquad |\Psi(\beta)\rangle = e^{-\beta\hat{H}} |\Psi(0)\rangle \tag{3.17}$$

3.4.3 Tensor network methods

In the following section I will give a very short review of the current tensor network methods to simulate real or imaginary time evolution. This overview is mainly based on the review paper [20].

3.4.3.1 Approximations to $\hat{U}(\delta)$

The goal is to approximately make a MPO for a small timestep δ which gives a new MPS at time $t + \delta$.

3.4.3.1.1 TEBD Time-evolving block decimation (TEBD) uses the Trotter-Suzuki decomposition. Suppose the chain is split in even and odd sites.

$$\hat{H} = \hat{H}_{\text{even}} + \hat{H}_{\text{even}} \tag{3.18}$$

$$\begin{aligned} \hat{U} &= e^{-i\delta\hat{H}_{\text{even}}} e^{-i\delta\hat{H}_{\text{even}}} e^{-i\delta\left[\hat{H}_{\text{even}}, \hat{H}_{\text{odd}}\right]} \\ &\approx e^{-i\delta\hat{H}_{\text{even}}} e^{-i\delta\hat{H}_{\text{even}}} \end{aligned} \tag{3.19}$$

This is now easy to solve: first apply $e^{-i\delta \hat{H}_{\text{even}}}$ for every even bond and aftwards $e^{-i\delta \hat{H}_{\text{odd}}}$. The error is $O(\delta^2)$ and the number of steps to reach temperature β is β/δ. The error can be made small. This can be generalized to higher order schemes.

3.4.3.1.2 MPO $W^{I,II}$ These method directly use the MPO representation of a certain hamiltonian This is a more recent method (2015) to construct an MPO first detailed in [21]. The idea is to generalise

MPO w1 and 2

$$1 + \delta \sum_x H_x \rightarrow \prod_x (1 + \delta H_x) \tag{3.20}$$

The error is formally still $O(\delta^2)$, but includes many more terms. The advantages lays in the fact that the form above has an efficient representation as an MPO. MPO W^I and $W^I I$ are capeble of dealing with long-ranged interaction terms which makes it suitable to simulate 2D systems **??**

3.4.3.2 global Krylov method

Krylov methods are widely used in linear algebra to calculate eigenvectors. An example is the Lanczos algorithm. These methods are applied to MPS's, but do not fully make use of its structure. For this method, only a MPO representation is needed.

expand on this

3.4.3.3 MPS-local methods

3.4.3.3.1 local Krylov The krylov methods from previous pragraph can be adapted to work on a reduced basis.

3.4.3.3.2 TDVP Time-dependent variational principle (TDVP) can be seen as a further developement of the local krylov method.

Its also been formulated in as a tangent space algorithm, similar to the VUMPS derivation (section 2.5.4.2).

$$i \frac{\partial \left|\Psi(A)\right\rangle}{\partial t} = \mathcal{P}_{\mathcal{A}(\sqcup)} H \left|\Psi(A)\right\rangle \tag{3.21}$$

Where the richt hand side is projected on the tanget space, because the left hand side is a also a tangent vector. (See [8]).

3.5 Conclusion

An introduction to phases of matter, and their surprising structure was discussed. 2 models, namely Heisenberg model and the Ising for different dimensions are discussed. The phases for different dimensions are listed. The last section explained the importance of operator exponentials to understand the physics of these models. Competing tensor network methods to approximate them numerically were very briefly listed.

Chapter 4

Construction Cluster expansion

4.1 Introduction

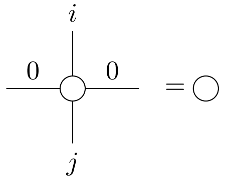
This is the key chapter of the whole thesis. Here, the novel method to construct the operator $e^{-\beta \hat{H}}$, is explained in detail. The basis of the method was first introduced in [22]. There are many variations on the same idea. Some of the most notable examples in 1D will be discussed. At this point, no simulation results will be given. These can be found in chapter 6. The section mentions some objective info about the construction, such as the bond dimension.

The 2D construction will generalise the best result from 1D. First, an anologous construction as in 1D will be presented. As can be expected, also some new ideas are needed to capture the rich phsycis of the models in 2D

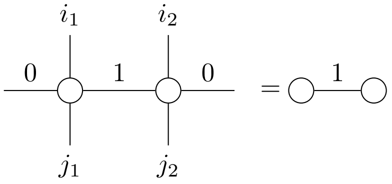
The question of how to construct these cluster expansions and other implementation details are reported in chapter 5.

4.1.1 Notation

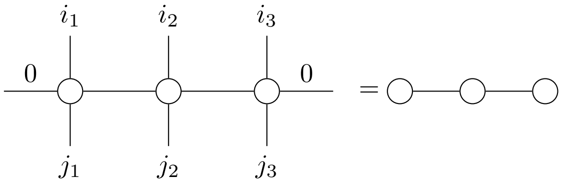
First, some extra clarification on the notation is needed in order to avoid confusion. In the following, the external legs and virtual level 0 will be omitted. Also, all the physical indices will not be shown. This should not be confused with the diagram earlier.



(4.1)



(4.2)



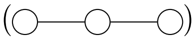
(4.3)

The goals is capture the exponential of the hamiltonian operator $\hat{H}$

$$\hat{H} = \left(\sum_{<ij>} H_2^i H_2^j + \sum_i H_1^i \right) \tag{4.4}$$

This hamiltonian consists of 1 and 2 site operators. Of course more general hamiltonians can also be captured.

The notation for the contraction of the tensor network will also be used to denote the hamiltonian evaluated on the given geometry:

H ()

$$\begin{aligned} =&H_1 \otimes 1 \quad \otimes 1 \\ +&1 \quad \otimes H_1 \otimes 1 \\ +&1 \quad \otimes 1 \quad \otimes H_1 \\ +&H_2 \otimes H_2 \otimes 1 \\ +&1 \quad \otimes H_2 \otimes H_2 \end{aligned} \tag{4.5}$$

This also works in 2D.

4.1.2 Idea

This chapter shows the main construction of dissertation. A cluster expansion is used to approximate $e^{\hat{H}}$ for every possible geometry. The goal is to make a MPO/PEPO which captures the tensor exponential in the thermodynamic limit.

symmetry, speed

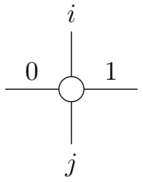
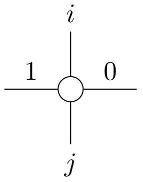
The main idea is to make an extensive expansion by adding blocks which solve the model exactly on a local patch. Crucially, the expansion is not in the inverse temperature β but in the size of the patches. The local patches are separated by a virtual level 0 bond.

To make this somewhat more precise, the first steps of the expansion are shown here. The smallest patch, i.e. 1 site, encodes the exponential of that hamiltonian.

$$\bigcirc = \exp\left(-\beta H(\bigcirc)\right) \tag{4.6}$$

If there were no 2 site interaction, this already captures the full diagonilsation. Of course, such a model wouldn't be useful. The next step is to introduce 2 site interactions, where the one site interactions previously introduced interaction are subtracted from the diagonalised hamiltonian.

$$\begin{aligned} \bigcirc\overset{1}{\text{---}}\bigcirc = \exp -\beta H(&\bigcirc\text{---}\bigcirc) \\ -&\bigcirc\overset{0}{\text{---}}\bigcirc \end{aligned} \tag{4.7}$$

At this stage, all seperated networks with maximally 2 connected sites in a row are diagonalised exactly. Notice that here, 2 new blocks are introduced:  and of course also  . As can be seen, the dimension of sublevel 1 needs to be d^2, with d the dimension of physical level. Although different possible constructions already differ in the next step, one more step is added to make te construction and notation clear.

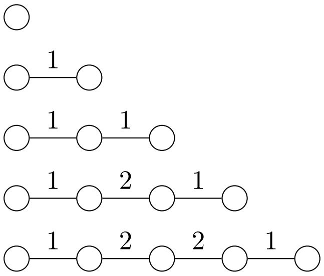
$$\begin{aligned} \bigcirc\overset{1}{\text{---}}\bigcirc\overset{1}{\text{---}}\bigcirc &= \exp -\beta H(\bigcirc\text{---}\bigcirc\text{---}\bigcirc) \\ &\quad -\bigcirc\overset{0}{\text{---}}\bigcirc\overset{0}{\text{---}}\bigcirc \\ &\quad -\bigcirc\overset{1}{\text{---}}\bigcirc\overset{0}{\text{---}}\bigcirc \\ &\quad -\bigcirc\overset{0}{\text{---}}\bigcirc\overset{1}{\text{---}}\bigcirc \\ &= \exp -\beta H(\bigcirc\text{---}\bigcirc\text{---}\bigcirc) \\ &\quad -\bigcirc\text{---}\bigcirc\text{---}\bigcirc \end{aligned} \tag{4.8}$$

It is clear that the right-hand side of eq. (4.8) can be ommited, as it is just evaluating the exponentiated hamiltonian on the same geometry as the left hand side and substructing all possible contractions of the blocks which were added previously. This very compact notation will be able to capture the essence of the different constructions.

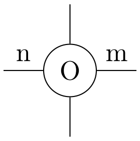
4.2 Construction MPO

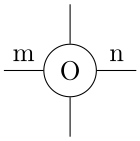
4.2.1 Type A

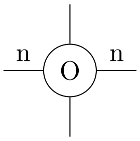
This type was originally proposed in [22]. The first few blocks in the expansion are the following:



(4.9)

The following types of blocks appear in the cluster expansion 

, 

and 

with $n \in \mathbb{N}_0$ and $m = n - 1$.

The O^{nn} block is in defined for a chain with an odd number of sites. The contraction of O^{nm} and O^{mn} is defined by for a chain with even order. The decomposition is defined up to a gauge transformation.

4.2.1.1 Dimension

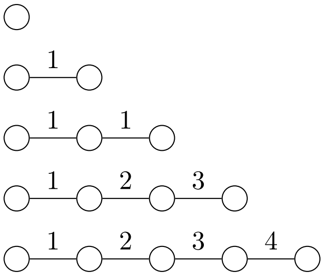
In this scheme, virtual level n has dimension d^n. Of course, this dimension can be lowered if some error is allowed for the longest chain.

4.2.1.2 discussion

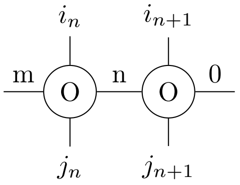
Type A can form long chains, which where not explicitly optimised for. The question arise whether this will results in accurate results for cyclic systems or not.

4.2.2 Type B

Type B only contains blocks of the following form; O^{mn} and O^{n0}.The first few blocks are:



(4.10)

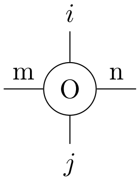


$$= U^n \Sigma V^\dagger \tag{4.11}$$

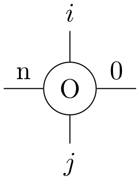
The following split is made: $O^{mn} \cong U^n$ and $O^{n0} \cong \Sigma V^\dagger$. In this way the left inverse exists and doesn't need any calculation: $O^{mn} = U^\dagger$.

4.2.2.1 dimension

From the construction the bond dimension grows from the left to the right. For the last step, there are only d^2 non zero singular values. Each steps adds d^2 to the dimension. For the last step, only d^2 non zero singular values need to be keeped. With the following natation:



$$= A^m_{(\alpha ij)\beta}$$



$$= B^n_{(\alpha ij)\beta} \tag{4.12}$$

The bond dimension of lower virtual levels can be reduced if we can solve the following equations simultaneously:

Then the MPO doesn’t change if there are matrices A'^n, A'^{n+1} and B'^n such that

$$\begin{aligned} S &= A^m A^n = A'^m A'^n \\ T &= A^m B^n = A'^m B'^n \end{aligned} \tag{4.13}$$

Such matrices with optimal bond dimension can be found with generalised SVD. Generalised SVD decomposes 2 matrices as follows:

$$\begin{aligned} S^\dagger &= (U\Sigma_1)Q^\dagger \\ T^\dagger &= (V\Sigma_2)Q^\dagger \end{aligned} \tag{4.14}$$

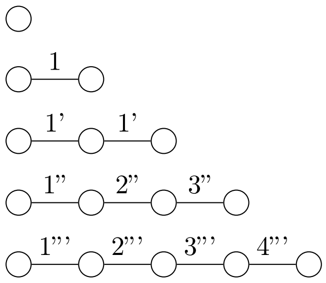
The new bond dimension is the $\dim n' = d^2 \cdot \min(\dim n - 1, \dim(n+1))$. This is higher than the dimension of type A.

4.2.2.2 Discussion

The bond dimension is larger than type A, but the long chains from A are absent. The left inverse is always well defined and doesn’t need any computation, because hermitian matrix U can be inverted easily. One major drawback is that for long chains, the virtula bonds are very large before they can be shrunk with the gsvd procedure.

4.2.3 Type C

This type implements the same strict type as Type B, but in a different way. No calculation is involved, except the calculation of the the exponentiated hamiltonian to certain order. The following kind of MPO strings are allowed:



(4.15)

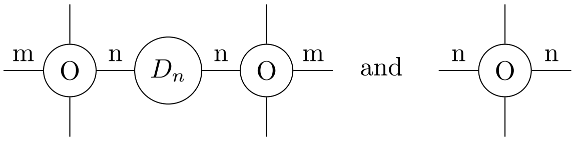
and so forth. All but one MPO elements are chosen to be the identity matrix. The middle one is the exponentiated hamiltonian with reshaped legs.

4.2.3.1 discussion

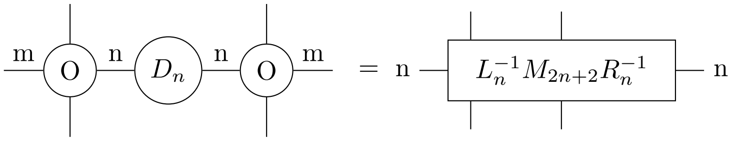
As can be expected from the construction, the bond dimension grows very fast. This type is just as precise as Type B.

4.2.4 Type D

This type uses a differemt setup which tries to capture the best of both Type A and B. Type could handle long range correlation better because of the introuction of O^{nn}, but the inverse was not necesarily well defined. Type B had well conditioned inverses, but performed in most of the cases worse. The block appearing in type D are as follows:



Similar to type A,



(4.16)

$$= U\Sigma V^\dagger$$

Matrix D_n is the singular value diagonal matrix devided by a normalisation factor ϕ. Both U and V are multiplied by $\sqrt{\phi}$.

4.2.4.1 discussion

It's not completely clear what the values of ϕ should be. If ϕ is to large, large chains are not surpressed. If phi is to small, the O^{nn} blocks will become large and hence the chain will diverge again. A reasonable value is the sum of the singular values. Other combination could be tried.

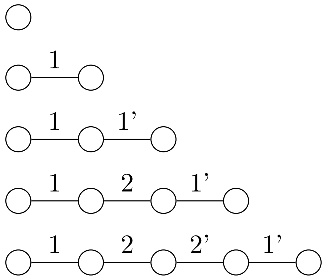
4.2.4.2 matrisation

The cost of this type lies in the fact that it has no compact way of casting it to a matrix. The following works, but has quite a large dimension:

$$\begin{array}{ccc|cc|cc|cc}
O_{00} & O_{01} & & -2O_{01} & & O_{01} & & O_{01}D_1^{1/2} & \\
O_{10} & & O_{12} & & -2O_{12} & & O_{12} & & \\
 & O_{21} & & & & & & & \\
\hline
O_{10} & & & & & & & & \\
 & O_{21} & & & & & & & \\
\hline
O_{10} & & & & & O_{11} & & & \\
 & O_{21} & & & & & O_{22} & & \\
\hline
D_1^{1/2}O_{10} & & & & & & & & D_1^{-1/2}O_{12}D_2^{1/2} \\
 & & & & & & & D_2^{1/2}O_{21}D_1^{-1/2} &
\end{array}$$

4.2.5 Type E

Again, this is a strict variant which needs exactly twice the bond dimension of type A. The idea is to split every chain in a left and a right part. For the left part, the numbers increase while right part they decrease. This construction carries over well to higher dimensions. The first few blocks are:

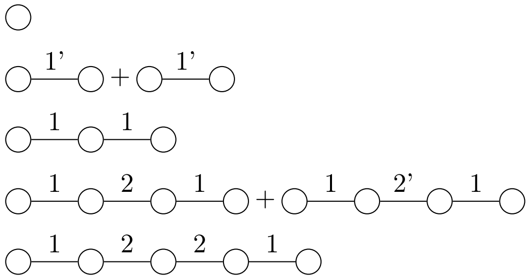


(4.17)

The construction is very similar to type A

4.2.6 Type F

The idea behind this type is very similar to type D. The blocks look as follows:



(4.18)

The blocks $O^{n-1,n}$ and $O^{n,n-1}$ unitary matrices from the svd decomposition, scaled by the largest singular value. The blocks $O^{n-1,n'}$ and $O^{n',n-1}$ are then used to actually solve the problem for the given chain. In the next step, $O^{n,n}$ is added as usual. The idea here is once again to keep this block small, in order to not cuase any divergences.

4.2.7 Conclusion

Many 1D constructions are discussed here. They can be roughly divided in 2 groups. B, C and E are strict variants, meaning only the explicitly constructed blocks will appear in the final expansion. As a consequence, they have exactly the same predictive power. While B has some advantageous properties such as its final bond dimension and well-defined inverses, type E will be used in the results chapter due to its simplicity and scalability. It also generalises well to 2D, in contrast to type B.

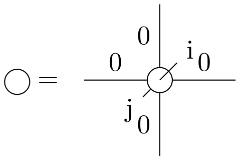
The second category are the unstrict types. Type A has the lowest possible bond dimension to exactly represent a chain of a given length. One hurdle to be overcome are the badly conditioned inverses, when implemented naively. Types D and F try to remedy this. D scales very badly with the maximum number of sites, and has a construction which doesn't fit in with the simple diagrams. The construction was only implemented in 1D code due to this. This type won't be reported, and has a similar performance to type F.

In short, type A, E and F will be reported in section 6.2.

4.3 Construction PEPO

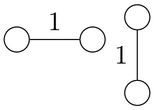
While there where some initeresting choices in the 1D construction, the number of possibilities in 2D is virtually limitless. The focus will mainly be to generalise type A to 2D. As can be expected, the construction starts of quite similar. The following blocks are called 'Linear', due to the way they will be solved.

4.3.1 Linear blocks



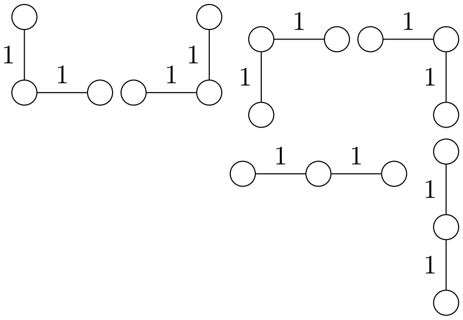
(4.19)

This block now contains 2 physical indices (which will again be hided later on) and 4 virtual legs. Similar to 1D construction, the 0 legs and physical indices are hiden. The next block are:



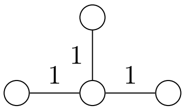
(4.20)

From 3 blocks onwards, the number of extra blocks start to increase:

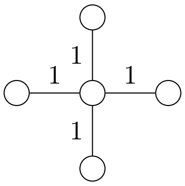


(4.21)

Of course, besides linear blocks also T and + shaped need to be constucted to fully capture the model.

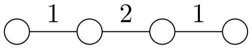


(4.22)



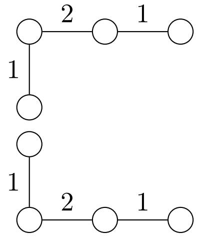
(4.23)

Here, and in the following, care has to be taken in which order the blocks are made. In general, every block which fits in the given block needs to be constructed earlier. The construction continues by introducing a 2 block as before:



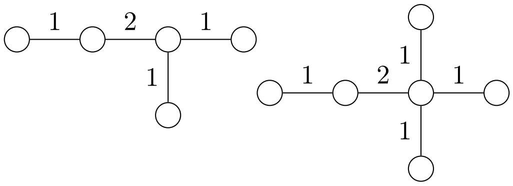
(4.24)

Again, 2 blocks are introduced. For other variations of the linear chain, only one block needs to be solved

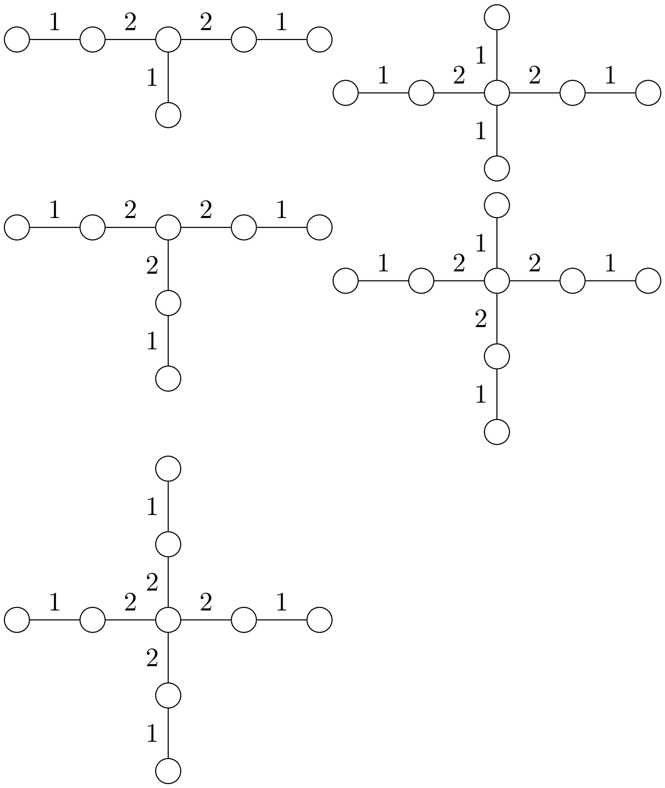


(4.25)

Due to the way they are constructed, the error for every linear chain of a given lenght will be de same as in the 1D case. Once again, all possible T and + blocks are created. Some of these blocks are shown here:



(4.26)



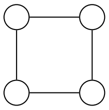
(4.27)

For each block show, there are still multiple permutations of the legs possible. It is clear that a completely automated solver is needed to construct all these different blocks. From here on the construction generalises easily to higher block numbers, and to higher dimensions.

The difference between eq. (4.26) and the blocks in eq. (4.26) is in the larges chain. For eq. (4.26), only chains of order 4 are present, while eq. (4.27) has chains of length 5. It seems that when a virtual level is present, it is most advantageous to create both chains and all blocks, but this will not be the case when a virtual level is truncated.

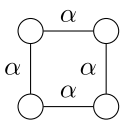
4.3.2 Loops

While the blocks above certainly encode a large number of finite size patches, there are still quite some patches need to be encoded. The simplest case is a 1 square loop.



(4.28)

It is clear that this problem cannot be solved with the techniques from the previous section. The square loop needs a new virtual level α. In general, all the loop levels will be named with greek letters for convenience. The simplest choice for the loop is:

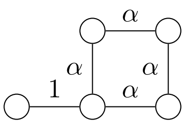


(4.29)

At this points all blocks of order 4 are solved.

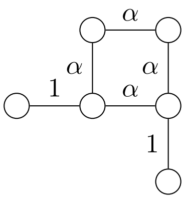
4.3.2.1 Single extensions

The loops need to be connected to the linear blocks. One way to do this is as follows:



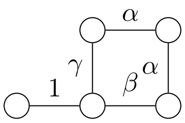
(4.30)

With the given blocks, the following combination is also possible



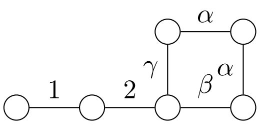
(4.31)

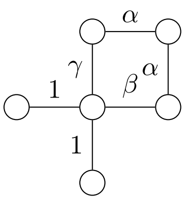
This results in very large errors, and it would require many more blocks to be added in order to counteract this. Luckily, there are many options in 2D. One example is:



(4.32)

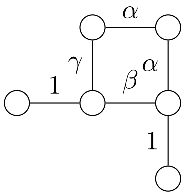
No other combinations are possible, except a loop with on one corner an extension. Of course, there is no need to stop here, the following blocks can now be constructed easily.





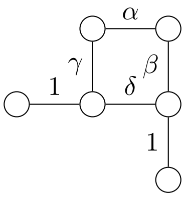
4.3.2.2 Double extensions

It seems as if one of the corner pieces can be used as follows:



(4.33)

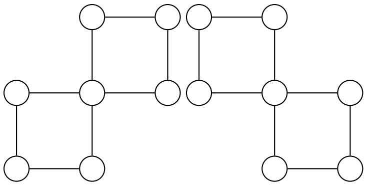
But in other to make a meaningful change in the residual error, the bond dimension of both α and β need to be enlarged significantly. It is more advantageous to introduce yet another level δ, which forms the link between the 2 parts. As both corner tensors can be optimised at once, the total bond dimension is lower than for the previous suggestion, but still larger than the dimensions of the other loop levels.



(4.34)

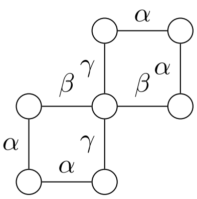
4.3.2.3 Larger loops

One question which comes to mind is where the focus should be for the construction. Making blocks for all possible single loop extensions comes at a increasing cost in total bond dimension. From a physics point of view, the model will be better approximated when smallest non solved patch is solved by introducing new blocks. On the other hand, the already included blocks may cause an error which was not present before the blocks were introduced. One example is a linear chain which closes upon itself, in for instance a 2x3 rectangle. Another example are + blocks which connect upon themselves in the following shape:



(4.35)

One way to solve this, without breaking the rotation pattern of β and γ, is



(4.36)

But this block introduces an infinite tiling, corrupting the expansion.

4.3.2.4 Other ideas

add more

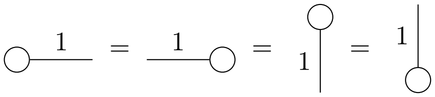
Here it starts to get really tricky.

4.3.2.5 Bond dimension

While for the linear blocks it is clear what the bond dimension should be, this is not the case for the graphs with loops. This problem of the ranks cannot be directly solved for tensor ring decomposition [23], but need to be deduced during the construction. In practice, a bond dimension of 6 is enough to fully solve eq. (4.29) (for physical dimension 2). For eq. (4.32), at least bond dimension 8 is required for β and γ level. This is also sufficient to solve longer extensions and multiple extensions from one corner.

4.4 Symmetry

When the linear blocks are constructed without symmetry considerations, quite some blocks are required. For any one of the 4 legs, the number can range from 1 to M, with M the maximum virtual bond dimension. This results in $(M+1)^4$ blocks. With the solver presented in next chapter, this can be done. Another possibility is to restrict eq. (4.20) further by imposing rotation symmetry of the PEPO legs:



(4.37)

In this way, only the blocks unique up to a permutation of the legs need to be solved.

4.5 Conclusion

The cluster expansions are intrudoced here for both 1D and 2D setting. This can be represented with some very compact notation, as explained in eq. (4.8). The best 1D type, namely A, was used as an inspiration for 2D cluster expansion.

For 2D, mainly the linear blocks, loops and single extension will be important in the results. The reason for this will be discussed in **??**.

Chapter 5

Framework implementation

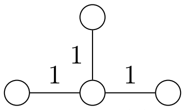
> Experience is the name everyone gives to their mistakes.
>
> Oscar Wilde

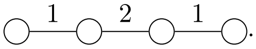
5.1 Introdcution

The construction was explained in the previous chapter, but there is a lot of work that needs to be done to go from an idea to a working implementation in Matlab. The framework to do these calculations was programmed starting from scratch during the course of this thesis. This chapter aims to show some of the work involved. First, the solvers, used to numerically extract the new blocks introduced in for each equation in the previous chapter, are explained. There are 3 different solvers: a linear solver based on matrix inversion, a non-linear solver based on matlab fsolve and a sequential linear solver, which iteratively solves each occurring tensor with the linear solvers takes a step in that direction. The solver need to be as fast as possible, numerically stable and accurate. The optimisation section gives more details about the framework in general. A section is dedicated to the code for generating the phase diagrams reported in the results chapter. The source code is freely available, and section 5.5 shows very briefly how the code is structured. Finally, some of limitations and possibilities relating to the framework are discussed.

5.2 Solvers

The construction can be written down quite compactly as done in the previous section, but actually implementing the code in full generality is somewhat more complex. In practice, the 1D and 2D implementation were performed separately. where a part of the code but mainly the ideas and lessons learned from 1D where taken to the 2D code. Of course, the 1D construction is just a subset of the 2D implementation. In fact, the 2D implementation even outperforms the 1D implementation for reasons with will be explained later. The main focus of this chapter will go to the 2D implementation. At the end some particular optimisations in 1D will be highlighted.

Solving the blocks introduced in the previous sections need 2 different approaches. The graphs of the maps can be split in 2 groups. The first one, considers problems where there are no loops and at most one node with more than 2 legs Examples include: 

and 

. These will be reduced to a standard matrix problem, and solve with matrix (pseudo-) inversion.

The other group, of course, constitutes the nonlinear problems. This includes every problem where a block (or rotated version) occurs more than once, problems which include loops, ...

5.2.1 Linear solver

5.2.1.1 Full inverse

The linear solver is a general purpose block solver which reduces the problem to a set of linear matrix equations. Linear block consist of a tree structure, where the new block is the root of the tree, and all the branches need to be inverted. Let $I^m = (i^1_1 i^1_2 \cdots i^1_{n_1})$ be all the phsycial indices of one leg m and α^m the index between leg m and unknown matrix X. Then the problem can in general, after some tedious tensor leg bookkeeping, be rewritten in the following form:

$$\begin{aligned} &A_{I_1 I_2 \cdots I_n \alpha^1 \alpha^2 \cdots \alpha^m} X_{\alpha^1 \alpha^2 \cdots \alpha^m j} \\ =&B_{I_1 I_2 \cdots I_m j} \end{aligned} \tag{5.1}$$

Here i^M_N has the following meaning: M numbers the differrent legs or branches of the tree, N number of sites of the leg and i numbers the bra and ket states and has dimension d^2. Hence the bond dimension of $I_n = d^{2n_m}$.

The most obvious way to solve this system is by using a linear solver. This results in some numerical problems. This is a result of the potentially ill conditioned inverses inherent to the construction. A pseudo-inverse of the full matrix can be easily obtained and resolves this issue (see for numerical example **??**).

The pseudoinverse A^+ of matrix A is calculated as follows:

$$A = U \Sigma V^\dagger \tag{5.2}$$

$$A^+ = V \Sigma^+ U^\dagger \tag{5.3}$$

Where for Σ^+ all the nonzero diagonal elements are replaced by there inverse. In the numerical pseudoinverse, every singular value below a given threshold e.g. $\sigma_0 = 10^{-12}$ is set to zero.

The problem with this full inverse is that the bond dimension increases very fast: matrix A has dimension $d^{2\sum_m n_m} \times d^{2\sum_m n_m}$. Although using a linear solver instead of full inversion is considerably faster, this method still becomes infeasible for very quickly.

5.2.1.2 Sequential inverse

A second method consist of solving the following sequence of linear problems one leg at a time:

$$\begin{aligned} A^1_{I^1\alpha^1} X_{\alpha^1 I^2 \ldots I^m j} &= B_{I_1 I_2 \cdots I_m j} \\ A^2_{I^2\alpha^2} X_{\alpha^1 \alpha^2 I^3 \ldots I^m j} &= B_{\alpha^1 I_2 \cdots I_m j} \\ &\vdots \\ A^m_{I^m\alpha^m} X_{\alpha^1 \alpha^2 \ldots \alpha^m j} &= B_{\alpha^1 \alpha^2 \ldots \alpha^{m-1} I_m j} \end{aligned} \tag{5.4}$$

While this method is very quick and scales well, in practice it results in unstable result. Solving in a sequential way, the errors of the pseudo-inverses (or worse full inverse) accumulate. If there are 4 legs, the threshold needs to be set at $\sigma_0 = \sqrt[4]{10^{-12}}$. The inverse now becomes a bad approximation of the problem, rendering the results useless.

5.2.1.3 Sparse full inverse

Luckily the problem can be resolved by first performing an SVD decomposition of $A^m_{I^m\alpha^m} = U^m_{I_m\beta^m} S^m_{\beta^m\gamma^m} V^{m\dagger}_{\gamma^m\alpha^m}$ matrices, with S diagonal and U and V unitary. All the U^m matrices can be inverted by applying the hermitian transpose to the leg m of B. The Tensor $S = S^1 \otimes S^2 \cdots \otimes S^m$ is very sparse and can be (pseudo)-inverted at once. For a full rank construction, S is already diagonal. For truncated constructions (or inverses involving loops), this is no longer the case. The last step consist of applying all the matrices V^m to the right hand side.This is shown in this equation:

$$A^1_{I^1\alpha^1} A^2_{I^2\alpha^2} \cdots A^m_{I^m\alpha^m} X_{\alpha^1\alpha^2\ldots\alpha^m j} = B_{I_1 I_2 \cdots I_m j} \tag{5.5}$$

$$\begin{aligned} U^1_{I_1\beta^1} S^1_{\beta^1\gamma^1} V^{1\dagger}_{\gamma^1\alpha^1} & \\ U^2_{I_2\beta^2} S^2_{\beta^2\gamma^2} V^{2\dagger}_{\gamma^2\alpha^2} \cdots & \\ U^m_{I_m\beta^m} S^m_{\beta^m\gamma^m} V^{m\dagger}_{\gamma^m\alpha^m} & \\ X_{\alpha^1\alpha^2\ldots\alpha^m j} &= B_{I_1 I_2 \cdots I_m j} \end{aligned} \tag{5.6}$$

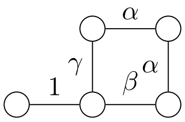
$$\begin{aligned} S_{(\beta^1\beta^2\ldots\beta^m)(\gamma^1\gamma^2\ldots\gamma^m)} & \\ V^{1\dagger}_{\gamma^1\alpha^1} V^{2\dagger}_{\gamma^2\alpha^2} \cdots V^{m\dagger}_{\gamma^m\alpha^m} & \\ X_{\alpha^1\alpha^2\ldots\alpha^m j} &= B'_{\beta_1\beta_2\cdots\beta_m j} \end{aligned} \tag{5.7}$$

The complexity is determined by the SVD decomposition of the individual legs. Due to its sparsity, S does not take much space to construct and is quite fast to pseudo-invert. Doing the pseudo-inverse at once means that it has the same precision as the full pseudo-inverse, as desired.

It is also possible to take a pseudoinverse of a matrix with a QR-decomposition, which is faster [24]. The Q could be inverted directly, and the $R = R^1 \otimes R^2 \cdots \otimes R^m$ matrix is still upper triangular. This method is not used because of the memory requirements to store this matrix is quite large.

5.2.2 Extension

The linear solver is made for problems linear problems. Nevertheless, it can solve every local patch appearing in a map, such as 2 neighbouring sites. These sites are split using an SVD decomposition. Another example is the following corner block, which can perfectly be solved with the linear solver.



(5.8)

The algorithm will treat this as 2 legs.

5.2.3 Nonlinear solver

In some cases, the above solver does not return the best possible solution to a given problem. The reason is that it is not able to incorporate symmetries or solve problems where the new blocks appear more than once. A new solver is needed which does not rely on methods from linear algebra, but on more general non-linear least squares solvers.

In essence, the non-linear least squares solver needs as input a vector with the error values $\vec{f}(\vec{x})$, and if possible also the jocabian, i.e. $J_{I,J} = \frac{\partial f_I}{\partial x_J}$. Tis info is used to chose a direction and a stepsize, minimising the error. An improved point x is chosen by the algoritm, untill some convergence criterium is reached. The implementation uses matlab fsolve routine, which uses Levenberg-Marquardt algoritm under the hood.

5.2.3.1 automatic differentiation

With some care, the jacobian can be calculated for a general tensor network in an automated way. Suppose we want to differentiate the contracted tensor $T^{i_1 \cdots i_n}$ with respect to one of the PEPO blocks $x_n = O^{i_n}_{\alpha\beta\gamma\delta}$. Denote $I = (i_1 \cdots i_n)$ and $J = (i_m \alpha\beta\gamma\delta)$, and this block only occurs once. Then $J_{IJ} = \frac{\partial T^{i_1 \cdots i_n}}{\partial O^{i_m}_{\alpha\beta\gamma\delta}} = T^{i_1 \cdots i_n}_{i_m \alpha\beta\gamma\delta} \delta^{i_n}_{i_m}$ amounts to contracting the network with the tensor x_m removed, and treating the non contracted indices as external ones.

If a tensor appears in multiple places, the sum rule for derivatives has to be used.

5.2.3.2 Symmetry

The non-linear solver can handle rotated and permuted blocks. For instance, a simple loop (square) can be solved by rotating one tensor $T^I_{\alpha\alpha 00}$ 4 times, once for every corner. The Jacobian should be adapted according to the chain rule. Another example is the following decomposition: $X^I_\alpha X^J_\alpha = T^{IJ}$, which is used in eq. (4.37).

5.2.3.3 Combining problems

As only solver, the non-linear solver can solve multiple (non neighbouring) tensors at once, and do this for multiple maps at once.

5.2.4 Sequential linear solver

While from the previous section it seems all non-linear problems need to be solved with the non-linear solver, this is in fact not the case. This solver takes as input multiple new tensors, and solves them one by one. As this is not truly a linear system, the error will not be zero after one pass. But solving the tensors repeatedly lowers the error at each step, giving a iterative procedure. This procedure can be sped up by reusing some parts of the calculations involved in the linear solver. For example, the exponentiated hamiltonian and contraction of all virtual levels that do not involve the optimised blocks only needs to be performed once.

The step is chosen as follows: suppose X is the current tensor and X' the newly computed one. Then the tensor is updated as follows: $X \leftarrow X + \alpha(X' - X)$. If the error has increased, the step is made smaller. The algorithm stops after a number of steps or when a certain threshold is reached.

5.2.4.1 Conclusion

The framework is equipped with 3 different solvers, designed to solve different problems. The code below shows how they are called from within the code:

```
[obj, ln_prefact, err] = solve_lin_and_assign(obj, map, {pattern},
                ln_prefact, struct);
[obj, ln_prefact, err] = solve_sequential_lin_and_assign(obj, map, {pattern},
                ln_prefact, struct, {rot_90});
[obj, ln_prefact, err] = solve_non_lin_and_assign(obj, {map}, {pattern},
                ln_prefact, struct, {rot_90});
```

They only need a map, which is the gemetry of the problem, and a pattern, i.e. the new block to add. `rot_90` is an optional argument, listing all the permutations (such as rotation symmetry over 90 degrees). For completeness, `ln_prefact` is the normalisation factor as discussed will be discussed in the next ??.

Whenever the linear solver can be used, it is the solver of choice. The sparse full inverse procedure is fast and handles the ill-conditioned inverses very well.

The sequential linear solver builds on this solver to handle the introduction of multiple new tensors, possibly related to each other through a permutation.

The non-linear solver is at the moment only the fastes for small highly non-linear problems, such as solving eq. (4.29) in a rotation invariant manner. The non-linear solver can optimise multiple problems at once, and could be extended to fully use internal symmetries of the model .

5.3 Optimasation

5.3.1 Bookkeeping

One important aspect of programming these general solvers is to devise a scheme that keeps track of all the involved tensors and transforms to problem to the form discribed above. In the code, the geometric info is represented by a map. This keeps track of the neighbours for each site, numbers the internal and external legs and a list to perform the contractions.

The framework provides some tools to transform these maps into other maps, for instance by removing 1 site.

5.3.2 Fast contraction

One particular task is to determine all the possible combinations of virtual levels for a given geometry. Simply looping over all possible combinations scales as n^m, with the number of virtual levels and m the number of internal legs. This quickly becomes a bottleneck. This problem can be restated as a PEPS contraction in the following way: for each site make a tensor $T^i_{\alpha\beta\gamma\delta}$ where i encodes all the non-empty combinations of legs $(\alpha\beta\gamma\delta)$. On each site, the right boundary conditions need to be applied to get the right geometry. After setting the boundary conditions, the sparse PEPS network can be contracted and the resulting tensor gives, after decoding, all the possible contractions. Due to its sparsity, this performs quite fast. As an added bonus, removing a tensor from T gives all contractions without this tensor. As both results are sorted list, the subset of contractions containing a given tensor can also be found fast.

5.3.3 Normalisation

??

For many of the end results, the PEPO cells can be divided by a normalisation factor. Normalising the calculations is important, because $\exp\left(\hat{H}\right)$ scales exponentially in the number of sites. Luckily, the exponential can be calculated directly in normalised form. Suppose H is the matrisation of the hamiltonian evaluated for a certain geometry. This is a hermitian matrix and can be diagonalised $H = QDQ^\dagger$ with Q unitary. Then

$$\begin{aligned} exp(H_{d^N} - N\alpha I) &= Qexp(D - N\log(\alpha)I)Q^\dagger && (5.9) \\ &= Q \begin{bmatrix} exp(D_{11} - N\log(\alpha)) & & \\ & \ddots & \\ & & exp(D_{d^N d^N} - N\log(\alpha)) \end{bmatrix} Q^\dagger && (5.10) \\ &= \frac{exp(H_{d^N})}{\alpha^N} && (5.11) \end{aligned}$$

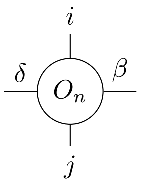
. With I the unit matrix. Next to a global normalisation factor, every block calculation calculates a specific normalisation factor such that the largest eigenvalue of $exp(H)$ is of the order 1.

5.3.4 Internal representation

Two main internal representations are used to construct the given MPO. Either, the MPO is stored as a cell of matrices, or as one big matrix where the blocks are added to during the construction. The output type can be chosen. For some types, sparse matrices are used during the construction. Given that Matlab doesn't support multidimensional matrices by default, this[25] library is used.

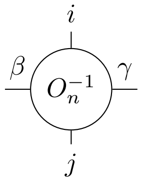
5.3.5 Even faster inverses

While the inversion procedure above states how to make use of pseudoinverses, it was not yet clear in the 1D case it was needed. The 1D implemetation uses a trick to get all the inverses for free from the SVD decomposition. Take the MPO which corresponds to a unitary matrix:



$$\cong U^n_{\alpha(ij\beta)} \qquad (5.12)$$

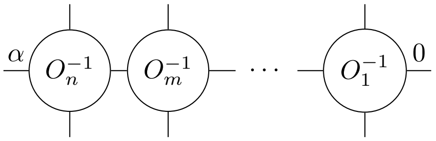
Then the inverse MPO can be calcuated by taking its Hermitian conjugate and reshaping.



$$\cong U^{n\dagger}_{(ij\beta)\gamma} \qquad (5.13)$$

The left inverse looks like

2 =



(5.14)

where the physical indices need to be contracted with the corresponding indices of the right hand side.

5.3.6 Buffering Results

Some calcalations, such as calculating the matrix exponential, take some time. In 1D code, the same calculations were performed over and over again, and hence a buffer mechanism was written to store these results. In the 2D framework, this not necessary as the solvers only calculate the matrix exponential once and return the blocks together with the made error.

5.3.7 Profiling

5.3.8 Calculating the error

profiling

5.4 Calculating phase diagrams

This section details how the phase diagrams are calculated, stored and the critical parameters fitted. The results are discussed in section 6.4

5.4.1 Points sampling

This concens the problem which temperatures to select to calculate the the phase diagram. As the transition between 2 phases is sharp, an uniform sampling in T is not the best option. A very fine grid is needed to capture the transition well, requiring high computation times. The sampling starts by calculating the magnetisation for N uniformly sample points between 2 temperatures. These calculations are performed in parallel on a multicore server. When they are all finished (or have run for a maximum amount of iterations), all the arch lengths are calculated, and repeatedly a new T point is inserted in the largest interval until N new points are selected. The arch length in the m-T plan can be changed to require more points in the m direction than T direction.

5.4.2 Storing the information

Each run has a template with all the common model info. For each point 2 files are stored. One file contains the info and results, such as temperature, magnetisation, correlation length, etc. The other file is much larger and contains the PEPO tensor, the calculated vumps environments,... The files of the first kind are used in other calculations, such as fitting procedure. Reassembling the files into one structure happens in a central function. Another function is able to reprocess already calculated points. The sampling can be continued from where is was last stopped.

5.4.3 Fitting

mps bond scaling, ...

5.5 How to use

All the code neede to generate all the results from this dissertation is available on my github page `https://github.com/DavidDevoogdt/Thesis_Tensor_Networks`. The starting points to explore the code are in the readme file.

5.5.1 Source code structure 1D

The source code for this project can be found on github. The implementation of these types can be found under `src/generateMPO.m`. In this class the different types of MPO can be constructed. It bundles some helper functions such as contracting a chain or cycle of MPO's or construction of an exponentiated hamiltonian for the given input hamiltonian. Other examples are making L_n^{-1} by sequential invers MPO contractions,...

`src/test.m` contains the code to create the plots to compare different types and orders. The other files in the folder are self-explanatory.

5.5.2 Source code structure 2D

source code 2D

5.6 Limitations and outlook

In short, all components are in place in the framework to generate easily and efficiently all the blocks. Since day one, this is a constant work in progress

5.6.1 Size limitation

The main bottleneck is, as expected, calculating the matrix exponential for large systems. The other components are efficient enough to not cause any troubles.

5.6.2 Lattices

The models studied were all on a square lattice. The universal physics does not depend on the details of the model, such as the lattice. It would be beneficial to be able to simulate to these lattices, but here also higher dimensions could be included. In essence all the information of the lattice is contained in the maps generated for each calculation, such as all the connected sites, how to contract them, etcetera. Although undoubtedly many details will need to be changed to use it in practice, the solvers can stay almost the same.

5.6.3 Symmetries

At the moment rotation and permutation symmetries can be included in the construction of the blocks. Internal symmetries are not yet included at the moment. Including them could push the computational boundaries further.

5.6.4 Code quality

The first and most important goal of writing numerical code is of course that it compiles and that the results are as correct. But this is only the first step. The 2D framework neatly orders the different task is functions to avoid as much code duplication as possible. For instance, there are functions generating and manipulating maps, which depends on the geometry of the problem. Other functions only use the bonds and contractions listed in that map object, and hence another lattice can be used without changing the other components too much.

Another example is the solvers, which get the blocks through a common function

finish this

Chapter 6

Results

> With four parameters I can fit an elephant, and with five I can make him wiggle his trunk
>
> John von Neumann

6.1 Introduction

This section explains the acuracy and performance of the given cluster expansions. The first section compares the different constructions in 1D based on the error relative to the exact solution. The best algorithm will form the basis for the 2D results. First, similarly to 1D, the results will be checked based on the error relative to the exact solution. Then, the expansion is used to calculate the phase diagram of the 2D transversal field Ising model.

6.2 Results 1D

In this section the results of the MPO construction detailed in

6.2.1 Exact tensor matrix exponential

The performance of the MPO construction can be compared with the exact diagonalisation of the hamiltonian for a given number of sites. To obtain a faithful results, the number of sites should be as high as possible. In practice, diagonalisation of large matrices becomes slow and memory consuming. The size grows exponentially in the number of sites: $d^n \times d^n$. A double takes 8 bytes of memory.A Rough estimated of the amount of RAM R needed to store this complex array is:

$$R = d^{2n} \times 16 bytes \tag{6.1}$$

Which means a 14 site chain already takes up more than 4 GB of RAM. The complexity to calculate a matrix exponential scales as $O(n^3)$ [26]. In practice this means that, without any tricks, the matrix exponential can be calculated for 12 sites.

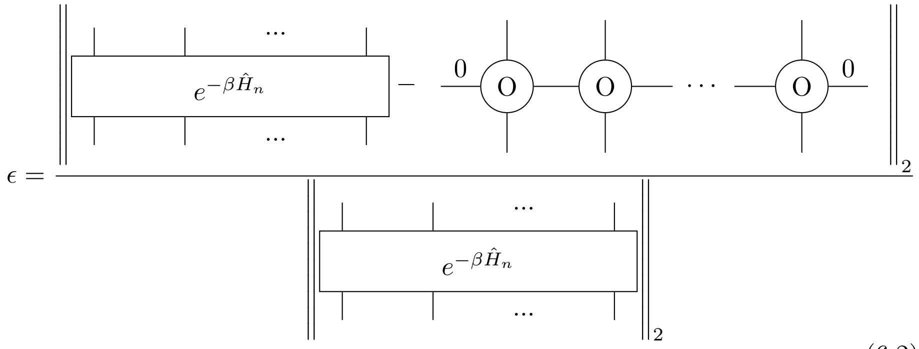
State of the art algoritm for exact diagonalisation, which include all symmetries and are optimased for parallelsation, can calculate up to 50 sites. [27]

make version for cyclic

trace norm, schatten p norm, ...

6.2.1.1 norms

The schatten 2 norm is used in the following analysis, dentoted by $\| \cdot \|_2$. In the figures the relative error ϵ is reported.



(6.2)

This norm can only be calculated for a finite number of sites. The influence of the number of sites for a linear and cyclic fig. 6.1. As expected, the cyclic norm represents large systems better for the same number of sites. The linear norm keeps increasing with every added site.

Calculating the cyclic norm comes at the extra cost of contracting a cyclic tensor network. In this chapter, the cyclic norm will be given for M=8 sites.

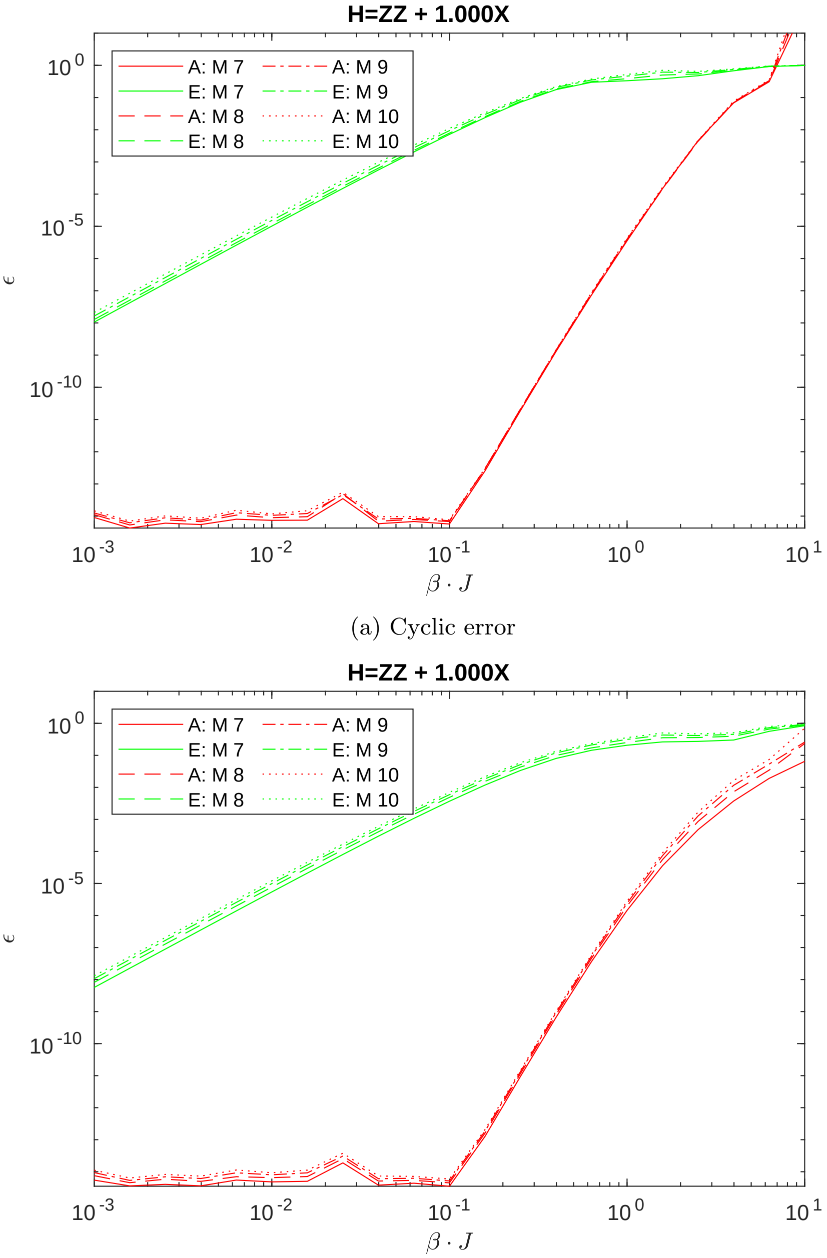
6.2.2 Inversion procedure

??

6.2.2.1 Full pseudo-inversion

The procedure for inversion of the linear blocks was detailed in section 5.2.1. To demonstrate the need for a pseudoinverse, the error for transversal Ising models is shown in fig. 6.2 . The pseudoinversion has 1 parameter σ_0, the cutoff below which the sinular values are set to zero. A value of 10^{-12} seems optimal in the sense that it doesn't introduce large fluctuations, but still is able to produce good inverses.

6.2.2.1.1 Truncation The original 1D code does not yet have this full inversion procedure. Instead an optimality criterium was devised to truncate the



(a) Cyclic error

(b) Linear error

Figure 6.1: Different error measures for 1D transversal Ising model

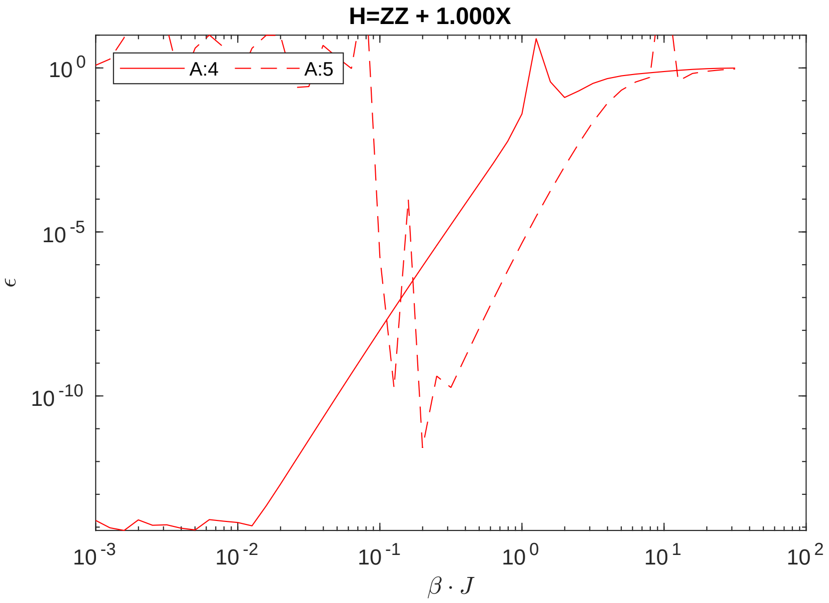


Figure 6.2: Error for transversal Ising, computed with full inversion

series. Needless to say, even with this criterium, the results were a lot worse at low beta

6.2.3 Models

With these basic definitions and findings out of the way, the different serie expansions can be tested against some different physical models.

6.2.3.1 Ising

The results for the different types are all bundled in fig. 6.3. The virtical axis show the logaritm of the relative error ϵ, horizontal axis the normalised inverse $beta = \frac{J}{T}$. The most surprising finding is the fact that the strict type E performs worse than the others. For low β, it is more than 5 orders of magnitude worse, Taken together with its large bond dimension, this expansion and all other strict variants are seen to be of no use.

Now lets focus on the 2 other variants. Type A clearly outperforms type F for all $\beta < 2$ by quite some margin. Only for large β, type F has the upper hand. While not completely clear in the picture, type A has quite a large error for some orders at $\beta \in (2, 10)$, but higher orders seems to solve the problem. The construction of type F requires twice the bond dimension, and hence type A performs clearly better overal

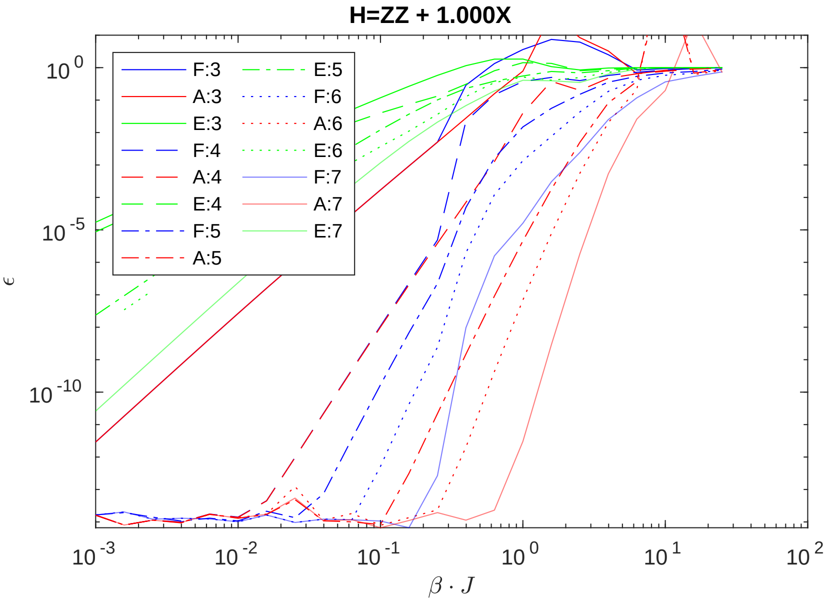


Figure 6.3: Comparison type A, E and F for Transversal Ising.

6.2.3.2 Heisenberg

Now we focus on the spin 1/2 Heisenberg model on a chain. The results are again displayd in fig. 6.4. The exact type seems to perform a lot better here, but still has consistently the largest error of the 3 methods for a given order. For low β, type A again performs better than type F. At $\beta \approx 1$ and larger, F starts to improve upon the result of A, and doesn't have an divergent error.

6.2.3.3 Random

To give a representative overview for random hamiltonians, several simulations were run. The single site and nearest neighbourgh hamiltonians are generated by making hermitian matrices with random real and complex numbers between -1 and 1. In order to compare the different graphs, the engergy scale is set such that the norm of the hamiltonian evaluated on 2 sites is 1.

run random hamiltonian

6.2.4 Conclusion

- strict variant are useless

write this

- full smmetry is not needed
- pseudoinverse neccesary
- Type A is the best

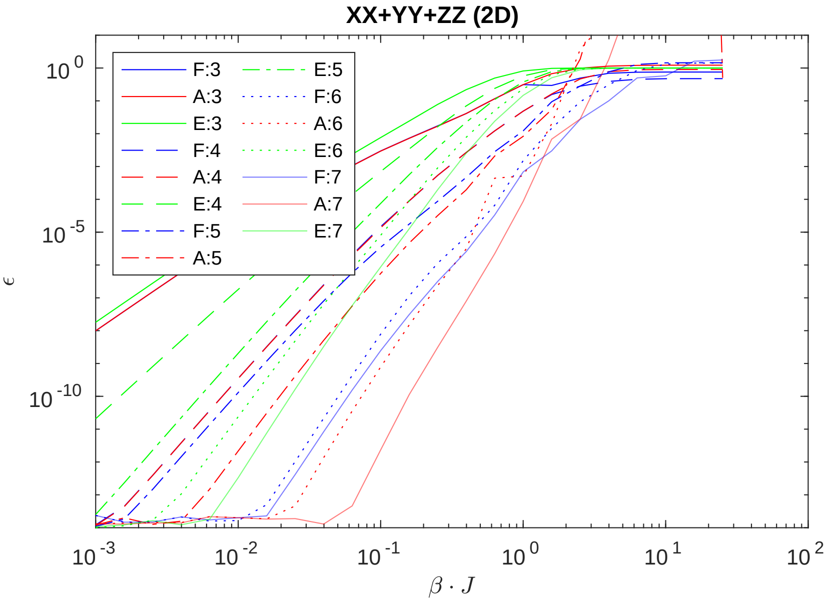


Figure 6.4: Comparison type A, E and F for Heisenberg model.

- never add new block if previous one was not fully solved (truncation)
- truncation adds possibility for extrapolation

6.3 Results 2D

The results in 1D are very promising, but the real open challenges are situated in 2 (or higher) dimensions. Therefore, it is an interesting and relevant question whether the results generalise to 2D. These results are twofold. On the one hand, a similar measure as in 1D is calculated to measure how good a method works on a specific 2D grid. Here, we will need to settle for what can be computed. As a second measure, and from physical point of view the most interesting one, some phase transitions of the 2D transveral Ising model are determined, and compared against values in literature.

6.3.1 Norm

Once again, a suitable norm has to be derived. The situation is more difficult than in 1D, because contracting a PEPO tensor network has a much larger computational complexity than in 1D. Ideally, the norm would be calculated on a nxn cyclical grid (2D grid on a torus). Here n has to be at least 1 larger than the largest explicitly constructed chain.

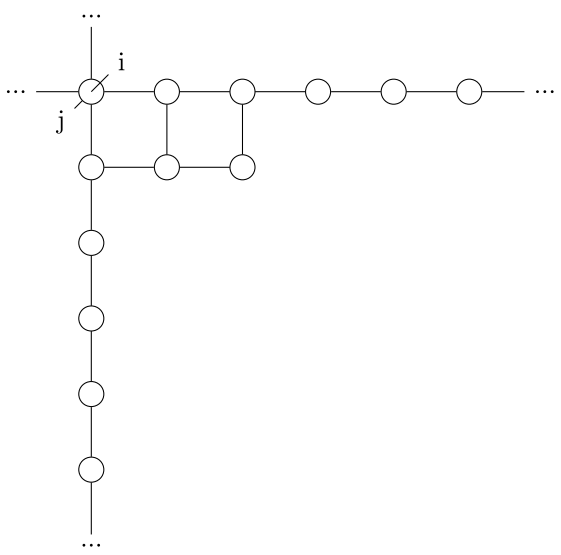
In practice, this is not achievable in reasonable amount of time. The limitations are twofold: the maximum number of sites to calclate the matrix exponential is still around 14, and the contraction of the tensor network is limited to

number

The limitations on the PEPO contraction can be somewhat relaxed by not computing the full network, but computing the reduced density matrix. Here, most of the sites have their physical indices traced out, except for one site.

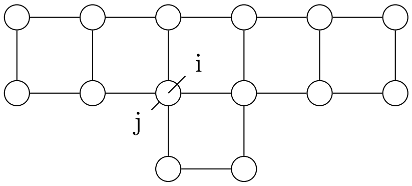
The final error measures are defined by:

$$\rho^1_{i,j} = \qquad (6.3)$$



and

$$\rho^2_{i,j} = \qquad (6.4)$$



For the first one, all the blocks in the expansion are present in, and it is cyclical in both x and y. The second reduced density matrix is not cyclic, but includes more loop type contributions. Both relative errors will be used, defined by :

$$\epsilon^\alpha = \frac{\left\|\rho^\alpha_{exact,i,j} - \rho^\alpha_{i,j}\right\|_2}{\left\|\rho^\alpha_{exact,i,j}\right\|_2} \quad \alpha \in [1, 2] \qquad (6.5)$$

For the given error measures, series expansion up till order 5 can be tested. The second norm takes the most time to compute, due to the larger complexity of contracting the tensor network.

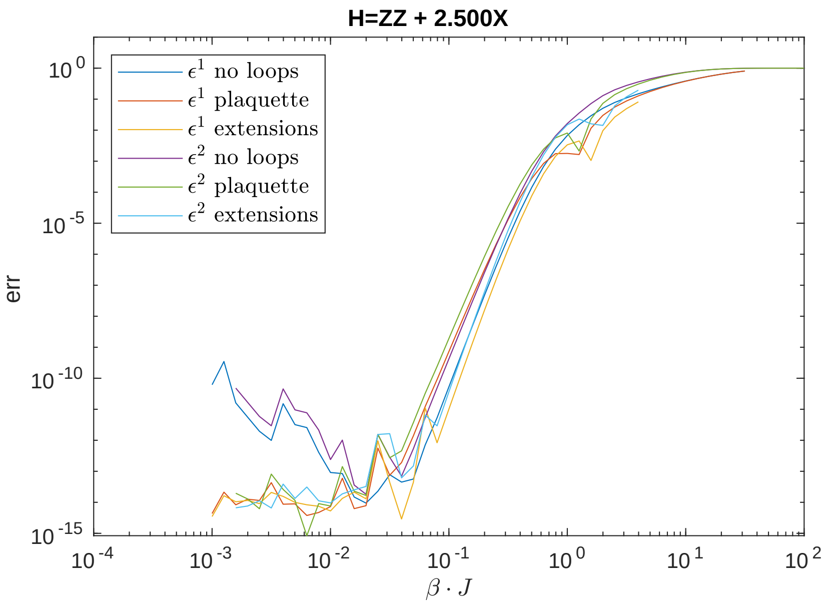


Figure 6.5

6.3.2 Models

6.3.2.1 Ising

We first determine the optimal value σ_0 for pseudoinversion and test the influence of the addition of plaquette term to the expansions. The series is of order 5, and the transversal field is of magnitude 2.5.The results can be seen in **??**.

The truncation parameter σ_0 was optimal at a value of about $\sigma_0 = 10^{-12}$ in 1D. Smaller values resulted in divergent series exapnsions. In the graph we see that even better behaved inverses of $\sigma_0 = 10^{-10}$ improve upon the error made.

The plaquette term (**??**) is mainly important at low β to keep the error low, and also has some considarable influence at higher temperatures.

fix caption

6.3.2.2 Heisenberg

6.3.3 Conclusion

6.4 Phase diagram 2D transveral Ising model

Now we have an idea how accurate the cluster expansions is, we can use it to calculate the thermodynamic properties of the transvere field Ising model. The sampling of points was explained in section 5.4

As a reference, fig. 3.2 is shown once more

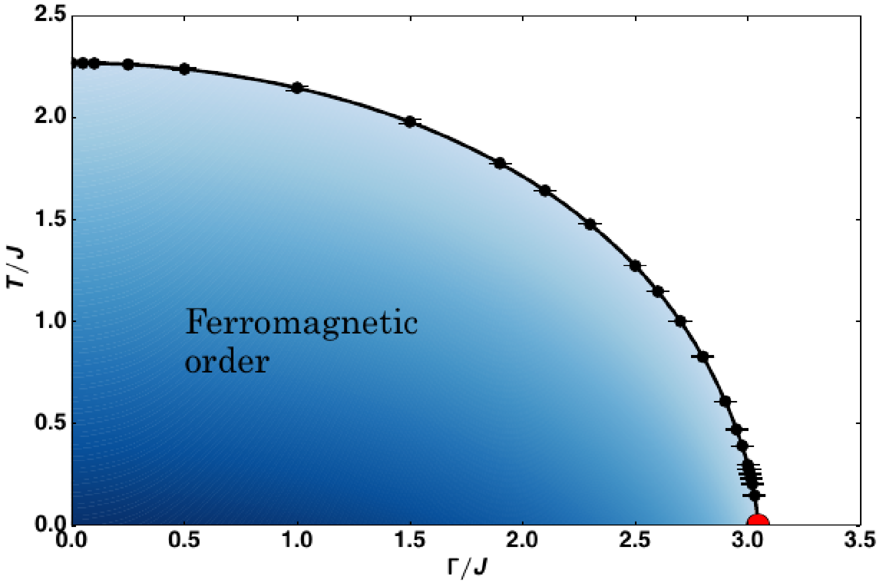


Figure 6.6: Phase diagram for 2D transversal Ising model. Figure taken from [18]

6.4.1 Classical Ising phase transition

The classical ising model on a square lattice has an exact solution. Onsager calculated the critical temperature to be $T_c = \frac{2J}{k \ln\left(1+\sqrt{2}\right)} \approx 2.69185$. A test for the construction is to simulate this and compare the fitted (see section 5.4.3) results.

At the relevant temperature, an order 5 series expansion without loops is sufficient. This keeps the bond dimension small and hence the environment can be computed faster with VUMPS.

The results are shown in fig. 6.7. Each of the figures has 4 different subplots. Left upper corner shows m vs T. For low T, there is clearly a nonzero macroscopic magnetisation, while high T has a zero expectation value as expected. The other three plots show the data collapse of the finite size scaling of the entanglement entropy S, the correlation length ξ and the magnetisation m. δ is marek gap, a measure for the system size as explained in section 3.2.5.

new figure with entropy formula fixed 6S/c instead of cS/6

Near criticality they collapse well, but away from the critical point there is quite some systematic variation. The reason is shown in fig. 3.1. Only in some limited range around the critiacal region, the universal scaling holds.

A more zoomed in version is shown in fig. 6.8. Clearly. the data collapses a lot better as expected. In fact, the data collapses so well we could determine the the critical exponents and the temperature with the given data.

provide numbers for fit

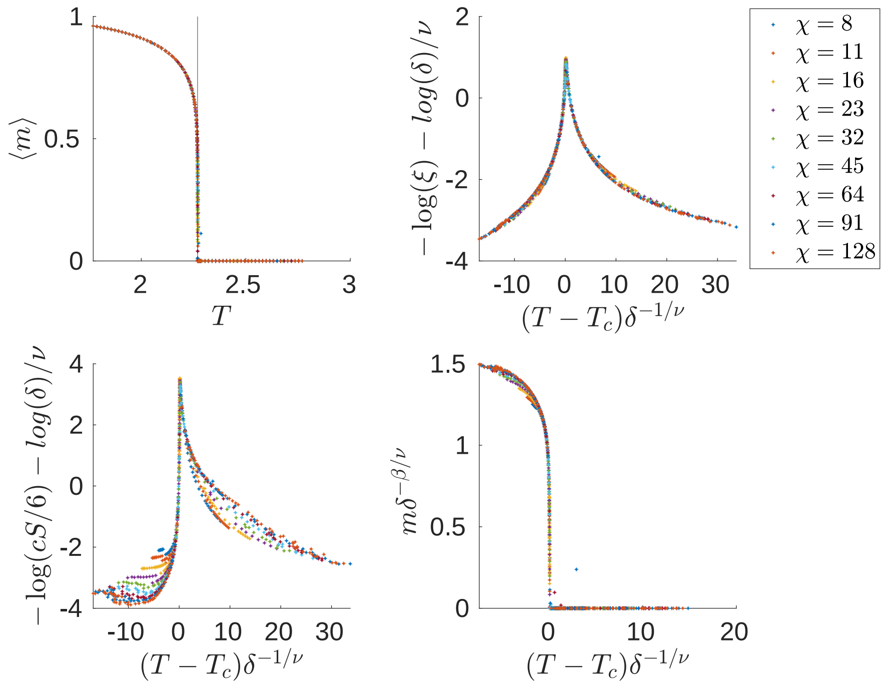

1
0.5
0
$\langle m \rangle$
2
2.5
3
T
2
0
-2
-4
$-\log(\xi) - log(\delta)/\nu$
-10
0
10
20
30
$(T - T_c)\delta^{-1/\nu}$
$\chi = 8$
$\chi = 11$
$\chi = 16$
$\chi = 23$
$\chi = 32$
$\chi = 45$
$\chi = 64$
$\chi = 91$
$\chi = 128$
4
2
0
-2
-4
$-\log(cS/6) - log(\delta)/\nu$
-10
0
10
20
30
$(T - T_c)\delta^{-1/\nu}$
1.5
1
0.5
0
$m\delta^{-\beta/\nu}$
0
10
20
$(T - T_c)\delta^{-1/\nu}$

Figure 6.7

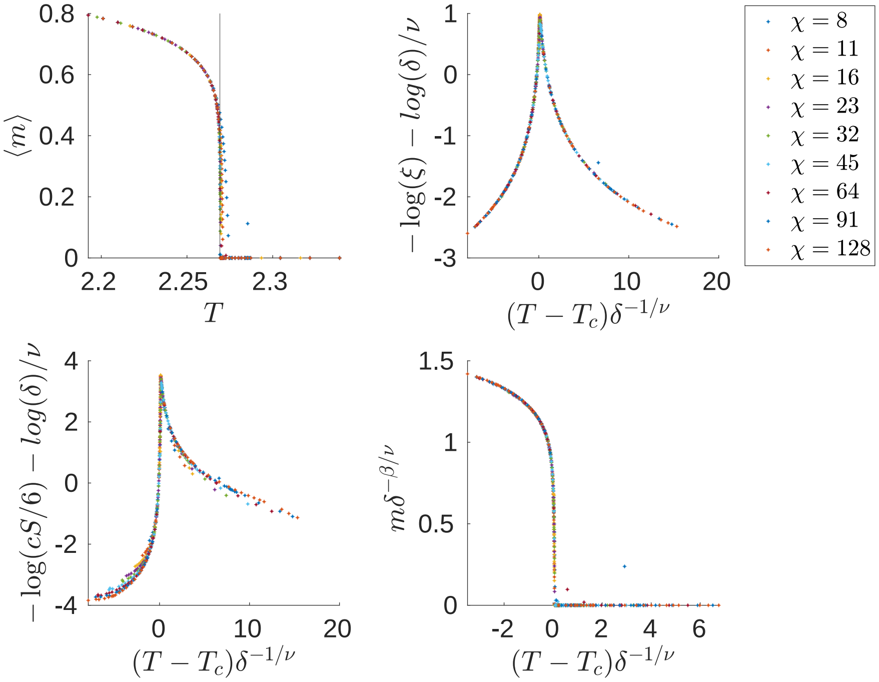

0.8
0.6
0.4
0.2
0
$\langle m \rangle$
2.2
2.25
2.3
T
1
0
-1
-2
-3
$-\log(\xi) - log(\delta)/\nu$
0
10
20
$(T - T_c)\delta^{-1/\nu}$
$\chi = 8$
$\chi = 11$
$\chi = 16$
$\chi = 23$
$\chi = 32$
$\chi = 45$
$\chi = 64$
$\chi = 91$
$\chi = 128$
4
2
0
-2
-4
$-\log(cS/6) - log(\delta)/\nu$
0
10
20
$(T - T_c)\delta^{-1/\nu}$
1.5
1
0.5
0
$m\delta^{-\beta/\nu}$
-2
0
2
4
6
$(T - T_c)\delta^{-1/\nu}$

Figure 6.8

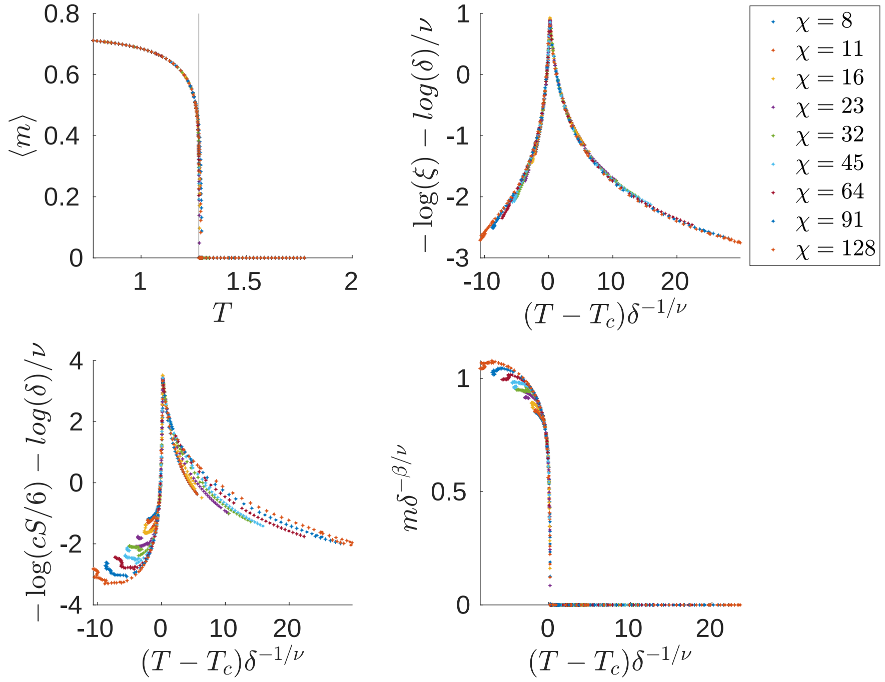


Figure 6.9

6.4.2 g=2.5 phase transition

The previous example simulated the classical 2D ising model. The question is whether this result will carry on into the quantum regime.The results for the full phase diagram fig. 6.9 and or points in direct neighbourhood of the phase transition fig. 6.10 are similar to the 1D case.

Of course, quantitive details are different: the phase transition happens at $T \approx 1.274$ and the maximum magnetisation is lower than 1, in accordance to fig. 3.2. The variation of m around the critical temperature for different bond dimensions is much higher. A much larger χ is needed for the fixed point MPS. This is not necessarily a problem, as this gives better data for the finite size scaling.

There is no analytical expression for critical temperature known. With quantum monte carlo techniques, a value of $T_c = 1.2737(6)$ is obtained, while state of the art tensor network techniques provide a value of $T_c = 1.2737(2)$ [28]

With the series expansion from this paper and Vumps, $T_c = 1.2736(6)$, indicating that this faithfully captures the physics. To put his into context: the directly calculated error ϵ^2 at $T = 1.2$ was around 0.006.

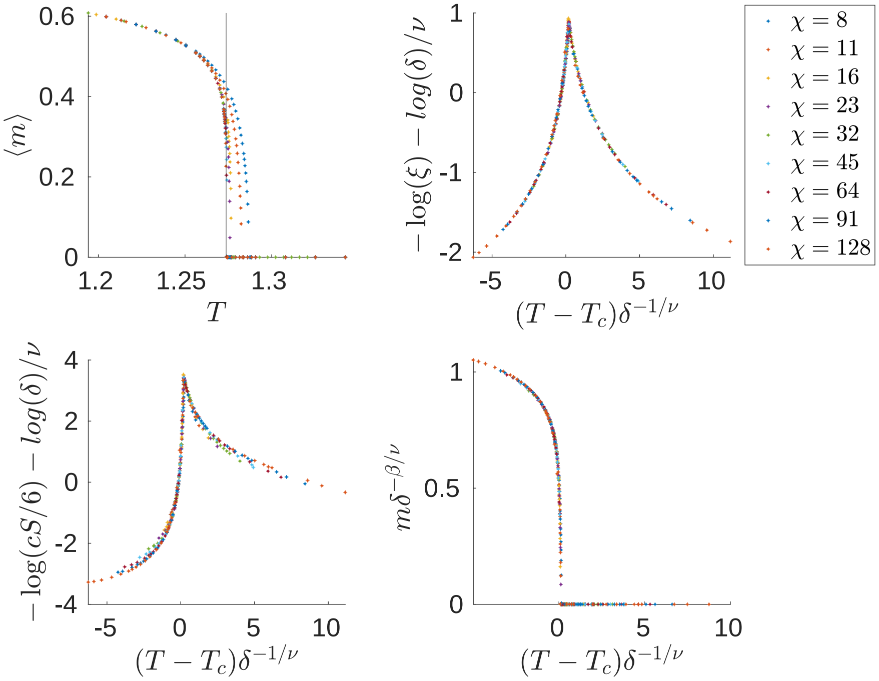

0.6
0.4
0.2
0
$\langle m \rangle$
1.2
1.25
1.3
T
1
0
-1
-2
$-\log(\xi) - log(\delta)/\nu$
-5
0
5
10
$(T - T_c)\delta^{-1/\nu}$
$\chi = 8$
$\chi = 11$
$\chi = 16$
$\chi = 23$
$\chi = 32$
$\chi = 45$
$\chi = 64$
$\chi = 91$
$\chi = 128$
4
2
0
-2
-4
$-\log(cS/6) - log(\delta)/\nu$
-5
0
5
10
$(T - T_c)\delta^{-1/\nu}$
1
0.5
0
$m\delta^{-\beta/\nu}$
0
5
10
$(T - T_c)\delta^{-1/\nu}$

Figure 6.10

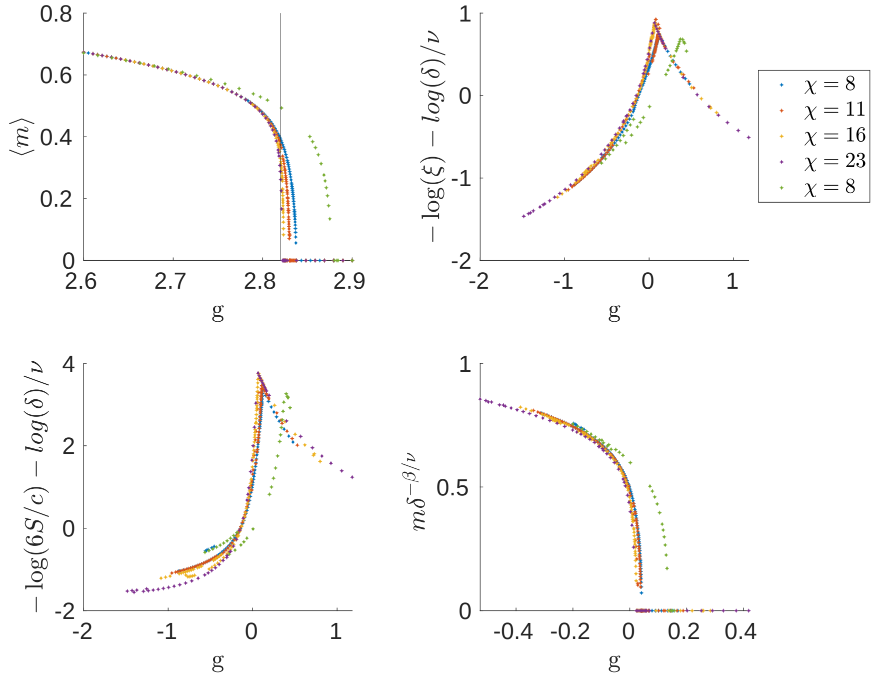


Figure 6.11: Results for $T = 0.7$ phase transition. The green points are a truncated order 6 construction, the others order 5.

6.4.3 T=0.7 quantum phase transition

Up untill now, the phase diagram in fig. 6.6 was explored at constant g in function of T, but of course there is also a transition at constant T and varying g. T = 0.7 is chosen. The results are shown in fig. 6.11.

Clearly, the results do not collapse as well as for the other models. The reason is that the order of the expension is not high enough. The green curve is order 6, where virtual level is truncated to dimension 20. The others are order 5. The green and blue curve both have MPS bond dimension $\chi = 8$, but predict quit different magnetisation fpr large g. This shows that for $g > 0.7$, order 5 is not sufficient anymore.

6.4.4 Tricritical point

Now that we know the critical transveral field can be determined, all the tools are present to extrapolate the Tricritical point, indicated by a red dot in fig. 6.6. To achieve this, the following scaling relation can be used:

$$T_c = |g_c - g_{c,q}|^{z\nu_{3D}} \tag{6.6}$$

Near the critical point, the critical temperature T_c and the critical transversal field g_c are related to each other by the value of the quantum critical point $g_{c,q}$ and critical exponent $z = 1$ and $\nu_{3D} \approx 0.62998$ [18].

This could perfectly be done, given enough computation time to calculate for many temperatures the critical transversal field with a finite size scaling, similar to section 6.4.3. Possibly, also a scaling in truncation dimension for the cluster expansion is needed.

6.4.5 Going beyond

With all the built machinery to construct cluster expansions, is logical to calculate the phase diagram with increased precision.

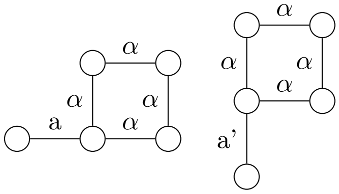
6.4.5.1 Higher order

Going to higher order (i.e. longer linear chains) and adding the single loop contribution works well. The bond dimension of largest virtual level (3 in this case) can be truncated in the construction. The linear and non-linear solver find the least squares solution to the problems.

Care has to be taken in order to not violate one of the conclusions in 1D: never construct a longer chain than the previously fully solved one. For instance if the bond is truncated to 10, the linear chains can be constructed up till order 4. This means eq. (4.26) can be added but not eq. (4.27), because the longest chains are order 5. Level 5 can still be constructed and contracted easily, but virtual level 3 has a bond dimension of 64. Compared to the previous bond dimensions (1,4 and 16) this is quite large and needs to be truncated.

6.4.5.2 Loops and extensions

?? Adding loops decreases the error. but there is also a surprising result: all possible loop extensions result in a higher error. The fluctuation increases drastically. With increasing bond dimension χ, this is somewhat better but still not good enough. The question is wether this is a result of a failing cluster expansion, or inability VUMPS to calculate the correct environment. During my thesis, my focus has largely been on the first case. After all, the framework was completely build from scratch and errors happen, and there is no guarantee that the series even converges. But introduction of very strict variants, such as the generalization to type E, where left/upper extension have level a and the right/lower extensions are of type a':



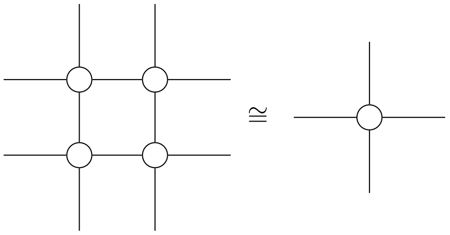
(6.7)

Introduce these variations.

The VUMPS producedure come with 2 implicit assumptions: the original version was derived for hermitian hamiltonians. The MPO resulting from the traced PEPO from the culster expansion is not an hermition MPO.

A second assumption made during this thesis is that the wave function can be represented by a 1by1 unit cell. Despite efforts made, the multisite version [5] doesn't seem to produce sensible results, even for version where 1by1 unit cell does give the right results.

It's clear that more research is needed here. One way to locate the problem would be to use another algorithm to contract the network, such as corner transfer matrix renormalization group (CTMRG) as used in citeCzarnik2019, or even using the single site VUMPS algorithm combined with blocking:



(6.8)

complex matrices instead of real ones??

6.4.5.3 Better extrapolation

The spirit behind finite size scaling is to calculate more with the data availalbe. In section 3.2.5, 2 additional vararations were suggested. One is to account for the subleading finite size corrections, the other to cange the way of calculating δ.

6.4.5.3.1 Subleading corrections Intruducing subleading corrections requires 4 parameters for each fitted universal function: ω,ϕ,c and d from eq. (3.4). Compared to the one parameter T_c or g_c, this is a lot and can be used to make many graphs collapse into a singlge graph. The analysis in [29] curefully estimates the error made with the fitting procedure. A similar amount of rigour would be required to fully trust the results.

Nevertheless, fitting with subleading corrections results in critical temperatures close to the ones fitted without, and the subleading exponents are close to 1, as expected from the subleading series expansion.

6.4.5.3.2 Choice δ Different choices of c_i in eq. (3.3) are possible to construct δ. Variantional optimisation was doen as suggested in [5]. Both the x and y axis should be normalised, because depending on δ the scale changes. If scaled based on the outer points, there is a flaw in the fitting procedure. The variational minimum goes to a point where at least for one point δ is extremely close to 0, and hence everything on the x axis except that point is pinched together, resulting in very small relative error. Therefore, it is better to normalise according to the points a e.g. 25th percentile and 75th percentile of the range on the axis.

This improves the collapse, but is not clear how much the prediction of T_c or g_c is improved.

6.4.6 Conclusion

6.5 Conclusion

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula.

Fusce mauris. Vestibulum luctus nibh at lectus. Sed bibendum, nulla a faucibus semper, leo velit ultricies tellus, ac venenatis arcu wisi vel nisl. Vestibulum diam. Aliquam pellentesque, augue quis sagittis posuere, turpis lacus congue quam, in hendrerit risus eros eget felis. Maecenas eget erat in sapien mattis porttitor. Vestibulum porttitor. Nulla facilisi. Sed a turpis eu lacus commodo facilisis. Morbi fringilla, wisi in dignissim interdum, justo lectus sagittis dui, et vehicula libero dui cursus dui. Mauris tempor ligula sed lacus. Duis cursus enim ut augue. Cras ac magna. Cras nulla. Nulla egestas. Curabitur a leo. Quisque egestas wisi eget nunc. Nam feugiat lacus vel est. Curabitur consectetuer.

Suspendisse vel felis. Ut lorem lorem, interdum eu, tincidunt sit amet, laoreet vitae, arcu. Aenean faucibus pede eu ante. Praesent enim elit, rutrum at, molestie non, nonummy vel, nisl. Ut lectus eros, malesuada sit amet, fermentum eu, sodales cursus, magna. Donec eu purus. Quisque vehicula, urna sed ultricies auctor, pede lorem egestas dui, et convallis elit erat sed nulla. Donec luctus. Curabitur et nunc. Aliquam dolor odio, commodo pretium, ultricies non, pharetra in, velit. Integer arcu est, nonummy in, fermentum faucibus, egestas vel, odio.

Sed commodo posuere pede. Mauris ut est. Ut quis purus. Sed ac odio. Sed vehicula hendrerit sem. Duis non odio. Morbi ut dui. Sed accumsan risus eget odio. In hac habitasse platea dictumst. Pellentesque non elit. Fusce sed justo eu urna porta tincidunt. Mauris felis odio, sollicitudin sed, volutpat a, ornare ac, erat. Morbi quis dolor. Donec pellentesque, erat ac sagittis semper, nunc dui lobortis purus, quis congue purus metus ultricies tellus. Proin et quam. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos hymenaeos. Praesent sapien turpis, fermentum vel, eleifend faucibus, vehicula eu, lacus.

Chapter 7

Conclusion and lookout

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis.

Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula.

Fusce mauris. Vestibulum luctus nibh at lectus. Sed bibendum, nulla a faucibus semper, leo velit ultricies tellus, ac venenatis arcu wisi vel nisl. Vestibulum diam. Aliquam pellentesque, augue quis sagittis posuere, turpis lacus congue quam, in hendrerit risus eros eget felis. Maecenas eget erat in sapien mattis porttitor. Vestibulum porttitor. Nulla facilisi. Sed a turpis eu lacus commodo facilisis. Morbi fringilla, wisi in dignissim interdum, justo lectus sagittis dui, et vehicula libero dui cursus dui. Mauris tempor ligula sed lacus. Duis cursus enim ut augue. Cras ac magna. Cras nulla. Nulla egestas. Curabitur a leo. Quisque egestas wisi eget nunc. Nam feugiat lacus vel est. Curabitur consectetuer.

Suspendisse vel felis. Ut lorem lorem, interdum eu, tincidunt sit amet, laoreet vitae, arcu. Aenean faucibus pede eu ante. Praesent enim elit, rutrum at, molestie non, nonummy vel, nisl. Ut lectus eros, malesuada sit amet, fermentum eu, sodales cursus, magna. Donec eu purus. Quisque vehicula, urna sed ultricies auctor, pede lorem egestas dui, et convallis elit erat sed nulla. Donec luctus. Curabitur et nunc. Aliquam dolor odio, commodo pretium, ultricies non, pharetra in, velit. Integer arcu est, nonummy in, fermentum faucibus, egestas vel, odio.

Sed commodo posuere pede. Mauris ut est. Ut quis purus. Sed ac odio. Sed vehicula hendrerit sem. Duis non odio. Morbi ut dui. Sed accumsan risus eget odio. In hac habitasse platea dictumst. Pellentesque non elit. Fusce sed justo eu urna porta tincidunt. Mauris felis odio, sollicitudin sed, volutpat a, ornare ac, erat. Morbi quis dolor. Donec pellentesque, erat ac sagittis semper, nunc dui lobortis purus, quis congue purus metus ultricies tellus. Proin et quam. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos hymenaeos. Praesent sapien turpis, fermentum vel, eleifend faucibus, vehicula eu, lacus.

Bibliography

[1] R. Sinatra, P. Deville, M. Szell, D. Wang, A. L. Barabási, A century of physics (oct 2015). arXiv:1612.00079, doi:10.1038/nphys3494.

[2] R. Orús, A practical introduction to tensor networks: Matrix product states and projected entangled pair states (oct 2014). arXiv:1306.2164, doi:10.1016/j.aop.2014.06.013.

[3] P. Corboz, Lecture 1: tensor network states, Tech. rep.

[4] S.-J. Ran, Tensor Network Contractions :Methods and Applications to Quantum Many-Body Systems, 2020.
URL http://www.springer.com/series/5304

[5] A. Nietner, B. Vanhecke, F. Verstraete, J. Eisert, L. Vanderstraeten, Efficient variational contraction of two-dimensional tensor networks with a non-trivial unit cell, Quantum 4 (2020). arXiv:2003.01142, doi:10.22331/Q-2020-09-21-328.

[6] M. Hauru, C. Delcamp, S. Mizera, Renormalization of tensor networks using graph independent local truncations, Tech. rep. arXiv:1709.07460v2.

[7] R. Orús, G. Vidal, Simulation of two-dimensional quantum systems on an infinite lattice revisited: Corner transfer matrix for tensor contractiondoi:10.1103/PhysRevB.80.094403.

[8] L. Vanderstraeten, J. Haegeman, F. Verstraete, Tangent-space methods for uniform matrix product states, SciPost Phys. Lect. Notes 7 (2019). doi:10.21468/SciPostPhysLectNotes.7.

[9] H. Nishimori, G. Ortiz, Elements of Phase Transitions and Critical Phenomena, Vol. 9780199577, Oxford University Press, 2011. doi:10.1093/acprof:oso/9780199577224.001.0001.

[10] G. Jaeger, The ehrenfest classification of phase transitions: Introduction and evolution, Archive for History of Exact Sciences 53 (1) (1998) 51–81. doi:10.1007/s004070050021.

[11] A. J. Beekman, L. Rademaker, J. van Wezel, An Introduction to Spontaneous Symmetry Breaking (sep 2019). arXiv:1909.01820, doi:10.21468/scipostphyslectnotes.11.

[12] B. Vanhecke, J. Haegeman, K. Van Acoleyen, L. Vanderstraeten, F. Verstraete, Scaling Hypothesis for Matrix Product States, Physical Review Letters 123 (25) (2019). arXiv:1907.08603, doi:10.1103/PhysRevLett.123.250604.

[13] K. S. D. Beach, L. Wang, A. W. Sandvik, Data collapse in the critical region using finite-size scaling with subleading corrections (may 2005). arXiv:0505194.
URL http://arxiv.org/abs/cond-mat/0505194

[14] P. Ginsparg, E. Brézin, J. Zinn-Justin, Applied Conformal Field Theory, Tech. rep. (1988).

[15] P. Calabrese, J. Cardy, Entanglement entropy and quantum field theory, Journal of Statistical Mechanics: Theory and Experiment (6) (2004). arXiv:0405152, doi:10.1088/1742-5468/2004/06/P06002.

[16] S. Sachdev, Quantum phase transitions, Physics World 12 (4) (1999) 33–38. doi:10.1088/2058-7058/12/4/23.
URL https://iopscience.iop.org/article/10.1088/2058-7058/12/4/23https://iopscience.iop.org/article/10.1088/2058-7058/12/4/23/meta

[17] A. Taroni, Statistical physics: 90 years of the Ising model (dec 2015). doi:10.1038/nphys3595.
URL www.nature.com/naturephysics

[18] S. Hesselmann, S. Wessel, Thermal Ising transitions in the vicinity of two-dimensional quantum critical points, PHYSICAL REVIEW B 93 (2016) 155157. doi:10.1103/PhysRevB.93.155157.

[19] T. H. Hsieh, From d-dimensional Quantum to d + 1-dimensional Classical Systems, Student review (2) (2015) 1–4.

[20] S. Paeckel, T. Köhler, A. Swoboda, S. R. Manmana, U. Schollwöck, C. Hubig, Time-evolution methods for matrix-product states, Annals of Physics 411 (2019). arXiv:1901.05824, doi:10.1016/j.aop.2019.167998.

[21] M. P. Zaletel, R. S. K. Mong, C. Karrasch, J. E. Moore, F. Pollmann, Time-evolving a matrix product state with long-ranged interactions, PHYSICAL REVIEW B 91 (2015) 165112. doi:10.1103/PhysRevB.91.165112.

[22] B. Vanhecke, L. Vanderstraeten, F. Verstraete, Symmetric cluster expansions with tensor networks, Physical Review A 103 (2) (2021). arXiv:1912.10512, doi:10.1103/PhysRevA.103.L020402.

[23] Q. Zhao, G. Zhou, S. Xie, L. Zhang, A. Cichocki, Tensor Ring Decomposition (2016). arXiv:1606.05535.
URL http://arxiv.org/abs/1606.05535

[24] P. Moylan, Qr factorisation and pseudoinverse of rank- deficient matrices, ftp://pmoylan.org/papers/90.QR

[25] J. Matt, N-dimensional sparse arrays.
URL https://www.mathworks.com/matlabcentral/fileexchange/29832-n-dimensional-sparse-arrays

[26] C. Moler, C. Van Loan, Nineteen dubious ways to compute the exponential of a matrix, twenty-five years later (2003). doi:10.1137/S00361445024180.

[27] A. Wietek, A. M. Läuchli, Sublattice coding algorithm and distributed memory parallelization for large-scale exact diagonalizations of quantum many-body systems, Physical Review E 98 (3) (2018). arXiv:1804.05028, doi:10.1103/PhysRevE.98.033309.

[28] P. Czarnik, P. Corboz, Finite correlation length scaling with infinite projected entangled pair states at finite temperature, Physical Review B 99 (2019) 245107. doi:10.1103/PhysRevB.99.245107.

[29] L. Wang, K. S. Beach, A. W. Sandvik, High-precision finite-size scaling analysis of the quantum-critical point of S=1 2 Heisenberg antiferromagnetic bilayers, Physical Review B - Condensed Matter and Materials Physics 73 (1) (2006). arXiv:0509747, doi:10.1103/PhysRevB.73.014431.