Machine learning approaches to strongly correlated spin systems

Tom Vieijra

Ghent University

GHENT
UNIVERSITY

Department of Physics and Astronomy
Theoretical Nuclear and Statistical Physics

Machine learning approaches to strongly correlated spin systems

Tom Vieijra

1. Promotor Prof. Dr. Jan Ryckebusch
Department of Physics and Astronomy
Ghent University

2. Supervisor Dr. Jannes Nys
Department of Physics and Astronomy
Ghent University

3. Supervisor Corneel Casert
Department of Physics and Astronomy
Ghent University

Thesis submitted in partial fulfillment for the degree
of Master of Science in Physics and Astronomy

FACULTY
OF SCIENCES

May 31, 2018

Tom Vieijra
Machine learning approaches to strongly correlated spin systems
Documentation, May 31, 2018
Reviewers: Prof. Dr. Jan Ryckebusch, Dr. Jannes Nys, Prof. Dr. Jutho Haegeman and Prof. Dr. Wesley De Neve
Supervisors: Prof. Dr. Jan Ryckebusch, Dr. Jannes Nys and Corneel Casert

Ghent University
Theoretical Nuclear and Statistical Physics
Department of Physics and Astronomy
Proeftuinstraat 86, building N3
B-9000 Ghent

Abstract

Understanding systems of many degrees of freedom forms one of the most complex problems in physics. These systems can be described by only taking their macroscopic properties into account (thermodynamics). Another way of getting insight in these systems is examining how the individual degrees of freedom give rise to the macroscopic properties. This branch of physics is called many-body physics. Finding how the macroscopic properties arise from the collection of individual degrees of freedom requires a way to connect microscopic and macroscopic properties of the system under study. How to make this connection is formally described by statistical physics and quantum many-body physics. However, making this connection in practice often requires approximations to the applicable physical laws and a substantial amount of computation. Numerous approximation methods have been developed in the physics community. Recently, a new toolbox was added to the body of approximation methods, namely machine learning. Machine learning techniques have the ability to make the connection between data and abstract concepts in a scalable way. In essence they are able to make the connection between microscopic variables (e.g. pixels in an image) and macroscopic concepts (e.g. the object contained in the image).

In this thesis, the connection between many-body physics and machine learning is first examined (chapters 1, 2 and 3). After giving the motivation to use machine learning techniques for problems in many-body physics, we proceed in chapter 4 with a specific application of it on the problem of finding the ground state of quantum many-body spin systems. Specifically, we model the ground-state wave function of the transverse field Ising and antiferromagnetic Heisenberg models, both in one and two spatial dimensions. This is done with the aid of a restricted Boltzmann machine (which is a stochastic neural network) to model the probability amplitudes in the expansion of the wave function. We examine the accuracy and the scaling of the technique, as well as the internal representation of the restricted Boltzmann machine and its relation to physical quantities. In chapter 5, we examine whether the restricted Boltzmann machine is capable of representing ground-state wave functions at or near a quantum phase transition. The motivation to study this is the fact that the wave function contains the largest amount of non-trivial correlations at the transition point between two phases, which makes it significantly harder to

describe. We perform a finite-size scaling analysis for the one-dimensional transverse field Ising system to determine the critical point and the critical exponents, and examine different relevant observables of the macroscopic system (e.g. the Binder cumulant and the correlation functions). We also compare our results with the results of other techniques available in literature.

Acknowledgement

Allereerst zou ik Jannes en professor Ryckebusch willen bedanken voor de opportuniteit die jullie mij gegeven hebben om deze thesis te maken. Jullie enthousiasme over dit onderwerp gaf mij telkens een boost om verder te werken aan hetgeen in de volgende bladzijden te lezen staat. Bedankt voor jullie kennis van zaken en constructieve feedback op mijn werk.

Andres and Corneel, thank you for helping me with the servers and other computer-related problems, and the coffee afterwards. Also thanks to the other people at INW for bumping into each other and the interesting things you had to say.

Mede-thesisstudenten, een welgemeende dankjulliewel voor de lunches, de dilemma's en het kletsen als pauze.

Hilde Van Oostenryck en Hilde Verheijen van het Xaveriuscollege in Borgerhout, bedankt voor jullie enthousiasme voor wiskunde en wetenschappen. Zonder jullie had ik hier nooit gestaan. Jullie speelden een ongelooflijk belangrijke rol in het kiezen van deze studierichting, waar ik geen moment spijt van heb gehad.

Vrienden van 't stad, bedankt voor de koffie en het luisteren naar mijn oneindige stroom aan fysica-praatjes.

Mama, papa en Michiel, bedankt voor jullie vertrouwen, de kansen die jullie mij gegeven hebben en het fietsen.

Annelies, bedankt om er altijd voor mij te zijn en mijn doemscenario's telkens weer te ontkrachten. Je positivisme bracht me telkens weer op het juiste spoor. Zonder jou was ik waarschijnlijk al lang ergens gestrand in een onmetelijk leeg veld of op een verlaten eiland.

Contents

1 Introduction to many-body physics

Many-body physics deals with physical problems with a large amount of interacting degrees of freedom. In this definition, large can have different meanings, ranging from the amount of electrons present in atoms ($\sim 10 - 100$), the amount of atoms present in nanomaterials ($\sim 10^6$) or the amount of atoms involved in macroscopic samples of matter ($\sim 10^{23}$). The physical problem is to connect the properties of the microscopic degrees of freedom to the macroscopic behaviour of the collective. Solving these problems is important both from a theoretical and experimental point of view. On the theoretical side, one is interested in the underlying mechanisms that generate the macroscopic properties of systems of many interacting constituents. From an experimental point of view, one uses the connection between microscopic and macroscopic properties to develop and engineer new materials with valuable and desirable characteristics.

Finding how the interactions between the degrees of freedom give rise to macroscopic properties is a challenging problem. The interactions between the constituents generate complexity in the system. This means that the physics cannot be solved by considering the behaviour of one degree of freedom and treating the system as a collection of them, independent of each other. While the occurrence of these so-called correlations makes these systems hard to deal with, it is also the reason for their elusive properties. Examples of emergent properties are long-range order, topological properties and phase transitions. Another way to coin this is the famous phrase: “The whole is more than the sum of its parts”.

The many-body problem can be approximately solved with a number of techniques. For classical microscopic degrees of freedom, statistical physics can be used.[1] For quantum mechanical degrees of freedom, quantum statistical physics can be used for problems at finite temperature and quantum many-body physics can be used for problems at zero temperature. In this work, we use tools from both statistical and quantum many-body physics.

[1] Statistical physics can also be used to describe quantum mechanical systems, as long as the eigenspectrum of the energy is known.

1.1 Classical statistical physics

Statistical physics describes systems with many degrees of freedom from a statistical point of view. To this end, one defines a probability distribution over the possible microscopic configurations of the system. This probability distribution can then be used to find the macroscopic properties of the system as expectation values over the probability distribution. To define the probability distribution, one must define some properties of the macroscopic system. For example, the system can be viewed at constant temperature T or at constant energy E. Other variables that determine the behaviour of the system are pressure P or volume V, and the number of degrees of freedom N or the chemical potential μ. These variables are called control variables; they represent the conditions under which the system is studied. The combination of the control variables defines an ensemble, for which one can find the probability distribution. A common ensemble is the (N, V, T), or canonical, ensemble where the number of degrees of freedom, the volume and the temperature are kept constant.

A common example of a non-trivial system in statistical physics is the Ising model, which is a model for (ferro)magnetism. The degrees of freedom are N spins $\{s^i\}_{i=1\ldots N}$ which are spatially localized on lattice sites. The spins can have spin projections $s^i_z = -1/2$ or $s^i_z = 1/2$ in the s_z-basis. The microscopic configurations of the system are thus the different combinations of the spin values in the system. For this system the control variables are $(N, h_\parallel, T)$, where $h_\parallel$ is the magnetic field longitudinal to the z-direction. To define the probability distribution for this system, we define an energy function. In the Ising model, this is defined as [1]

$$E(\mathcal{S}) = -j \sum_{\langle i,k \rangle} s^i_z s^k_z - h_\parallel \sum_{i=1}^{N} s^i_z, \tag{1.1}$$

where, $\mathcal{S} \equiv \{s^i_z\}_{i=1\ldots N}$ is a specific configuration of the spin projections and $\sum_{\langle i,k \rangle}$ is a sum over all pairs of nearest neighbours. Further, j is the magnitude of the nearest neighbour interactions, where $j > 0$ leads to ferromagnetic interactions (the system has a lower energy when spins are aligned) and $j < 0$ leads to antiferromagnetic interactions (the system has a lower energy when spins are antialigned). The function of Eq. (1.1) defines an energy value for all the different configurations of the system. The probability in the $(N, h_\parallel, T)$ ensemble for a spin configuration $\mathcal{S}$ is then given as

$$p_{(N,h_\parallel,T)}(\mathcal{S}) = \frac{\exp\left(-\frac{E(\mathcal{S})}{k_B T}\right)}{\sum_{\mathcal{S}'} \exp\left(-\frac{E(\mathcal{S}')}{k_B T}\right)} = \frac{\exp\left(-\frac{E(\mathcal{S})}{kT}\right)}{Z(N, h_\parallel, T)}. \tag{1.2}$$

Here, k_B is the Boltzmann constant and Z is the partition function. The partition function is central for statistical physics as many properties of the system can be derived from it. Unfortunately, the sum over the different configurations is often intractable and can only be computed exactly in a very limited amount of cases. To solve this, one needs to resort to computational approaches such as Markov Chain Monte Carlo [2] or approximating techniques such as cluster expansions [3].

A nice feature of statistical physics is that it can be used outside of physics for problems with many interacting degrees of freedom. Examples are social systems [4], flocking [5], epidemiology [6] and notably machine learning [7].

1.2 Quantum many-body physics

Quantum mechanics is the theory which describes physical degrees of freedom on microscopic scales. Examples are the physics of atoms and molecules, and the physics of subatomic particles. Quantum many-body physics seeks to compute the zero-temperature properties of quantum mechanical systems with many degrees of freedom, defined by a Hamiltonian $\hat{H}$, at zero temperature (see for example [8]). The properties of quantum mechanical systems are encoded in states $|\psi\rangle$, living in a Hilbert space $\mathcal{H}$. These states are conventionally called quantum states or wave functions. For a system of N distinguishable[2] degrees of freedom (e.g. when they are spatially localized on lattice sites), the Hilbert space of the system is the direct product of the Hilbert spaces of the separate degrees of freedom:

$$\mathcal{H} = \mathcal{H}_1 \otimes \mathcal{H}_2 \otimes ... \otimes \mathcal{H}_N, \tag{1.3}$$

where $\mathcal{H}_i$ is the Hilbert space of the i-th degree of freedom. This shows that quantum mechanical problems with many degrees of freedom suffer from the curse of dimensionality, as the dimension D of the many-body Hilbert space scales exponentially with N as

$$D = \dim(\mathcal{H}) = \prod_{i=1}^{N} \dim(\mathcal{H}_i). \tag{1.4}$$

In principle, the non-relativistic many-body problem is fully solved upon integrating the time independent Schrödinger equation (for systems where $\hat{H}$ is not explicitly time dependent) for the energy eigenvalues and eigenfunctions

$$\hat{H} |\psi_n\rangle = E_n |\psi_n\rangle , \tag{1.5}$$

[2] For indistinguishable degrees of freedom, there are certain relations the states should obey (e.g. the states need to be antisymmetric under exchange of two degrees of freedom in fermionic systems), making the effective size of the Hilbert space of possible physical states smaller. In this work, we only work with distinguishable degrees of freedom.

where E_n is the energy eigenvalue belonging to the n-th eigenfunction $|\psi_n\rangle$. After choosing an appropriate basis $\{|i\rangle\}_{i=1\ldots D}$, the eigenvalue equation reduces to a matrix diagonalization problem of the Hamiltonian matrix $\mathbf{H}$, which is defined as

$$[\mathbf{H}]_{ij} = \langle i|\hat{H}|j\rangle\,. \tag{1.6}$$

However, the dimension of the matrix grows exponentially with the number of degrees of freedom. The dimension of the matrix equals the dimension of the Hilbert space, which scales as Eq. (1.4). Diagonalizing a matrix with dimension D generally scales as $\mathcal{O}(D^3)$ [9]. Unless the matrix has some property which speeds up the diagonalization algorithm (diagonal, block-diagonal, very sparse, ...), one has to resort to approximation techniques such as variational wave functions [2], quantum Monte Carlo [10] or perturbation theory [8].

In this thesis, we will work with quantum spin systems on a lattice, where the number of degrees of freedom N is fixed. For spins with total spin $s = 1/2$, the Hilbert space of a single degree of freedom has dimension 2. We can define the Hilbert space in terms of the eigenstates of the $\hat{s}_z$-operator (which is the spin-projection operator on the z-axis) , denoted as

$$\left|s_z = +\frac{1}{2}\right\rangle = \begin{pmatrix}1\\0\end{pmatrix}, \qquad \left|s_z = -\frac{1}{2}\right\rangle = \begin{pmatrix}0\\1\end{pmatrix}. \tag{1.7}$$

The spin operators in the three Cartesian directions are related to the Pauli matrices via $\hat{s}_x = \frac{1}{2}\sigma_x$, $\hat{s}_y = \frac{1}{2}\sigma_y$, $\hat{s}_z = \frac{1}{2}\sigma_z$. In the basis defined in Eq. (1.7), the Pauli-operators are defined as

$$\sigma_x = \begin{pmatrix}0 & 1\\1 & 0\end{pmatrix}, \quad \sigma_y = \begin{pmatrix}0 & -i\\i & 0\end{pmatrix}, \quad \sigma_z = \begin{pmatrix}1 & 0\\0 & -1\end{pmatrix}. \tag{1.8}$$

These operators satisfy the commutation relations

$$[\sigma_a, \sigma_b] = 2i\epsilon_{abc}\sigma_c, \tag{1.9}$$

where ϵ_{abc} is the Levi-Civita tensor. The anti-commutation relations for the Pauli-operators are

$$\{\sigma_a, \sigma_b\} = 2\delta_{ab}\hat{I}, \tag{1.10}$$

where δ_{ab} is the Kronecker delta and $\hat{I}$ is the identity operator. From the definition of σ_z in Eq. (1.8), we see that the states in Eq. (1.7) have eigenvalues $+1$ and -1 respectively. Due to this fact, the first state in Eq. (1.7) is called the spin up state $|\uparrow\rangle$ and the second one the spin down state $|\downarrow\rangle$. A state in a many-body Hilbert space can be written as a direct product of the states in the Hilbert spaces of the individual

degrees of freedom. For a $\{s^i = 1/2\}_{i=1...N}$ many-body spin system, we can define a basis for its Hilbert space as the direct product of the basis states in Eq. (1.7)

$$|s_z^1 s_z^2 ... s_z^N\rangle = |s_z^1\rangle \otimes |s_z^2\rangle \otimes ... \otimes |s_z^N\rangle\,, \tag{1.11}$$

where $|s_z^i\rangle$ is the basis state of the i-th degree of freedom. In this work, Hamiltonians operating on spin degrees of freedom will be written in terms of the Pauli matrices. The transition between spin operators and Pauli matrices will not be given explicitly. The constant factors relating the Pauli matrices to the spin operators will be either decimated via a rescaling of the energy or absorbed in prefactors. For example

$$\hat{H} = c\hat{s}_x^1 \hat{s}_x^2 \rightarrow \tilde{c}\sigma_x^1 \sigma_x^2, \tag{1.12}$$

where $\tilde{c} = c/4$. When writing an operator depending on a single degree of freedom i, for example $\hat{s}_x^i$ in Eq. (1.12), we implicitly mean that it only operates non-trivially on degree of freedom i, i.e.

$$\hat{s}_x^i \equiv \hat{I}^1 \otimes ... \otimes \hat{I}^{i-1} \otimes \hat{s}_x^i \otimes \hat{I}^{i+1} \otimes ... \otimes \hat{I}^N, \tag{1.13}$$

where $\hat{I}^j$ is the unit operator acting on degree of freedom j.

Quantum many-body physics shares many properties with statistical physics. To show this, we will examine the transverse field Ising model (TFI). This model is defined in the same way as the Ising model of Eq. (1.1), with the addition of a transverse magnetic field and no magnetic field parallel to the spins. The described situation can be modelled with the following Hamiltonian

$$\hat{H} = \hat{H}_z + \hat{H}_x = -j \sum_{\langle i,k \rangle} \sigma_z^i \sigma_z^k - h \sum_{i=1}^{N} \sigma_x^i. \tag{1.14}$$

Here, σ_z^i and σ_x^i are the Pauli matrices of Eq. (1.7) operating only on the spin state of the i-th spin. Further, j is the magnitude of the nearest-neighbour interactions and h is the magnitude of the external transverse field. We will denote the ratio of h and j as g

$$g \equiv \frac{h}{j}. \tag{1.15}$$

Note the resemblance with the classical Ising model. One can raise the question of what a quantum mechanical problem at zero temperature has to do with a classical problem at non-zero temperature, apart from the fact that many degrees of freedom are involved. One way to connect these two areas of physics is with the concept of uncertainty. We noticed in section 1.1 that temperature introduces uncertainty in Nature due to the fact that it introduces a probability distribution for the different configurations. Another mechanism to generate uncertainty is

superposition in quantum mechanics. Superposition means that, given a specific basis for the problem at hand, the quantum states which solve the Schrödinger equation are a linear combination of these basis states. For example, in the TFI problem we can choose the eigenbasis of the σ^z-operators of Eq. (1.11), yielding as a basis the configurations consisting of all possible combinations of up and down spins. Superposition arises when the Hamiltonian contains contributions which do not commute with each other. In the TFI model, this is the case because $[\hat{H}_z, \hat{H}_x] \neq 0$ due to Eq. (1.9). A consequence of superposition is that expectation values of a given observable, for example the magnetization in the z-direction, become weighted sums for the different basis functions. This is just like expectation values in statistical physics are weighted sums of the observable for the different configurations. While in statistical physics the uncertainty is introduced by the surroundings of the system, the uncertainty in quantum mechanics is an inherent part of the theory. This parallelism is striking. Differences remain as both settings have additional features not found in the other, such as entanglement in quantum mechanics and the ergodic hypothesis in statistical physics.

1.3 Emergence in many-body physics

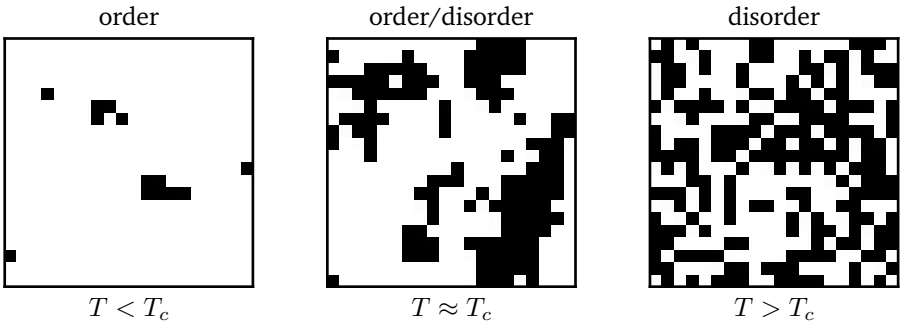

Fig. 1.1: Depiction of the phases in the 2D classical Ising model of Eq. (1.1) (with $h_{\parallel} = 0$) and the transition between them. The spins on the square lattice are depicted as black or white squares, where black squares have $s_z^i = 1/2$ and white squares have $s_z^i = -1/2$. The phase transition occurs when the temperature reaches T_c.

As stated in the beginning of this chapter, many-body systems have some features which are not found in the physics of the individual constituents. This property of many-body physics is often referred to as *emergence*. For example, phase transitions are one of the most notable manifestations of emergent properties of physical systems. The emergence of phases and transition between them in a system of many interacting degrees of freedom is a feature that cannot be understood starting from the individual degrees of freedom. It is the result of the subtle interplay between two mechanisms with opposing effects which is referred to as frustration. In statistical

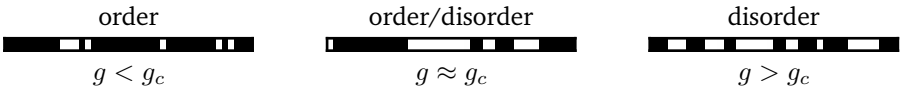

Fig. 1.2: Schematic representation of the phases of the 1D Transverse field Ising model of Eq. (1.14) and the transition between them. The black bars denote spins in state $|\uparrow\rangle$ and the white bars spins in state $|\downarrow\rangle$. The transition occurs when $g = 1$ (see section 4.4.1). Note the qualitative resemblance with figure 1.1.

physics, the equilibrium state of the system is defined by the minimization of the free energy F

$$F = E - TS. \tag{1.16}$$

In this equation, E is the energy, T is the temperature and S is the entropy. Minimizing the free energy comes down to energy minimization and entropy maximization. These two operations have an influence on each other as energy minimization tends to also minimize the entropy and energy maximization tends to maximize the entropy. The upshot is a state of the system in between these two extremes, governed by the value of the temperature. When the temperature is low, energy minimization tends to play a more important role. In the Ising system introduced in Eq. (1.14), this boils down to a state where almost all the spins are aligned, i.e. an ordered state. When the temperature is high, the entropy maximization becomes the dominant contribution to F and therefore a disordered state is favoured. In between these extremes, the system undergoes a transition. The transition described here is depicted in figure 1.1

In the spirit of the relation between statistical physics and quantum many-body physics, phase transitions can also be found in (the ground state of) quantum many-body systems. Under these circumstances, energy minimization leads to frustration. Note that the Hamiltonian of the transverse field Ising model of Eq. (1.14) consists of two distinct terms. Suppose that $h = 0$, we see easily that the state which minimizes the $\hat{H}_z$-part of the energy is the state with all spins aligned. Suppose now that $h \neq 0$. The term involving σ_x-operators does not commute with the term involving σ_z-operators. The quantum state which minimizes the $\hat{H}_z$-part of the Hamiltonian is not an eigenstate of the $\hat{H}_x$-part of the Hamiltonian. It follows that this state is not the ground state because the ground state is an eigenstate of the full Hamiltonian. The ground state is thus not a state which minimizes both parts of the Hamiltonian independently. This generates frustration as the true ground state compromises between opposing terms in the Hamiltonian. Unlike in the classical case, it is not temperature which generates transitions between the two phases, but rather the

strength of the transverse field h. When h tends to zero, the resulting state will lie close to the eigenstate of the $\hat{H}_z$-part of the Hamiltonian, i.e. with all spins up or down. When h is large, the resulting state will lie close to the eigenstates of the $\hat{H}_x$-part, i.e. states with zero magnetization. In between, a phase transition occurs. The phase transition in the 1D transverse field Ising model is schematically depicted in figure 1.2.

1.4 Quantum statistical physics

In section 1.1, we introduced statistical physics for systems for which the energy values of the different configurations are known. This enables one to solve temperature dependent many-body problems. In section 1.2, quantum mechanical problems with many interacting degrees of freedom were introduced. However, the discussion was limited to physics at zero temperature. Combining both areas of physics leads to quantum statistical physics, where temperature dependent quantum many-body problems are treated. To develop quantum statistical physics, it is necessary to express a classical probability distribution over a collection of quantum states. This is most easily done in the density operator formalism of quantum mechanics (see for example [11]). A density operator $\hat{\rho}$ is a non-negative operator with trace one, defined on the Hilbert space of the system. This operator describes the state of the system. For example, the density operator of a single state $|\Psi\rangle$ is

$$\hat{\rho} = |\Psi\rangle \langle\Psi| . \tag{1.17}$$

The density operator allows to represent both single states and probability distributions over multiple states with the same object. The expectation value of an operator $\hat{A}$ is given by

$$\mathrm{Tr}(\hat{A}\hat{\rho}). \tag{1.18}$$

A system with Hamiltonian $\hat{H}$ in the (N, V, T) ensemble is defined by the density operator [12]

$$\hat{\rho}_{(N,V,T)} = \frac{\exp\left(-\frac{\hat{H}}{kT}\right)}{\mathrm{Tr}\left(\exp\left(-\frac{\hat{H}}{kT}\right)\right)} = \frac{\exp\left(-\frac{\hat{H}}{kT}\right)}{Z(N,V,T)}. \tag{1.19}$$

Here, the exponential of an operator $\hat{A}$ is defined as

$$\exp(\hat{A}) = \sum_{i=0}^{\infty} \frac{1}{i!} \hat{A}^i, \tag{1.20}$$

where $\hat{A}^0 = \hat{I}$. The denominator of Eq. (1.19) ($Z(N,V,T)$) is called the partition function, analogously to Eq. (1.2). Note that Eq. (1.19) reduces to Eq. (1.2) when the Hamiltonian is represented in its eigenbasis.

In section 1.2, a qualitative relation between statistical physics at non-zero temperature and quantum many-body physics at zero temperature was given. This relationship can be made more rigorous using quantum statistical physics [12, 13]. Considering the 1D transverse field Ising model of Eq. (1.14), the partition function for this Hamiltonian can be written down:

$$Z_{TFI} = \mathrm{Tr}(\exp(-\beta \hat{H}_{TFI})), \tag{1.21}$$

where $\beta = 1/k_B T$. Calculating this trace is non-trivial as $\hat{H}$ is not diagonal. The energy of the ground state E_{gs} of this system corresponds with

$$E_{gs} = \lim_{T \to 0} -k_B T \ln\left(Z_{TFI}\right). \tag{1.22}$$

We introduce the dimensionless quantity

$$m \equiv h/k_B T = h\beta, \tag{1.23}$$

where h is the coupling of the transverse field in Eq. (1.14). The limit $T \to 0$ corresponds to $m \to \infty$. In the following, we restrict m to integer values. This has no effect on the analysis because the limit of m to infinity is taken.

Using Eq. (1.23) in Eq. (1.22) yields

$$E_{gs} = \lim_{m \to \infty} -\frac{h}{m} \ln\left(\mathrm{Tr}\left(\exp\left(-\frac{m}{h}\hat{H}_{TFI}\right)\right)\right). \tag{1.24}$$

The operator

$$\exp\left(-\frac{m}{h}\hat{H}_{TFI}\right) \tag{1.25}$$

can be approximated by applying the Suzuki-Trotter decomposition [14],

$$\exp\left(\sum_i \hat{A}_i\right) = \lim_{n \to \infty} \left(\prod_i \exp(\hat{A}_i/n)\right)^n. \tag{1.26}$$

By writing the argument of the exponential in Eq. (1.25) as a sum over m identical terms $-\hat{H}_{TFI}/h$, applying the Suzuki-Trotter decomposition on Eq. (1.24) yields

$$E_{gs} = \lim_{m \to \infty} \lim_{n \to \infty} -\frac{h}{m} \ln\left(\mathrm{Tr}\left(\left(\prod_{i=1}^{m} \exp\left(-\frac{1}{hn}\hat{H}_{TFI}\right)\right)^n\right)\right). \tag{1.27}$$

To evaluate the trace, we will work in the σ_z-basis. Using the notation $|S_z^i\rangle \equiv |s_z^1 s_z^2 ... s_z^N\rangle$ for a specific basis state, this yields

$$E_{gs} = \lim_{m\to\infty}\lim_{n\to\infty} -\frac{h}{m}\ln\left(\sum_{\{S_z^i\}}\left\langle S_z^i\right|\left(\prod_{i=1}^{m}\exp\left(-\frac{1}{hn}\hat{H}_{TFI}\right)\right)^n\left|S_z^i\right\rangle\right), \quad (1.28)$$

where $\sum_{\{S_z^i\}}$ denotes a sum over all basis states (or equivalently spin configurations in the σ_z-basis). By inserting the identity operator $\hat{I} = \sum_{\{S_z^i\}}|S_z^i\rangle\langle S_z^i|$ between every subsequent pair of exponential operators, we get for the trace in Eq. (1.28) (denoting with the second index of S the place of the identity operator, where $S_z^{i,1}$ is used for the bra and ket of the trace)

$$\sum_{\{S_z^{i,1}\}}\left\langle S_z^{i,1}\right|\left(\prod_{i=1}^{m}\exp\left(-\frac{1}{hn}\hat{H}_{TFI}\right)\right)^n\left|S_z^{i,1}\right\rangle = \sum_{\{S_z^{i,j}\}}\left\langle S_z^{i,1}\right|\exp\left(-\frac{1}{hn}\hat{H}_{TFI}\right)\left|S_z^{i,2}\right\rangle$$
$$\left\langle S_z^{i,2}\right|\exp\left(-\frac{1}{hn}\hat{H}_{TFI}\right)\left|S_z^{i,3}\right\rangle ... \left\langle S_z^{i,nm}\right|\exp\left(-\frac{1}{hn}\hat{H}_{TFI}\right)\left|S_z^{i,1}\right\rangle. \quad (1.29)$$

The expectation values in Eq. (1.29) can now be calculated, using the definition of $\hat{H}_z$ and $\hat{H}_x$ in Eq. (1.14). Denoting $|S_z\rangle \equiv |s_z^1 s_z^2 ... s_z^n\rangle$ and $|S'_z\rangle \equiv |s'^1_z s'^2_z ... s'^n_z\rangle$, these yield

$$\left\langle S_z\right|\exp\left(-\frac{1}{hn}\hat{H}_{TFI}\right)\left|S'_z\right\rangle =$$
$$\prod_{k=1}^{N}\exp\left(\frac{j}{hn}s_z^k s_z^{k+1}\right)\left(\frac{1}{2}\sinh\left(\frac{2}{n}\right)\right)^{1/2}\exp\left(\frac{1}{2}s_z^k s'^k_z\ln\left(\coth\left(\frac{1}{n}\right)\right)\right). \quad (1.30)$$

Using this in Eq. (1.29) yields

$$\sum_{\{S_z^i\}}\left\langle S_z^i\right|\left(\prod_{i=1}^{m}\exp\left(-\frac{1}{hn}\hat{H}_{TFI}\right)\right)^n\left|S_z^i\right\rangle =$$
$$\sum_{\{s_z^{i,j}\}}\prod_{k=1}^{N}\prod_{l=1}^{nm}\left(\frac{1}{2}\sinh\left(\frac{2}{n}\right)\right)^{1/2}\exp\left(\frac{j}{hn}s_z^{k,l}s_z^{k+1,l}+\frac{1}{2}\ln\left(\coth\left(\frac{1}{n}\right)\right)s_z^{k,l}s_z^{k,l+1}\right), \quad (1.31)$$

where the sum $\sum_{\{S_z^{i,j}\}}$ over basis states is replaced by the sum $\sum_{\{s_z^{i,j}\}}$ over spin configurations, which is equivalent.

Using Eq. (1.31) in the expression for E_{gs} of Eq. (1.28), we obtain

$$E_{gs} = \lim_{m\to\infty}\lim_{n\to\infty} -\frac{h}{m}\ln\left(\left(\frac{1}{2}\sinh\left(\frac{2}{n}\right)\right)^{Nmn/2} \sum_{\{s_z^{i,j}\}} \exp\left(\sum_{k=1}^{N}\sum_{l=1}^{nm}\left(\frac{j}{hn}s_z^{k,l}s_z^{k+1,l} + \frac{1}{2}\ln\left(\coth\left(\frac{1}{n}\right)\right)s_z^{k,l}s_z^{k,l+1}\right)\right)\right), \tag{1.32}$$

where the products $\prod_{k=1}^{N}\prod_{l=1}^{nm}$ have been cast to sums in the exponential. We see that the ground state energy written as in Eq. (1.32) corresponds to the free energy of a classical Ising system with anisotropic interactions in two dimensions, where h plays the role of temperature. The partition function Z_{AI} of this system is

$$Z_{AI} = \sum_{\{s_z^{i,j}\}} \exp\left(\sum_{k=1}^{N}\sum_{l=1}^{nm}\left(\frac{j}{hn}s_z^{k,l}s_z^{k+1,l} + \frac{1}{2}\ln\left(\coth\left(\frac{1}{n}\right)\right)s_z^{k,l}s_z^{k,l+1}\right)\right). \tag{1.33}$$

The analysis above was applied for the specific Hamiltonian of Eq. (1.14). However, the formal analogy between the ground state of a d-dimensional quantum system and a $(d+1)$-dimensional finite temperature classical system can be proven to hold for a general Hamiltonian [13].

2 Introduction to machine learning

Machine learning stands for a class of computational techniques which aim at performing a specified task, given some input and given some guidance of how the task can be performed. Tom M. Mitchell provided the following formal definition: *"A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P if its performance at tasks in T, as measured by P, improves with experience E"* [15]. Machine learning spans a broad range of techniques. Examples are

- Dimensionality reduction techniques such as Principal Component Analysis (PCA) and t-SNE, which search for a transformation of the data that reduces the dimensionality of the system (see section 2.3.1).

- Function approximation techniques which try to approximate a function $f(x)$ with $\tilde{f}(x)$, typically by tuning some well-defined parameters in $\tilde{f}$ (see chapter 4 for an example where it is used on the wave function of a many-body quantum mechanical spin system).

- Clustering techniques such as K-means [16], which are designed to find a meaningful partitioning of the data in subsets.

- Classification techniques such as support vector machines (see section 2.3.2) and neural networks (see section 2.3.3), which try to perform a classification of data samples in classes defined beforehand by the user.

- Probability distribution estimation such as generative adversarial networks (see section 2.3.4) and Markov Models [17], which are designed to approximate a stochastic probability distribution. This is closely related to function approximation.

Machine learning is used in an exponentially increasing number of settings. Due to the ever-increasing power of computers, machine learning techniques have become more accessible and have been adopted in many different branches of society. Some examples are game playing [18], speech and pattern recognition [19], optimization [20], medical diagnosis [21], fraud detection [22] and forecasting [23]. All these

applications have in common that it is generally hard to translate the problem in a sequence of logical steps, as would be required for the development of a computer code for the given problem. Machine learning techniques have the capability of extracting features of the data (e.g. the shape of a car in a picture) and make abstraction of the irrelevant details in the data. For example a picture containing a sportscar looks different compared to a picture containing a SUV. A machine learning algorithm can make abstraction of this difference and detect in both cases that the picture contains a car. In this section, machine learning will be introduced and a broad overview of this body of techniques will be given.

2.1 The machine learning strategy

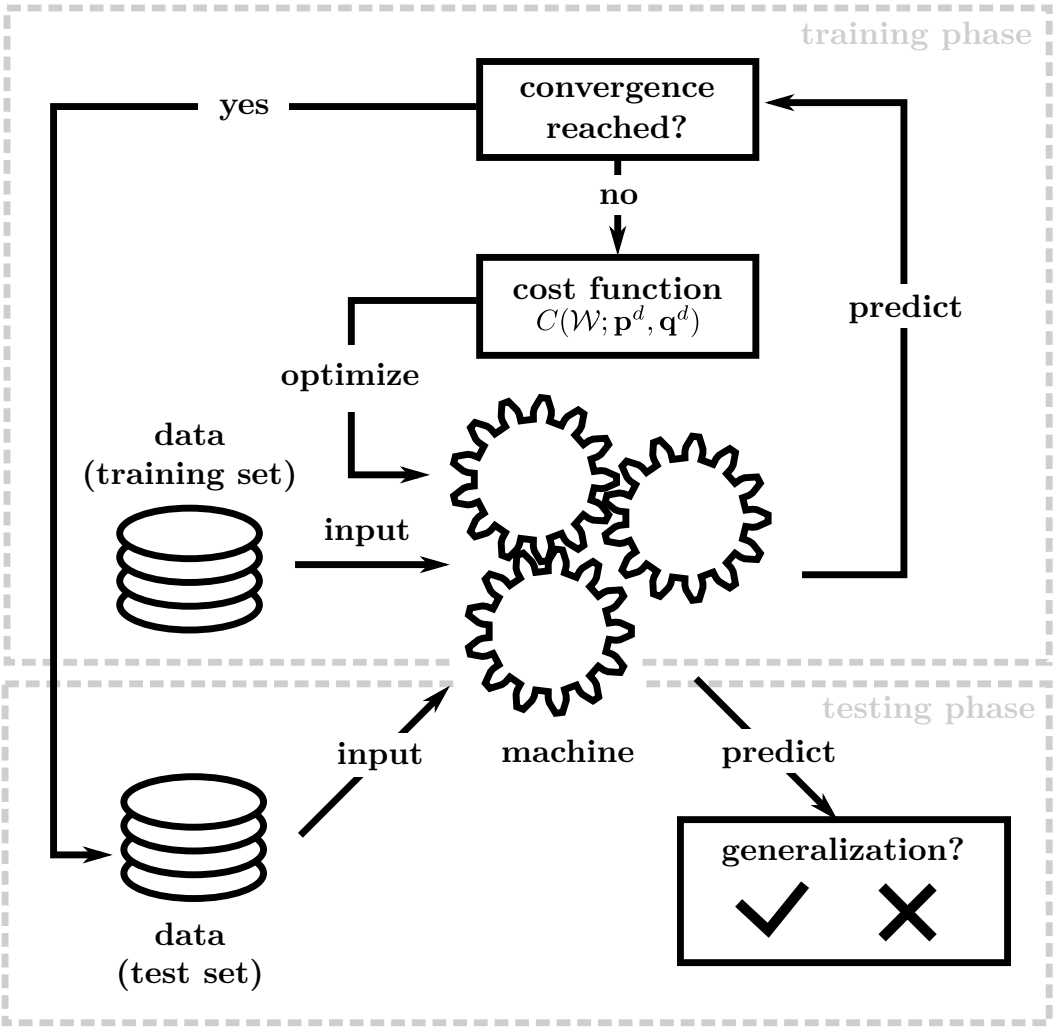

Fig. 2.1: Schematic representation of the machine learning strategy. Solving a problem with machine learning starts at the training dataset. Then, the machine (model) is optimized such that the cost function is minimized. Finally, the generalization is assessed in the testing phase.

Machine learning techniques can often be separated in a number of different steps. To illustrate this, the example of image recognition will be treated. The different steps are depicted schematically in figure 2.1.

1. **Definition.** The problem needs to be defined in a clear and (mathematically) approachable way. This step entails choosing and assimilating the dataset which will be used to perform the given task. For the image recognition example, the data is a set of images, all with the same number of pixels, and a set of classes every image can be assigned to. An example of such a dataset is the MNIST dataset [24] consisting of grayscale images of handwritten digits with 28×28 pixels. Every element of this dataset is labeled with the correct class it belongs to. An important step is to partition the dataset in a set which will be used as a training set and a test set. The training set is used in the optimization of the model (step 3) and the test set is used as a check for the predictive power of the model (step 4). The problem amounts to predicting the probabilities that a given image belongs to a specific class.

2. **Representation.** The defined task needs to be molded in a way suitable to be tackled by a machine learning technique. Aspects which need to be taken into account are how the data and the solved task are (mathematically) represented. An important question in this step is which model will be used to perform the machine learning task. Some examples are support vector machines (see section 2.3.2) and neural networks (see section 2.3.3). The representability of these models is contained in parameters which need to be optimized. For the image recognition example with grayscale images, every image can be represented as a two-dimensional array of grayscale values for every pixel. Every image is now considered a data point. The classes can be represented by a set of real values between 0 and 1, where every value denotes the probability of belonging to a certain class. For the MNIST dataset, this results in a vector $\mathbf{p}$ of 10 probabilities (one for every digit from 0 to 9). The model which performs the classification is chosen to be a convolutional neural network, which is highly suitable for image recognition tasks (see section 2.3.3). This network takes an image as input and has the vector $\mathbf{p}$ as output.

3. **Optimization.** The given dataset with examples and the model is used in an algorithm which optimizes a cost function (which is the performance measure P in the formal definition in the beginning of this chapter) representing how well the model performs on the given task. This function translates the problem into mathematics. The cost function can be defined in many ways, depending on the problem, on the model and on the data. The optimization often entails minimization for which e.g. gradient descent techniques can be used. Gradient descent is an iterative technique, in which, for a given set of current parameters,

the gradient of the cost function is calculated and the parameters are updated using the gradient. This leads the cost function to a (local) minimum (see section 4.2.2 for more details). Also other techniques exist such as greedy search algorithms, in which at every iteration step a number of parameter updates are proposed and the one which reduces the cost function the most is chosen. The optimization step is also called the training step. In the image recognition example, a common cost function C is the cross-entropy between the labels of the data $\mathbf{q}^d$ and the predictions of the model $\mathbf{p}^d$:

$$C(\mathcal{W}; \mathbf{p}^d, \mathbf{q}^d) = -\sum_{d=data} \sum_{c=class} q_c^d \log(p_c^d(\mathcal{W})), \tag{2.1}$$

where $\mathcal{W}$ is the set of optimizable parameters. The cost function is thus constructed by predicting the class of the data samples and measuring the deviation (as measured by the cross-entropy) from the provided labels. A perfect classification minimizes the cost function to zero. The cost function can be minimized using the gradient descent algorithm for which the gradients of the parameters are calculated via the backpropagation algorithm [25].

4. **Generalization.** This step entails examining how well the model is capable of performing the given task on data samples which were not included in the optimization procedure. This is the main objective of machine learning as stated by the formal definition. In the image recognition example, the generalization can be assessed by predicting the class of the examples in the test set as the class which has the highest probability. Given the labels of the data samples in the test set, we can calculate the error rate on the predicted classes. This error rate then defines a measure for the generalization.

2.2 Classification of machine learning approaches

Machine learning approaches can be classified in the following way.

- **Supervised learning.** This class spans the techniques for which the solution of the task is given in the training dataset. For example, in image recognition, for each image in the dataset, the true label is known. The algorithm can use this information (by introducing it in the cost function, see Eq. (2.1)) to solve the task at hand. The goal is to create a model which correctly predicts the label of a given example. Examples of this class of techniques are classification and regression.

- **Unsupervised learning.** In this class, the labels corresponding to the data points are unknown. The goal is to generate a labeling in order to make sense of the data. Clustering problems and outlier detection are prototypical examples of problems for which unsupervised learning can be used.

2.3 Some notable machine learning algorithms

2.3.1 PCA and t-SNE

PCA and t-SNE are dimensionality reduction techniques, i.e. their goal is to find a transformation of high-dimensional data points such that after the transformation some dimensions are redundant.

Pricipal component analysis (PCA) is a linear technique which performs a reorienting of the set of axes of the phase space of the data such that the new axes lie along the directions containing the most variance in the data [26]. The dimension of the new space can be reduced by discarding the dimensions along which the variance of the dataset is close to zero. In practice, one centers and normalizes the data (i.e. transform the dataset such that it has zero mean and unit variance in all directions) and calculates the covariance matrix of the dataset. The covariance matrix is diagonalized, resulting in eigenvectors containing the new coordinate system (the principal components) and eigenvalues containing the explained variance of each axis. By only considering the axes which have an explained variance above a certain threshold and leaving out the other axes, the dimension of the dataset is reduced to the number of principal components taken into account. PCA is schematically explained in figure 2.2

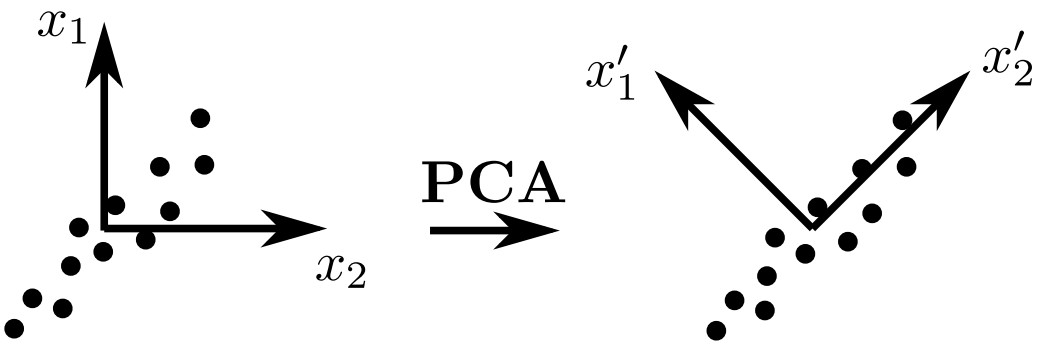

Fig. 2.2: Schematic depiction of PCA. The data (dots) has a high positive correlation. The original coordinate system (x_1 and x_2) is rotated by PCA to a new coordinate system (x'_1 and x'_2). The data now has a high variance along the axis x'_2 and a low variance along x'_1, making the axis x'_2 more important to describe the dataset than x'_1.

Non-linear techniques such as t-SNE aim at projecting a high-dimensional dataset to a 2D plane, such that the properties of the data distribution still hold. Performing these projections is quite technical, but they can be intuitively explained. For t-SNE, one tries to embed the data points in a 2D plane such that their relative distance is conserved [27]. That is, given two data points are close in the original coordinate system (according to some distance metric, e.g. Euler distance), the equivalent points in the 2D embedding should also be close.

2.3.2 Support vector machines

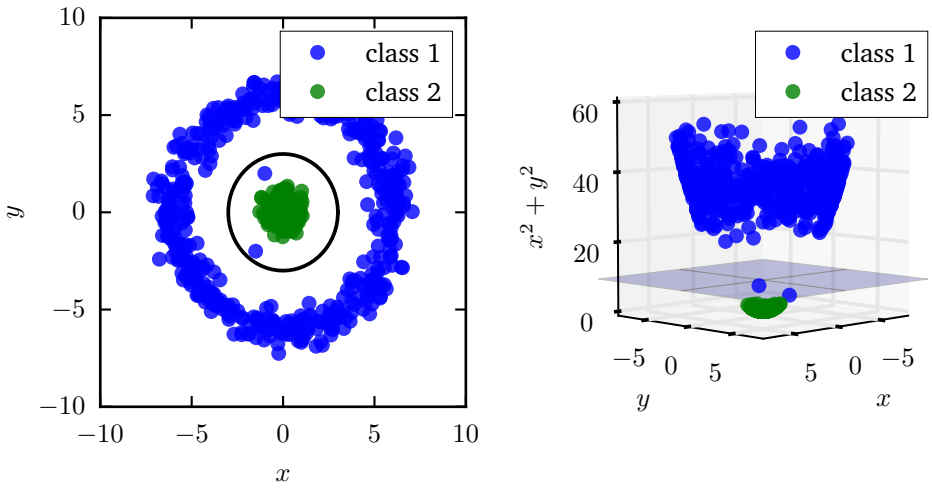

Fig. 2.3: Illustration of an SVM. Left: the data points are distributed such that the two classes cannot be separated by a linear hyperplane (i.e. a straight line in a two-dimensional plane). The black circle denotes the best boundary between the two classes (as measured by the maximization of class separation). Right: the data is embedded in a higher-dimensional space by a non-linear transformation. The data is separable by a linear hyperplane (i.e. a linear plane in three dimensions) in this space. Two noisy data points of class 1 are present, but don't influence the boundary.

Support vector machines (SVM) [28] are machine learning models which implement supervised learning. The aim is to perform a (binary) classification of a labeled dataset consisting of data points belonging to one of two classes. The approach of SVMs is to construct a hyperplane in the phase space of the data such that the data points are separated in the correct classes by this plane. The hyperplane is constructed in such a way that the distance from this plane to the closest data points is maximal. Mathematically, this problem boils down to a constrained optimization problem, where the distance from the hyperplane to the closest data points is maximized with the constraints that the data points are partitioned by the plane in their respective classes.

The SVM approach described above suffers from two problems. First, it is not suitable for noisy data because data points of a certain class might lie in regions

where the other class is abundant (due to noise). This makes the separation of the data points in the correct class by a hyperplane impossible. Second, the method fails when the data points are not separable by a hyperplane, for example when the data points lie on two concentric hyperspheres. However, these problems can be solved in an elegant way, making the SVM approach a widely applicable and interpretable method.

The problem of dealing with noisy data can be solved by calculating, for a given hyperplane, how far the incorrectly classified data points are from their correct class (i.e. the distance perpendicular to the hyperplane). The sum of the distances times a tunable factor l is then introduced as a cost in the optimization problem. Effectively, this cost will enforce that as few as possible data points fall in the wrong class. Large values of l enforce that the wrongly labeled data points have short distances to the hyperplane.

The problem related to data points that cannot be separated by a hyperplane can be solved by embedding the data points in a space with higher dimension. To do this, a mapping $\mathbf{z} = f(\mathbf{x})$ of the data points is performed, where $\mathbf{x}$ is a vector with dimension n and $\mathbf{z}$ is a vector with dimension m where $m > n$. The SVM approach is then performed in this new space (often called the feature space), where the data points are linearly separable. As the cost function to be optimized is a scalar, the vectors appear only in scalar products. In practice, the embedding in another space is not performed explicitly. Rather, the scalar product is redefined, implicitly embedding the vectors in another space. This approach is called the kernel trick [29]. It is not only used in the context of SVMs but also more generally to perform non-linear machine learning with a linear technique. An example is PCA (section 2.3.1), where it is used as part of the kernel PCA technique, which does PCA with non-linear transformations. An example of SVMs and the kernel trick is provided in figure 2.3.

2.3.3 Neural networks

Since neural networks will be used as a machine learning tool in this work, we will introduce them in more detail. Neural networks are models which are loosely based on the structure of collections of neurons in the brain. Neural networks typically represent a mathematical function, i.e. a mapping from an input $\mathbf{x}$ to an output $f(\mathbf{x})$. They are designed to handle high-dimensional input such as pictures, time series and DNA-sequences.

Neural networks can be conveniently introduced by examining the perceptron, depicted in figure 2.4. This neural network is the most simple one and consists of

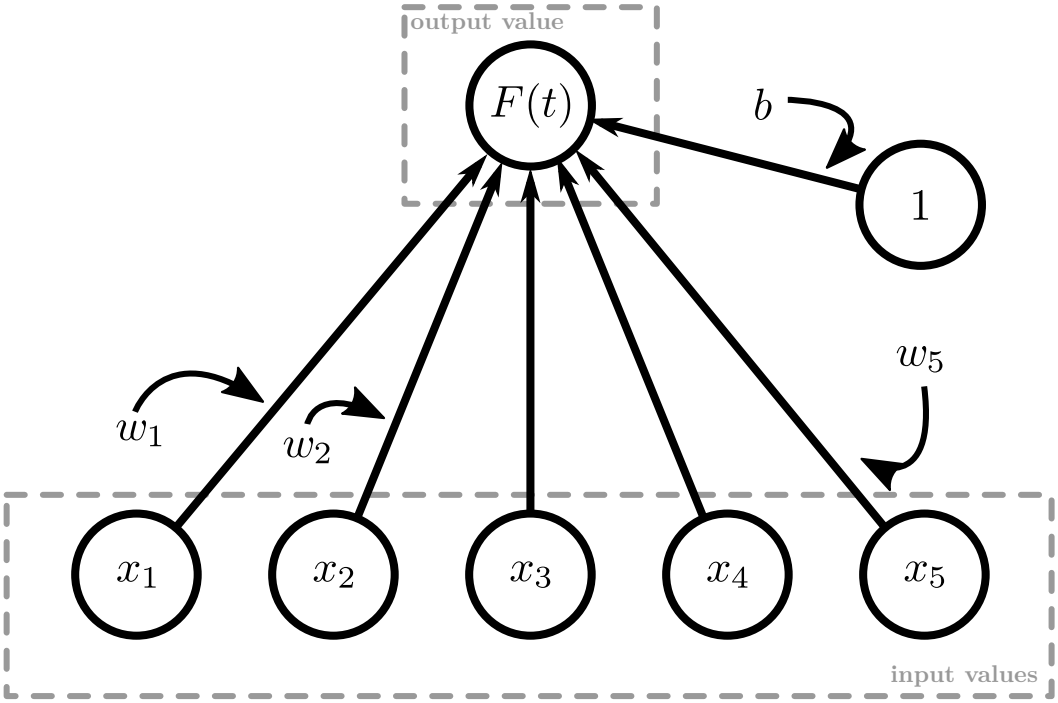

Fig. 2.4: Visualization of the perceptron. This perceptron takes a five-dimensional vector $\mathbf{x}$ as input and computes from this an output value $f(\mathbf{x}) = F(t)$. The bias b of the perceptron (see Eq. (2.2)) is introduced using an additional node with a fixed input of 1.

an array of input values $\mathbf{x}$ with dimension n, and a single output value $f(\mathbf{x})$. In the perceptron, all the input values x_i are multiplied with a weight w_i and added together to produce a scalar value, t. Typically, also a bias b is introduced, i.e. some constant is added to the sum. Mathematically, this comes down to

$$t = w_1 x_1 + w_2 x_2 + ... + w_n x_n + b = \sum_{i=1}^{n} w_i x_i + b. \tag{2.2}$$

This linear transformation rescales the data using the weights, and shifts the result of this operation using the bias. The weights $\{w_i\}_{i=1...n}$ act as a feature detector of the input. For example, some input x_i may not be important for the function approximation and consequently its weight w_i will be zero. Another possibility is that important inputs x_j will be multiplied with a high weight w_j (in absolute value) relative to the other weights. This operation transforms the input to a scalar output in a linear fashion. To approximate non-linear functions, it is important to incorporate some non-linearity in the model. This is done by feeding the scalar value t to a non-linear and non-polynomial function resulting in the output $h = F(t)$. This function can in principle take on different forms. Popular ones are the hyperbolic tangent $F(t) = \tanh(t)$, the logistic function $F(t) = (1 + e^{-t})^{-1}$, and the rectified linear function $F(t) = max(0, t)$. The choice depends on the problem and the training stability. One defines the neuron as the unit propagating t to $F(t)$. In this

network, the weights $\mathbf{w}$ are optimized to minimize the cost function, which implicitly depends on the weights (see section 2.1). An important point to take into account is that the cost function should not be optimized to its minimum, as this generally means that the model overfits the data it used for the training. This means that the model is optimized such that it starts to model the noise in the training data, resulting in poor results when used on data other than the training set (i.e. the generalization fails). This problem can be circumvented by taking another dataset into account, the validation set, which is used to assess the generalization during training. One stops the training process when the error (for example classification error) starts to grow on the validation set.

An example how this network can be used is phase classification for the Ising model of Eq. (1.1). The order parameter m (the physical observable distinguishing the two phases) is defined as the expectation value, with respect to the probability distribution of Eq. (1.2), of the sum of the spins divided by the total number of spins N

$$m = \frac{\langle \sum_{i=1}^{N} s_z^i \rangle}{N}. \tag{2.3}$$

It distinguishes between the ordered and disordered phase as the order parameter is zero for the disordered phase and non-zero for the ordered phase. If the weights $\mathbf{w}$ of the perceptron are equal to 1, the bias zero and the non-linear function $F(t) = (1+e^{-|t|})^{-1}$, the output of the perceptron is approximately 1 in the disordered phase and approximately 0 in the ordered phase [30].

The perceptron is typically not complex enough to capture all the features of the given problem. To overcome this, neural networks involve more neurons and more layers, essentially forming a network of perceptrons with interconnections. The simplest of such networks is the fully connected feedforward neural network. In this type of network, one or more layers of neurons sequentially operate on the input. The term fully connected indicates that connections between all neurons of subsequent layers are present. The term feedforward implies that the data flows through the network from input to output without recursive features in the computational flow. The fully connected feedforward neural network with one hidden layer is depicted in figure 2.5. The hidden layer consists of m perceptrons, all with their unique weight vectors and biases. The vector of perceptron outputs is a multi-dimensional representation of the input $\mathbf{x}$, denoted as

$$\mathbf{h^{(1)}} = (f(t_1^{(1)}) \quad f(t_2^{(1)}) \quad \dots \quad f(t_m^{(1)}))^T, \tag{2.4}$$

where $t_j^{(1)} = \sum_{i=1}^{n} w_{ji}^{(1)} x_i + b_j^{(1)}$ (the superscript denotes the layer). This vector is then used as the input for another perceptron, determining the output of the

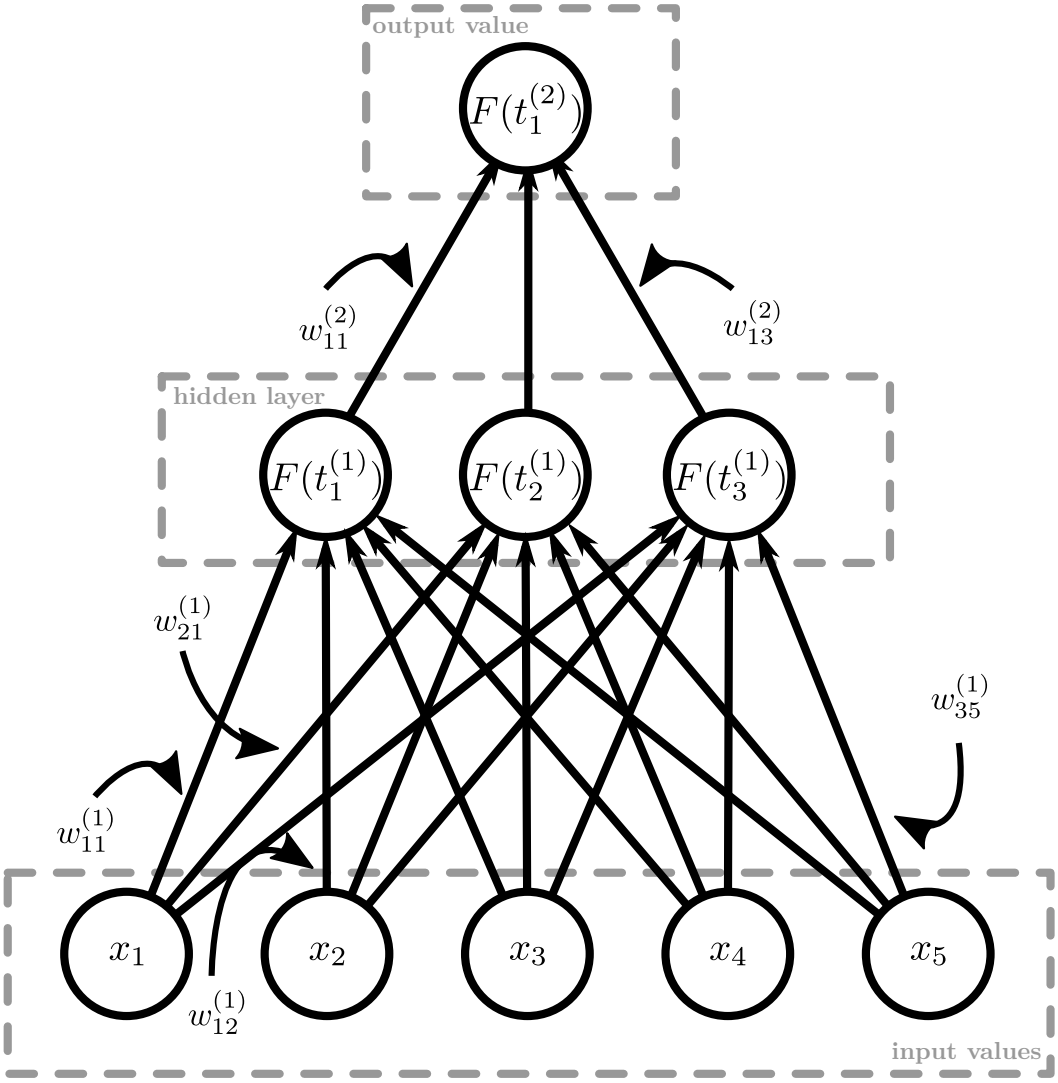

Fig. 2.5: Visualization of the fully connected feedforward neural network. In the illustration, the network takes a five-dimensional vector $\mathbf{x}$ as input, has 3 hidden neurons in the hidden layer and computes from this an output value $f(\mathbf{x}) = F(t_1^{(2)})$. The biases are not explicitly shown. Note how this network is built from perceptrons as shown in figure 2.4.

network. An important feature of the fully connected feedforward neural network is the Universal Approximation theorem [31], which is stated as follows

Universal Approximation Theorem

Given a continuous function $f(\mathbf{x})$, which is lower and upper bounded in its input. A fully connected feedforward neural network with non-polynomial activation function $F(t)$ and one hidden layer is able to represent the function $f(\mathbf{x})$ to arbitrary precision, given the number of hidden neurons is high enough. This also holds for networks with more than one hidden layer.

A wide range of architectures for networks are available, all excelling in different tasks. Some important networks, which will also be used in this thesis work are introduced below.

Convolutional Neural Networks (CNN)

Convolutional neural networks are a form of feedforward neural networks. They are

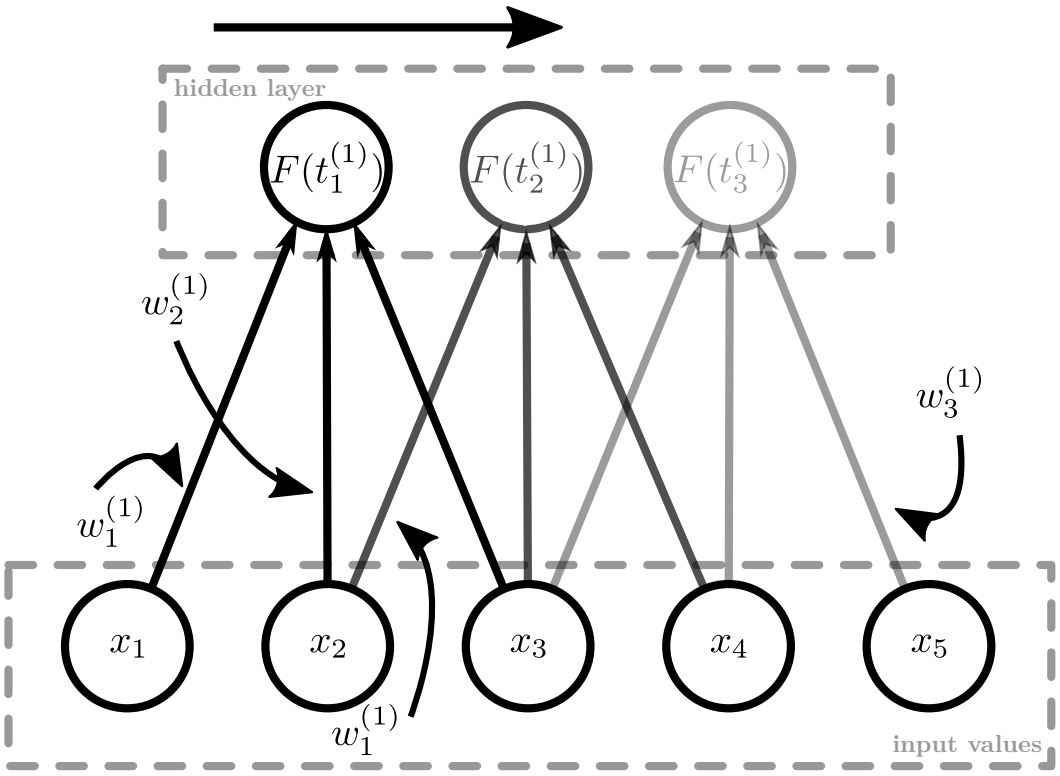

Fig. 2.6: Visualization of the convolution operation defined in Eq. (2.5). The input is a five-dimensional vector x. The convolution operator is subsequently applied 3 times indicated by the arrow and the shading. The biases are not explicitly shown. Note that the number of hidden neurons is the same as in figure 2.5, but the number of weights is five times lower, showing the efficiency of convolutional networks.

conventionally deep, i.e. contain more than one layer between input and output. Instead of all-to-all connections between two subsequent layers, the connectivity between layers is defined by a convolution operation. One or more convolutional filters are defined with a certain locality, i.e. far smaller than the dimension of the neurons it is applied upon. Given some range of the convolution operation r, the convolution operator on some (one-dimensional) input x is defined as

$$t_j^{(1)} = w_1^{(1)} x_{j-r} + w_2^{(1)} x_{j-r+1} + ... + w_{2r+1}^{(1)} x_{j+r} + b^{(1)}. \tag{2.5}$$

The input may be padded with zeros at its borders. The procedure is depicted in figure 2.6. The convolution operation can be generalised to more than one dimension. For example for images of millions of pixels, convolutional filters are defined with dimensions of 5 by 5 pixels. This convolutional filter is applied across

the output of the previous layer (input neurons or neurons from an intermediate layer). Often the filter is applied with a certain stride length, i.e. the filter is applied by moving it with the stride length between subsequent applications. The output of this operation is a set of neurons, the number of which is the number of times the filters are applied times the number of filters. The application of the convolutional filters acts as a feature detector in a local neighbourhood. By using multiple layers, a hierarchy of feature detectors is defined, starting with very local features such as edges and ending with global features such as faces or cars in image data. After these convolutional layers, a fully connected network is built upon the resulting neurons. This fully connected network then results in the output.

The convolutional neural network is mostly used in situations with high-dimensional data that exhibit local properties. Examples are image datasets and speech samples. The convolutional approach is computationally cheap compared to the fully connected networks for high-dimensional data.

Restricted Boltzmann Machines (RBM)

Restricted Boltzmann machines are bipartite networks consisting of input variables

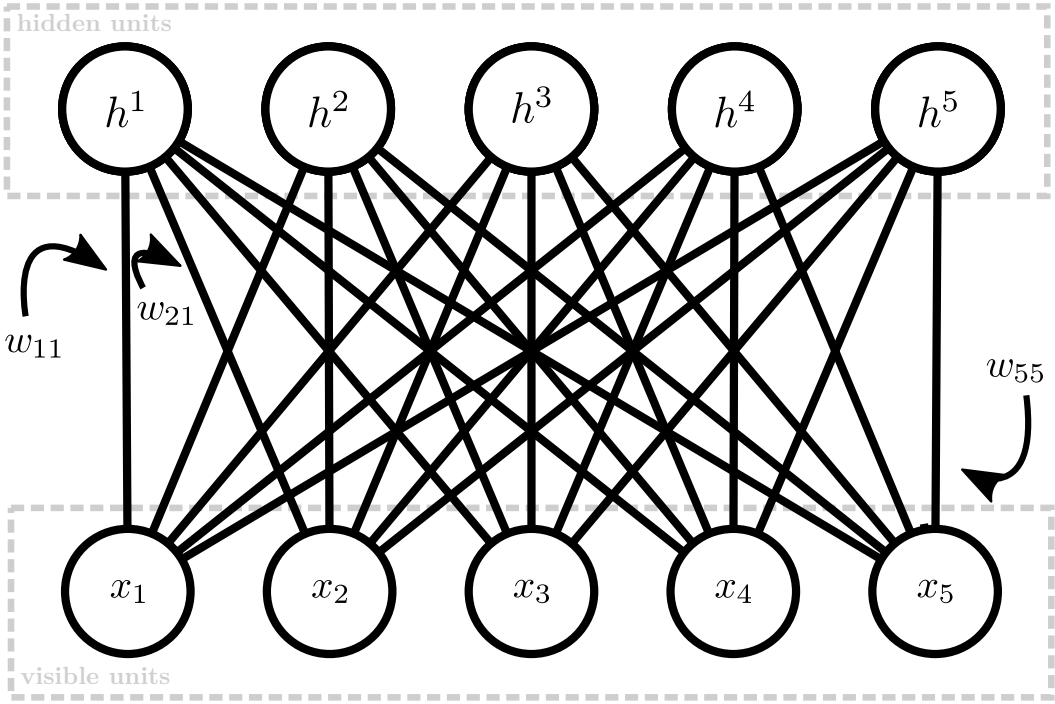

Fig. 2.7: Visualization of (the energy function of) the restricted Boltzmann machine defined in Eq. (2.7). The input is a five-dimensional vector $\mathbf{x}$. The correlations between the input variables are modelled by the five hidden units $\mathbf{h}$. The biases are not explicitly shown.

$\mathbf{x}$ (in the context of RBMs also called visible units or visible spins) and a layer of hidden variables $\mathbf{h}$ (or, hidden units or hidden spins), connected by a weight matrix $\mathbf{w}$. The structure of an RBM is shown in figure 2.7. Restricted Boltzmann machines

are a subclass of Boltzmann machines, meaning that in the restricted variant no connections are allowed between the visible units and between the hidden units. They are mostly used as a tool for approximating probability distributions. This is done by defining an energy function (as in a statistical physics context) for given weights and biases

$$E(\mathbf{x}, \mathbf{h}; \mathcal{W}) = \sum_{i=1}^{N_v} a_i x_i + \sum_{i=1}^{N_h} b_i h^i + \sum_{i=1}^{N_v} \sum_{j=1}^{N_h} w_{ij} x_i h^j. \tag{2.6}$$

In this equation, N_v and N_h are the number of visible and hidden units, respectively. This energy function is subsequently used to define a Boltzmann distribution for the different configurations of units

$$p(\mathbf{x}, \mathbf{h}; \mathcal{W}) = \frac{\exp\left(-E(\mathbf{x}, \mathbf{h}; \mathcal{W})\right)}{\sum_{\mathbf{x},\mathbf{h}} \exp\left(-E(\mathbf{x}, \mathbf{h}; \mathcal{W})\right)} = \frac{\exp\left(-E(\mathbf{x}, \mathbf{h}; \mathcal{W})\right)}{Z(\mathcal{W})}, \tag{2.7}$$

where the sum in the denominator runs over all possible configurations. The probability distribution for the given data $p(\mathbf{x}; \mathcal{W})$ can be found by summing over the hidden units

$$p(\mathbf{x}; \mathcal{W}) = \sum_{\{h^i\}} p(\mathbf{x}, \mathbf{h}; \mathcal{W}), \tag{2.8}$$

where $\sum_{\{h^i\}}$ denotes a sum over all configurations of the hidden units. The hidden units determine a latent space which encodes the abstract features in the data. From Eq. (2.7), one sees that the biases $\mathbf{a}$ and $\mathbf{b}$ determine how the units they are assigned to are favoured independently from the other units. For example, a large positive bias a_i will favour large negative values of x_i because the energy of Eq. (2.6) will be lower, which means that the exponential $\exp(-E)$ determining the probability in Eq. (2.7) will be larger. However, this favouring is (possibly) counteracted by the interaction with other units as determined by the term $\sum_{i=1}^{N_v} \sum_{j=1}^{N_h} w_{ij} x_i h^j$, which can increase (or further decrease) the energy of Eq. (2.6).

2.3.4 Generative adversarial networks

Generative adversarial networks (GAN), are models which aim at approximating the probability distribution of a dataset in its phase space. After the training phase, GANs allow to sample the probability distribution, generating unseen data samples. The model consists (typically) of two neural networks, the generator and the discriminator. The generator generates data samples from some low-dimensional latent space. Often, the generator is a deconvolutional neural network (i.e. the inverse of a convolutional network, section 2.3.3) which takes some input values generated randomly from e.g. a Gaussian distribution and constructs from these a high-dimensional data sample with the same shape as the given dataset. The

discriminator takes data samples and determines if the samples are real data (i.e. from the dataset) or pseudo-data (i.e. generated by the generator). In practice, it is e.g. a convolutional neural network which takes data samples as input and has two output nodes. GANs use the adversarial principle to construct a generator which describes the probability distribution of the dataset faithfully. The adversarial principle in the context of GANs is as follows:

1. The generator generates data samples, i.e. random Gaussian variables are fed to the network and data samples are constructed from them.

2. The discriminator tries to deduce whether the generated data is real data or pseudo-data from the combined dataset consisting of the generated data and the training dataset. The data is labeled as real data or pseudo-data. Using this information, the discriminator is trained to discriminate real data and pseudo-data better.

3. Given the output of the discriminator, the generator is trained to generate samples which cannot be discriminated by the discriminator.

4. Go back to 1 and repeat until the discriminator is unable to discriminate the data.

3 Combining many-body physics and machine learning

How can machine learning help physicists in tackling problems in many-body physics? By examining both subjects in the previous two chapters, we observe that many-body physics entails some characteristics which are also found in machine learning and data science. These are

- Both subjects treat the problem of *many degrees of freedom*. In many-body physics, this is the defining property. Outside of physics, the increase in data generated by companies, governments and other societal actors, not only entails an increase in the size of the datasets (big data), but also in the dimensionality of the datasets. This sparked the specialization of some parts of the machine learning community in problems with high-dimensional datasets. These include neural networks and the field of dimensionality reduction.

- An important aspect is unraveling the *correlations* between the degrees of freedom. For many-body physics, correlations are present when degrees of freedom are interconnected. These correlations give rise to emergent collective properties such as phases. Also in machine learning, correlations between the degrees of freedom are present and important. Often, these correlations are the reason for machine learning to exist, because they introduce complexity in the problem and render the problem unattainable to a simple approach. Consider for example pictures of cars. If its pixels would be uncorrelated, the depiction would certainly not be a car because every pixel would attain a value independently, rendering just noise.

These parallels make it clear that a cross-fertilization between physics and information technology is feasible and is likely to lead to breakthroughs in both fields. For the machine learning community, this has been clear for a long time and many concepts of (many-body) physics have entered the world of machine learning. For the physics community, the potential of machine learning to solve complicated problems has been recognized for a long time, but only since 2016 there has been a boost in activities.

3.1 Physics in machine learning

There have been strong connections between machine learning and physics for many decades. In 1982, John Hopfield proposed the Hopfield network, which he proved could learn patterns in data [32]. The Hopfield network is derived from a spin glass model in statistical physics. The network is defined by a spin glass energy function

$$E(\mathcal{S}) = \sum_{i,j} w_{ij} s^i s^j + \sum_i b_i s^i, \tag{3.1}$$

where $s^i = \pm 1$ denotes the binary value of the i-th degree of freedom (input), w_{ij} describes the interaction between degrees of freedom s^i and s^j and b_i regulates the offset of degree of freedom s^i. Further, $\mathcal{S}$ denotes a configuration of the spins $\{s^i\}$. In a machine learning context, the spins $\{s^i\}$ are the input values (e.g. binary black-white values in a picture). Eq. (3.1) thus allows to calculate the energy of a given data sample. During training, the network (i.e. the parameters w_{ij} and b_i) is optimized such that the data points represent (local) minima on the energy surface. This is done via the Hebbian rule [33]

$$w_{ij} = \frac{1}{n} \sum_{p=1}^{N} s^i(p) s^j(p), \tag{3.2}$$

where the index p runs over all data samples of the dataset and N is the number of data samples. From this rule, it can be seen that the weights w_{ij} by construction encode information of the data points. Given some new data sample, the input values s^i are updated according to the rule

$$s^i = \begin{cases} +1 & \text{if} \quad \sum_j w_{ij} s^i s^j > \theta_i \\ -1 & \text{otherwise,} \end{cases} \tag{3.3}$$

where θ_i is a chosen threshold. These updates are repeated until no changes occur and a new state $\{s^{i'}\}$ is obtained. Hopfield proved that $\{s^{i'}\}$ is a local minimum of the energy and that the final state is reached from the initial state by taking steps on the energy surface which only lower the energy. Because the network was optimized such that the data samples on which the network was trained correspond to the minima of the energy surface, the state of the network converges to one of the training states when some input state is presented. In this sense, the algorithm is capable of performing associative learning, meaning that the network converges to some state it has "remembered" from the training phase.

The Boltzmann machine (of which the restricted variant was introduced in section 2.3.3) is also a model originating from physics. It was invented in 1985 by Geoffrey Hinton and colleagues [34]. Like the Hopfield network, the model is also defined by

an energy function and the input is represented by spins. Furthermore, hidden spins are introduced in the description of the model. These encode the hidden features of the model and allow for connections to be made between input variables in the form of interaction weights. Rather than memorizing the data in the minima of the energy function (as is the case in the Hopfield network), the energy function is used to define a statistical ensemble, i.e. it defines a Boltzmann distribution for all the possible data configurations. This network encodes features of the data in its weights and hidden units, and the defined probability distribution can be used for generative purposes. Due to their grounding in physics, physical properties of the models can be exploited and investigated.

One physical property which has been (and is still being) investigated is the renormalization group procedure in the context of unsupervised learning. The renormalization group procedure is an iterative procedure reducing the degrees of freedom of a many-body physical system. The original degrees of freedom are iteratively mapped to fewer degrees of freedom such that the long-range (or macroscopic) properties of the system are minimally altered. For example, the Kadanoff block decimation technique [35] for spin systems entails summing small groups of spins together to produce a single effective degree of freedom, i.e. a new spin is formed from a few original spins. The collection of effective spins forms in itself a new spin system, for which the mapping can be repeated. This is repeated a number of times, until one ends up with a very small system consisting of spins which are the image of a very large block of original spins. As the macroscopic properties are by construction approximately conserved, the effective degrees of freedom still describe the macroscopic system. The difference is that the irrelevant properties of the system are factored out and the relevant properties remain present in the effective system. It is clear that there is an informal connection between deep learning and the renormalization group procedure. For example, looking at deep convolutional networks (see section 2.3.3), blocks of the original degrees of freedom are also iteratively mapped to effective degrees of freedom (the hidden nodes). This results in a much smaller set of degrees of freedom, while retaining the relevant global information of the input (for example, from the output of the convolutional layers, a classification can be performed).

The connection between the renormalization group procedure and (unsupervised) machine learning has been more thoroughly investigated in the context of deep Boltzmann machines [36]. The deep Boltzmann machine is a stack of RBMs, where the visible layer of a given RBM in the stack is the hidden layer of the RBM under it. The deep Boltzmann machine is trained layer by layer to optimally represent a certain probability distribution. Ref. [36] approximated the probability distribution of physical models (e.g. the Ising model) with deep Boltzmann machines, and found that the deep Boltzmann machine indeed learned a Kadanoff block decimation procedure, mapping at each layer small blocks of the input spins to the output spins.

This work thus proved the relation between the renormalization group procedure and deep learning explicitly, however only for a specific dataset and a specific model architecture. It remains an open question whether this result is more generally applicable to other datasets and other model architectures.

Not only the renormalization group but also other connections between machine learning and physics have been investigated. Machine learning problems share some very general properties with physical systems, such as locality, symmetry and hierarchical data generation citeLin17. In the context of image recognition, locality is found in the fact that two pixels are highly correlated only if they are close to each other. Symmetry is found in the translational and rotational invariance: the label of an image does not change by translating or rotating the image. The data in image recognition tasks is generated hierarchically, starting with the label of the image, generating from this some features such as gender, color or shape, mapping these features to a set of pixels, mapping this collection of pixels to a transformed image, e.g. rotating or translating and generating from this image the background on which the object which needs to be recognised is placed. This is reminiscent of physical data generation, for example the observed cosmic background is a hierarchy of transformations on the cosmological constants [37]. In Ref. [37] it was shown that data generated according to these properties gives rise to a problem which can be easily solved by deep neural networks.

Also physical concepts originating from quantum many-body physics found their way in machine learning. Just like models from statistical physics are used to perform machine learning tasks (the Hopfield network and the Boltzmann machine, see above), models from quantum many-body physics such as tensor networks (see section 4.3.2 for a very short introduction on tensor networks) have been used recently in a machine learning context [38]. Ref. [38] used tensor networks as a model for doing image classification. The inputs are the pixels of an image, on which a two-dimensional tensor network is applied, resulting in an operator which can be applied on a vector containing the different classes (i.e. every class corresponds with a vector element; for a given class, the corresponding entry is one and the other entries are zero). This yields a quantum state for every data sample. Given this quantum state, concepts from quantum many-body physics can be used to determine properties of the data. For example the overlap of two quantum states can be computed, resulting in a measure for the similarity of two classes (e.g. for a dataset consisting of images of handwritten digits, the digits 4 and 9 look similar, resulting in a high value of the overlap). Another measure is the entanglement entropy, which measures how much information can be gained about a certain part of the system when measuring the complement of this part. For example, when the lower part of a zero is measured, a lot of information is gained about the other half of the zero because a zero is symmetric.

In the context of tensor networks, entanglement entropy reveals how many degrees of freedom are needed to describe a physical state. This concept has been ported to the machine learning community where efforts have been done to use quantum entanglement in the design of neural networks [39]. Quantum entanglement describes the correlations and thus provides direct information regarding the representability of neural networks.

3.2 Machine learning in physics

Section 3.1 explained how physics has influenced the machine learning community already a long time ago. However, machine learning has only recently been proposed as a new technique in theoretical physics (note that machine learning has found many applications in experimental physics [40, 41]). Machine learning applications in theoretical physics can be roughly divided in three different categories.

1. Machine learning for model selection. This category resides on the edge between theoretical and experimental physics.

2. Machine learning to improve the performance of algorithms in computational physics.

3. Machine learning to detect the high-level features of systems with many interacting degrees of freedom.

This categorization is quite rough and many of the available literature can be classified in more than one category. In the following subsections, each of these categories will be clarified with some examples of recent literature.

3.2.1 Machine learning for model selection

This research branch entails the use of machine learning to select suitable models for physical phenomena. It is often the case that physical models contain (effective) parameters of nature, which cannot be determined starting from first principles. These parameters need to be fitted to the experimental data, which can be highly non-trivial due to measurement limitations or large amounts of noise in the data. To alleviate this, machine learning can be used as a complex fitting technique. Machine learning is capable of finding relevant features of data and make abstraction of possible noise or uncertainty in the dataset. The applications in this category use measured data points as input and have information about the model as output (e.g. model parameters or model classes). The property that machine learning

techniques can make a highly non-trivial connection between input values and output is used. However non-trivial, it is important that the machine learning technique can generalize to data it has not seen. In this way, experimental data points can be provided to the machine learning model which outputs the physical model parameters according to these data points.

This strategy has been used in the context of neutron stars, where machine learning has been used to find the equation of state for hadronic matter from a limited dataset with a large amount of noise (astronomical observations of the mass and radius of neutron stars) [42]. A neural network is used to connect a number of n mass-radius pairs (i.e. $2n$ input values) to the physical model parameters characterizing the equation of state. The network is trained and tested on a set of mass-radius collections generated from different equations of state. The network is able to generalize to unseen sets of mass-radius collections and can handle large amounts of noise.

Another application in nuclear physics is the search for "the missing theory" to determine electric nuclear charge radii [43]. Thereby, one compares measured charge radii of atomic nuclei with computed ones using state of the art theoretical models. The obtained deviations are then learned by a Bayesian neural network with the number of protons and the total number of nucleons as input. It is found that the Bayesian neural network can find the deviations of the charge radii of unseen data points, effectively learning the missing physics in the existing models.

A last example is the use of convolutional neural networks to derive the equation of state for relativistic heavy ion reactions [44]. Here, the measured values in experiments (the distribution of particles after the collision as a function of the transverse momentum p_T and azimuthal angle ϕ at different values of longitudinal momentum) are provided as input to the convolutional neural network, which outputs the type of equation of state, either an equation of state with a crossover or an equation of state with a first-order phase transition. The distribution of particles after the collision is the product of a multitude of non-trivial interactions. The convolutional neural network is able to deduce these interactions and connect the equation of state of the collision to the final distribution of particles.

3.2.2 Machine learning for improvement of simulations in physics

It is often found in computational physics that algorithms fail or become untrustworthy for some combinations of physical parameters. The reason is often a combination of the construction of the algorithms and some physical effect which the algorithms

cannot handle. Famous examples are the critical slowing down at, or near, phase transitions in statistical physics or the sign problem in fermionic quantum mechanics. The first one entails that the autocorrelation time τ_t for the generation of configurations of physical models becomes very large, essentially spoiling the statistics of the numerical experiments. The autocorrelation time is defined as the typical time scale of the correlations in a statistical variable $X(t)$ between different times:

$$\langle (X(t) - \langle X(t) \rangle)(X(t+s) - \langle X(t+s) \rangle) \rangle \propto \exp(-\tau_t s). \tag{3.4}$$

The sign problem arises due to the Fermi-principle which states that the wave function of fermionic particles changes sign whenever the quantum numbers of two of the particles are interchanged. This creates a highly oscillating function which is generally hard to describe. Moreover, the evaluation of integrals becomes hard due to the cancellation of positive and negative contributions, which presents a problem when evaluated on a computer due to numerical rounding errors. Machine learning can be used to circumvent or solve these problems with physical simulations. However, there is no general machine learning principle which applies to all the techniques in this category. The machine learning principles used to solve the above described problems will be presented in this section.

The impact of critical slowing down can be circumvented with machine learning in two ways. First, the conventional methods to generate configurations of systems in statistical physics can be adapted. An example is Markov Chain Monte Carlo (MCMC, see section 4.2.2) which generates a sequence of configurations of the degrees of freedom by sequentially applying updates to a starting configuration. The updates are accepted or rejected according to some criterion. Most conventional methods generate a sequence of configurations of the degrees of freedom, by performing *local* updates starting from some initial configuration. Local updates change one or a few degrees of freedom between two consecutive configurations. When the system is in such a state that changing degrees of freedom in a local region is highly disfavoured, the algorithm has difficulties to propose a suitable update. This is for example the case at criticality where the correlations become long-range. Global update schemes, which change multiple degrees of freedom between subsequent configurations, exist but cannot be implemented for every system (see for example [45]). The machine learning approach proposed in [46] provides a way to perform global updates for arbitrary systems. This is done by generating training configurations via MCMC with local updates. Machine learning is used to fit an effective Ising-like Hamiltonian to the training samples. After these steps, the actual configuration generation is done by performing MCMC with a global algorithm from this effective Hamiltonian and rejecting or accepting the updates using the real Hamiltonian.

A different approach is to use models from machine learning (e.g. neural networks) to encode the physical models. For this, generative models are used (i.e. models with which one can generate data points according to a distribution determined by training the data distribution). The first example is the use of (deep) restricted Boltzmann machines (RBM, see section 2.3.3) [47]. By mapping the physical models to these RBMs, one can use all the available methods applicable to RBMs to generate configurations (e.g. Gibbs sampling) and to analyze the physics of the model. The second technique is the use of generative adversarial networks (GANs, see section 2.3.3) to implement physical models [48]. Once trained, GANs can generate data samples from random variables in some low-dimensional latent space. Upon accomplishing this for physical systems, the autocorrelation time essentially becomes zero and subsequent samples are totally uncorrelated from each other. This makes the generation of data much easier than conventional (MCMC) methods.

The fermionic sign problem for distinguishing phases can also be circumvented by machine learning algorithms [49]. The fact that machine learning models can represent functions of complex data representations is used to define an observable $\hat{F}$ which distinguishes between two phases. One has $\langle \hat{F} \rangle = 0$ for wave functions in one phase and $\langle \hat{F} \rangle = 1$ for wave functions in the other phase. The Hamiltonian used in [49] is

$$\hat{H} = -t \sum_{\langle i,j \rangle, \sigma} c^{\dagger}_{i,\sigma} c_{j,\sigma} + U \sum_{i} n_{i,\uparrow} n_{i,\downarrow}, \tag{3.5}$$

where t and U are interaction strengths, $c_{i,\sigma}(c^{\dagger}_{i,\sigma})$ is the fermion annihilator (creator) of a spin with spin-value σ at site i and $n_{i,\uparrow}(n_{i,\downarrow})$ is the number of up (down)-spins on site i. Instead of evaluating the observable $\hat{F}$ in the ensemble defined by the wave function (which fluctuates due to the fermionic character), the absolute value of the wave function is used as the ensemble. The observable $\hat{F}$ is represented by a convolutional neural network which takes Green's function data as input and outputs the expectation value $\langle \hat{F} \rangle$. The Green's function is defined as

$$G(i,j) = \langle c_{i,\sigma} c^{\dagger}_{j,\sigma} \rangle. \tag{3.6}$$

It is found that this approach is able to perform phase classification without using the sign problematic part of the wave function.

3.2.3 Machine learning to detect high-level features in complex systems

The problem of finding high-level features from low-level degrees of freedom in complex systems is possibly the area where machine learning had the biggest impact recently. Active research, including the investigation of phase transitions in many-

body systems and finding the ground states (and features of it) of quantum many-body systems is ongoing. Solving problems in many-body physics requires carefully chosen approximations. Investigating phase transitions is hard due to the fact that many-body systems at a phase transition exhibit a large amount of correlations, requiring a description which does not neglect the interactions of the degrees of freedom. Moreover the theory of phase transitions requires a substantial amount of engineering. For example, there is no known recipe to find the order parameter (an observable of the system which is zero in one phase and non-zero in the other) of second order phase transitions. The same problems arise in quantum many-body physics. In sections 1.2 and 1.4, we already made clear the connection between classical many-body physics and quantum many-body physics. The ability of machine learning models to describe and make sense of correlations present in data samples is the main driving force for the research in this area. The data are configurations of the system, generated with e.g. a Markov Chain Monte Carlo algorithm. For phase transitions, the discriminative ability of supervised machine learning algorithms can be used for phase classification. Also unsupervised techniques can be used to investigate structural differences in configurations of many-body systems. The representational ability of generative machine learning models is used in quantum many-body physics to describe wave functions.

For the investigation of phase transitions in many-body systems, one attempts to map the phase diagram and find features such as order parameters. To map the phase diagram, different techniques are available. The first one is principal component analysis (PCA, see section 2.3.1), a technique which is conventionally used for dimensionality reduction. This technique entails a linear transformation of the phase space of the data to a set of basis vectors which are aligned according to the largest variance of the dataset. This technique has been used for physical systems, where the data are the configurations at different temperatures [50]. It was found for the Ising model that the first principal component coincided with the order parameter of the model, thus providing insight into the relevant physics. Of course, the technique performs a linear transformation, meaning that the transformation can only describe order parameters which are linear in the degrees of freedom. Non-linear transformation techniques exist (such as kernel PCA or t-SNE, see section 2.3.1), but these are more difficult to interpret.

Also supervised machine learning can be used to investigate phase transitions. Here, one knows where the transition point occurs and data samples are being labeled according to this transition point. A machine learning algorithm is used to perform a classification. Typically, neural networks have been used to perform this task [30, 51]. The classification of phases relies on the fact that the value of the control variable (e.g. temperature) at the transition is known. This is not always the case. Machine learning, however, can also be used for the task of finding the transition

point. One way of doing this is the confusion learning technique of [52]. This technique can find the value of the temperature (or some other control variable) at which phase transitions occur. It can be called semi-supervised as it uses machine learning techniques which are supervised for an unsupervised task. The task is unsupervised because there is no known labeling available for the different data points. The algorithm entails defining a range of temperatures $\{T_i\}$ to test for the occurrence of a phase transition and for which data is generated. The data is generated according to the partition function of Eq. (1.2) for classical systems. For every temperature T_i in the range, one labels the data points such that the data points are labeled as phase 1 when $T < T_i$ and as phase 2 when $T > T_i$. One uses a classifier to try to classify the (deliberately mislabeled) data samples. The point T_b at which the classifier performs best is found as the transition point. This technique relies on the fact that away from the phase transition, the dataset is not well labeled and the classifier cannot find features to classify on. Some data samples which are in phase 1 have the label of phase 2, while they are substantially different from the samples which are in phase 2. The phase classifier has difficulty to find ways to perform a classification for this mislabeled data. Only at the phase transition point, the data is partitioned in meaningful classes with different characteristics. This technique is used for quantum mechanical phase transitions in the original paper [52] and for the XY-model in [53].

While neural networks are very powerful, they are also very complex and thus difficult to interpret. While this is a problem at the moment, progress is being made towards interpretable neural network learning via various algorithms such as occlusion, deepLIFT [54] and guided backpropagation [55]. Also other machine learning algorithms have been proposed to perform phase classification. A notable one is the use of (kernel) support vector machines (SVM, see section 2.3.2) [56]. SVM is a supervised technique which tries to find a linear hyperplane in the phase space of the data such that this linear hyperplane is as far as possible from an element of both classes. The kernel in kernel SVM can yield information about the properties of the data. For example, in [56] is pointed out that for the 2D Ising model the kernel embeds the data points in a space such that the classes can be discriminated by the square magnetization, i.e. the order parameter.

Machine learning approaches have also been used in quantum many-body physics. This endeavour started with the work of [57], where restricted Boltzmann machines are used to approximate the ground state of quantum mechanical spin systems. As explained in section 1.2, finding wave functions of many-body Hamiltonians is a challenging problem as it involves the diagonalization of a matrix with high dimension. Finding the ground state of many-body Hamiltonians thus requires approximations, often in the form of variational wave functions. These are wave functions which require some optimization of free parameters such that the optimized

wave function approximates the ground state as well as possible. Machine learning techniques have proven that they can represent very complex functions with a small amount of computational resources. Furthermore, the universal approximation theorem (see section 2.3.3) provides a rigorous proof that a neural network can represent any continuous function. Using specifically the RBM as an approximation for the wave function is motivated by its stochastic interpretation as a classical spin system, which represents correlations via its hidden layers. The connection between classical spin systems and quantum spin systems was treated in chapter 1. In [57], the RBM (with complex valued weights) (see section 2.3.3) is used as a function which determines the expansion coefficients of the ground-state wave function in some selected basis. The hidden units in the RBM act as mediators of (quantum) correlations between different spins in the system. The proposed technique uses the (non-linear) variational principle to optimize the parameters in the RBM such that the ground state of the system is approximated as well as possible. Already in this first paper, the technique was found to perform better than other variational ansätze on the studied systems. After this first work, which focussed on prototypical quantum spin systems, other publications followed. The RBM technique has been used on fermionic many-body systems on a lattice [58], on bosonic many-body systems on a lattice [59, 60] and on chiral topological spin systems [61].

4 Modelling ground states of quantum systems with restricted Boltzmann machines

In this chapter, we present the restricted Boltzmann machine (RBM) technique to represent the ground state of quantum many-body systems in detail [57]. The variational ansatz for the wave function will be introduced. The methodology of the computer program used to find ground states of quantum-many body systems will be explained and the robustness of the adopted methods will be investigated. Some features of the technique will be presented in detail. We will focus on the problem of finding many-body ground states of quantum spin systems on a lattice in one and two dimensions. This problem was introduced in section 1.2.

4.1 Variational ansatz

4.1.1 RBM representation of wave functions

A quantum mechanical wave function for a spin system with N_v spins can be written as an expansion in a given basis, e.g. the σ_z-basis of Eq. (1.11). For $s = 1/2$ degrees of freedom one has

$$|\Psi\rangle \equiv \sum_{\mathcal{S}} \Psi(\mathcal{S};\mathcal{W})\,|\mathcal{S}\rangle = \sum_{\{s_z^i\}} \Psi(s_z^1, s_z^2, ..., s_z^{N_v};\mathcal{W})\,|s_z^1 s_z^2 ... s_z^{N_v}\rangle\,, \tag{4.1}$$

where $\mathcal{S}$ is a spin configuration in the σ_z-basis, $\mathcal{W}$ is a set of parameters (weights) the description may depend upon and s_z^i denotes the eigenvalue belonging to the eigenfunction of a single spin at site i in the σ_z-basis of Eqs. (1.7) and (1.8). The sum $\sum_{\{s_z^i\}}$ extends over all the different combinations of single-spin eigenfunctions (configurations). The wave function is determined if one knows the value of $\Psi(\mathcal{S};\mathcal{W})$ for every $\mathcal{S}$. In practice, this is intractable for large systems. Eq. (1.4) made it clear that the size of this set of expansion coefficients scales exponentially with the system's size. To overcome this, one introduces a variational ansatz for the expansion coefficients $\Psi(\mathcal{S};\mathcal{W})$. The weight parameters $\mathcal{W}$ need to be optimized such that Eq. (4.1) approximates the true wave function (e.g. the ground state) as well as possible.

We adopt an RBM, as introduced in section 2.3.3, as the variational ansatz for the expansion coefficients:

$$\Psi(\mathcal{S};\mathcal{W}) = \sum_{\{h^j\}} \exp\left(\sum_{i=1}^{N_v} a_i s_z^i + \sum_{i=1}^{N_h} b_i h^i + \sum_{i,j=1}^{N_v,N_h} w_{ij} s_z^i h^j\right). \tag{4.2}$$

Here $\mathcal{W} = \{a_i, b_i, w_{ij}\}$ are the variational parameters, s_z^i is the eigenvalue ($s_z^i = -1$ or $s_z^i = +1$) of the σ_z-eigenfunction of spin i, and h^j are the hidden spins which constitute the restricted Boltzmann approach of modelling correlations between (visible) spins. Furthermore, N_v (N_h) is the number of visible (hidden) spins. The visible spins correspond with the physical (effective) degrees of freedom. A graphical representation of an RBM with 5 visible and 5 hidden spins is shown in figure 4.1. Compared to Eqs. (2.6) and (2.7), the notation in Eq. (4.2) is adapted to fit

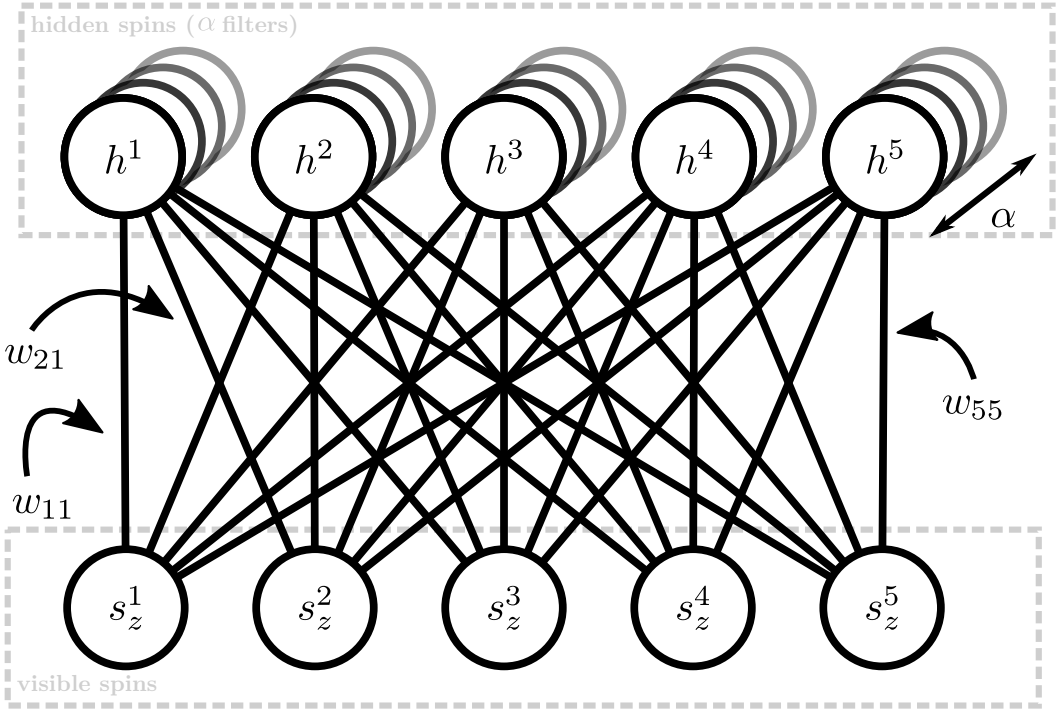

Fig. 4.1: Visual depiction of the RBM used to model the expansion coefficients of Eq. (4.1). The correlations between the visible spins s_z^i are represented by the weights w_{ij} and the interactions with the hidden spins h^j. The layer of hidden spins can be partitioned in $\alpha = N_h/N_v$ filters, which is especially natural for systems with transaltional invariance (see section 4.1.2). The biases are not explicitly shown.

our problem. The input values x_i in Eq. (2.6) are replaced with s_z^i in Eq. (4.2). Furthermore, the minus sign in Eq. (2.7) is absorbed in the weights in Eq. (4.2) and the normalization by the partition function Z in Eq. (2.7) (which is in general intractable) is absorbed in the normalization of the wave function. In general, the weights are complex numbers (since the wave function is in general complex). For some Hamiltonians, however (i.e. those that are real), the ground state wave function is real.

The sum over the hidden spin configurations in Eq. (4.2) can be calculated explicitly. This yields the following form for $\Psi(\mathcal{S};\mathcal{W})$

$$\Psi(\mathcal{S};\mathcal{W}) = \exp\left(\sum_{i=1}^{N_v} a_i s_z^i\right) \prod_{j=1}^{N_h} 2\cosh\left(b_j + \sum_{i=1}^{N_v} w_{ij} s_z^i\right). \tag{4.3}$$

The representation can be made more physical by imposing symmetry in the ansatz of Eq. (4.3). The wave function must respect certain symmetries of the Hamiltonian. Another motivation for implementing symmetries is that the number of parameters reduces substantially.

4.1.2 Implementing symmetries

The ansatz in Eq. (4.2) allows for both open and periodic boundaries. Imposing periodic boundary conditions generates a translationally invariant wave function. Another symmetry which is often found in spin systems is the invariance under flipping all the spins along a given axis. If this axis is the z-axis, the RBM representation of the wave function can be reduced further. The influence of these symmetries is shown in the following.

Translational symmetry

Suppose we have the translation operator $\hat{T}_j$ which translates the spins in a one-dimensional spin chain by j places to the right. Using the expansion of Eq. (4.1), operating with $\hat{T}_j$ on the wave function yields

$$\hat{T}_j |\Psi\rangle = \hat{T}_j \sum_{\mathcal{S}} \Psi(\mathcal{S};\mathcal{W}) |s_z^1 s_z^2 ... s_z^{N_v}\rangle = \sum_{\mathcal{S}} \Psi(\mathcal{S};\mathcal{W}) |s_z^{1-j} s_z^{2-j} ... s_z^{N_v-j}\rangle. \tag{4.4}$$

Here, periodicity in the indices is assumed, i.e. $s_z^{N_v+1} = s_z^1$. A similar operator can be defined for a system in arbitrary dimensions. If translation is a symmetry of the Hamiltonian $\hat{H}$, i.e. if $[\hat{T}_j, \hat{H}] = 0$, then the ground state should be invariant under translations. This implies that $\Psi(\mathcal{S};\mathcal{W})$ should be identical for translated copies of spin configurations. We first assume $N_v = N_h$. By looking at the definition of $\Psi(\mathcal{S};\mathcal{W})$ in Eq. (4.3) , we see that translation invariance can be enforced by setting the parameters $a_i \equiv a$, $b_j \equiv b$, and w_{ij} should have a form in which consecutive columns have periodically translated entries, i.e. a matrix of the form

$$w_{ij} = \begin{pmatrix} w_1 & w_3 & w_2 \\ w_2 & w_1 & w_3 \\ w_3 & w_2 & w_1 \end{pmatrix}. \tag{4.5}$$

With these restrictions, the form of $\Psi(\mathcal{S};\mathcal{W})$ becomes translationally invariant

$$\begin{aligned}\Psi(\mathcal{S};\mathcal{W}) &= \exp\left(\sum_{i=1}^{N_v} a_i s_z^i\right)\prod_{j=1}^{N_h} 2\cosh\left(b_j + \sum_{i=1}^{N_v} w_{ij} s_z^i\right) \\ &\overset{\text{TI-symm.}}{\rightarrow} \exp\left(a\sum_{i=1}^{N_v} s_z^i\right)\prod_{j=1}^{N_h} 2\cosh\left(b + \sum_{i=1}^{N_v} w_{1,(i+j-1)} s_z^i\right),\end{aligned} \tag{4.6}$$

where in the last line, we wrote the elements of the weight matrix w_{ij} of Eq. (4.5) explicitly in terms of the elements of the first column of the weight matrix

$$w_{ij} \rightarrow w_{1,(i+j-1)}. \tag{4.7}$$

Periodicity of the weights is implied by $w_{1,(N_v+1)} = w_{1,1}$. The parameter set is reduced from vectors with elements a_i and b_i to scalars a and b, and a matrix with elements w_{ij} to a vector with elements w_i. The translation invariance can be seen in Eq. (4.6). The exponential part is invariant under translation because the sum of the spins stays the same under translations. The product over the hidden nodes is also invariant under translations, as translating the spins comes down to reshuffling the order of the factors in the product.

The above considerations apply to the case $N_v = N_h$. One can introduce more hidden spins by an analogous construction. For example when $N_h = 2N_v$, the weight matrix of Eq. (4.5) now becomes rectangular and the periodic translation of consecutive columns is now implemented in square blocks

$$w_{ij} = \left(\begin{array}{ccc|ccc} w_1^1 & w_3^1 & w_2^1 & w_1^2 & w_3^2 & w_2^2 \\ w_2^1 & w_1^1 & w_3^1 & w_2^2 & w_1^2 & w_3^2 \\ w_3^1 & w_2^1 & w_1^1 & w_3^2 & w_2^2 & w_1^2 \end{array}\right). \tag{4.8}$$

In Eq. (4.8) the weight matrix consists of two square blocks which both consist of periodically translated columns in the same way as in the weight matrix of Eq. (4.5). The superscripts denote the index of the square blocks. There can now also be 2 hidden biases. The hidden layer is thus in general split up in N_h/N_v filters which all have the same structure as in the case of $N_h = N_v$ but have of course different values of the weights. The number of filters is denoted as

$$\alpha \equiv \frac{N_v}{N_h}. \tag{4.9}$$

We use the term filter because the operation on the spins in the arguments of the cosine hyperbolics of Eq. (4.6) resembles Eq. (2.5) defined in the context of convolutional neural networks. The weight vector in Eq. (4.6) is applied on the input spin configuration and the cosine hyperbolic is taken of the resulting value. After that, the weight vector is moved one place to the left (or right) and the same

operation is done. This is repeated until we arive in the starting situation and all the cosine hyperbolics are multiplied. A minor downside of implementing translational invariance in this way is that the possible amount of hidden spins is a multiple of the amount of physical spins, whereas it is any natural number in the general case.

The translation invariant $\Psi(\mathcal{S};\mathcal{W})$ for a general weight matrix can now be written as follows

$$\Psi(\mathcal{S},\mathcal{W}) = \exp\left(a\sum_{i=1}^{N_v} s_z^i\right)\prod_{j=1}^{N_v}\prod_{f=1}^{\alpha} 2\cosh\left(b_f + \sum_{i=1}^{N_v} w_{i,(j+fN_v)}s_z^i\right). \tag{4.10}$$

Spin flip symmetry

Another symmetry which can be implemented is spin flip invariance. The spin flip operator for flips of all spins along the z-axis is defined as

$$\hat{F}_z = \sigma_x^1 \otimes \sigma_x^2 \otimes ... \otimes \sigma_x^{N_v}, \tag{4.11}$$

as can be seen from Eqs. (1.7) and (1.8)

$$\sigma_x\left|s_z = +\frac{1}{2}\right\rangle = \left|s_z = -\frac{1}{2}\right\rangle \quad \text{and} \quad \sigma_x\left|s_z = -\frac{1}{2}\right\rangle = \left|s_z = +\frac{1}{2}\right\rangle. \tag{4.12}$$

For $[\hat{H},\hat{F}_z] = 0$, the wave function becomes invariant under spin flips , i.e. $\Psi(\mathcal{S};\mathcal{W})$ should be the same when all spins are flipped in the state. From Eq. (4.10), it is clear that both biases can be set to zero because they both introduce an asymmetry with respect to spin flips. This yields the form

$$\Psi(\mathcal{S};\mathcal{W}) = \prod_{j=1}^{N_v}\prod_{f=1}^{\alpha} 2\cosh\left(\sum_{i=1}^{N_v} w_{i,(j+fN_v)}s_z^i\right). \tag{4.13}$$

The above symmetries are straightforward to implement due to the structure of the RBM. Other symmetries may exist such as spin rotations

$$\hat{R}(\boldsymbol{\theta}) = \exp\left(-i\boldsymbol{\theta}\cdot\boldsymbol{\sigma}\right), \tag{4.14}$$

where $\boldsymbol{\sigma}$ is the vector of Pauli matrices defined in Eq. (1.8) and $\boldsymbol{\theta} = \theta\boldsymbol{e_\theta}$ is the rotation vector defining a rotation around $\boldsymbol{e_\theta}$ by the angle θ. These symmetries can not readily be implemented in the RBM.

Note that the implementation of symmetries reduces the parameters of the model and enhances the computation time of the calculations. On the contrary, they can also render the model not rich enough to learn efficiently. Not implementing these

symmetries can lead to more freedom in the ansatz which can help to guide the wave function to the set of parameters optimizing the state during the optimization procedure. However, the parameter set optimizing the state should respect the physical symmetries which were explicitly implemented in this section. This is something which should be kept in mind while doing simulations.

4.2 Optimizing the wave function

Eq. (4.3) provides a representation of the expansion coefficients of a wave function. From now on we focus on finding ground-state wave functions. Only for selected values of the parameters a_i, b_i and w_{ij}, Eq. (4.3) approximates the ground state well. The problem thus reduces to finding the set of parameters that approximates the ground state the best. What we mean by "best" will be introduced in this section. The procedure of finding the optimal set of parameters is an iterative one. Thereby, we start with an initial state with random parameters and improve the approximation step by step by updating all the parameters with small amounts until some convergence criterion is reached. How this is done is outlined in this section.

4.2.1 Initializing the wave function

The first step is to initialize the parameters in the ansatz of Eq. (4.3). In the context of neural networks, proper initialization of the parameters is crucial for deep architectures [62]. Generally, initializing the parameters close to the expected values is desirable, as it can make the optimization converge faster. It is natural to assume that the initial weight distribution is centered around zero, because we don't know anything about the correlations in the system yet. The sign of a specific weight cannot be determined a priori. The variance of the distribution can be found by inspecting Eq. (4.3). We see that it is the product of two kinds of functions: the exponential and the cosine hyperbolic. The exponential diverges fast for large positive values of its input where a perturbation of the input makes a large difference in the output. On the other hand, it converges to zero for large negative values of its input, where a perturbation of the input makes negligible difference in the output. The cosine hyperbolic diverges for large negative and large positive values of its input and is insensitive to perturbations of the input around zero. It is natural to assume that the parameterset at the end of the optimization should be such that it results in inputs of the exponential and cosine hyperbolic which are in their most well-behaved region. For the exponential, we can define this region as $[-1, 1]$ where the output neither blows up or is approximately zero. For the cosine hyperbolic, this region lies around the values 1 and -1. From Eq. (4.3), we see that the parameters a_i and w_{ij} appear in a weighted sum of the input values. The sum of the spins (the

magnetization) is extensive in physical systems, i.e. $\sum_i^{N_v} s_z^i \propto N_v$. This means that if we select the weights a_i and w_{ij} to be approximately $1/N_v$, the weighted sums $\sum_i^{N_v} w_{ij} s_z^i$ and $\sum_i^{N_v} a_i s_z^i$ are proportional to 1, i.e. the input of the exponential and cosine hyperbolic is well-behaved.

From the above considerations, it is natural to initialize the weights from a Gaussian distribution with a mean of zero and standard deviation of $1/N_v$, or from a uniform distribution between $-1/N_v$ and $1/N_v$. The weights b_j appear without weighting in the argument of the cosine hyperbolics of Eq. (4.3) and can thus be initialized from a Gaussian with a mean of zero and a standard deviation of 1 or a uniform distribution between -1 and 1.

4.2.2 Update of the parameters

The next step is to update the parameters in the ansatz of Eq. (4.3) in an iterative way, such that it approximates the ground state as well as possible. The main physical ingredient we use to accomplish this is the variational principle. The variational principle states that a given state $|\Psi\rangle$ provides an upper bound on the exact ground-state energy of the system and converges to the ground state if it is tuned such that the energy functional

$$E[\Psi] = \frac{\langle\Psi|\hat{H}|\Psi\rangle}{\langle\Psi|\Psi\rangle} \tag{4.15}$$

is minimal [8]. Following the variational principle, we want to update the parameters in a way which decreases the energy of the wave function. This procedure is called variational Monte Carlo. The derivative of the energy g_w with respect to an unspecified parameter w of the RBM ($w \in \{a_i, b_i, w_{ij}\}$) determines the change in the energy induced by a variation of this parameter. Using the expansion of the wave function $|\Psi\rangle$ in the σ_z-basis (see Eq. (4.1)), we get

$$\begin{aligned} g_w = \frac{\partial}{\partial w}\left(\frac{\langle\Psi|\hat{H}|\Psi\rangle}{\langle\Psi|\Psi\rangle}\right) = &\sum_{\mathcal{S}}\left[\frac{\langle\mathcal{S}|\frac{\partial\Psi^*(\mathcal{S};\mathcal{W})}{\partial w}\hat{H}|\Psi\rangle + \langle\Psi|\frac{\partial\Psi(\mathcal{S};\mathcal{W})}{\partial w}\hat{H}|\mathcal{S}\rangle}{\langle\Psi|\Psi\rangle}\right] \\ &- \sum_{\mathcal{S}}\left[\frac{\langle\Psi|\hat{H}|\Psi\rangle\left(\langle\mathcal{S}|\frac{\partial\Psi^*(\mathcal{S};\mathcal{W})}{\partial w}|\Psi\rangle + \langle\Psi|\frac{\partial\Psi(\mathcal{S};\mathcal{W})}{\partial w}|\mathcal{S}\rangle\right)}{\langle\Psi|\Psi\rangle^2}\right]. \end{aligned} \tag{4.16}$$

By defining $\mathcal{O}_w \equiv \frac{1}{\Psi(\mathcal{S};\mathcal{W})}\frac{\partial\Psi(\mathcal{S};\mathcal{W})}{\partial w}$, Eq. (4.16) becomes

$$\frac{\partial}{\partial w}\left(\frac{\langle\Psi|\hat{H}|\Psi\rangle}{\langle\Psi|\Psi\rangle}\right) = 2\,\mathrm{Re}(\langle\hat{H}\mathcal{O}_w^*\rangle - \langle\hat{H}\rangle\langle\mathcal{O}_w^*\rangle), \tag{4.17}$$

where we used the shorthand notation $\langle \hat{A} \rangle \equiv \frac{\langle \Psi | \hat{H} | \Psi \rangle}{\langle \Psi | \Psi \rangle}$. The expectation values are with respect to the wave function $|\Psi\rangle$. The expectation value of an operator $\hat{A}$ with respect to the wave function $|\Psi\rangle$ can be written as

$$\langle \hat{A} \rangle = \frac{\langle \Psi | \hat{A} | \Psi \rangle}{\langle \Psi | \Psi \rangle} = \sum_{\mathcal{S}} \frac{\Psi(\mathcal{S}; \mathcal{W}) \Psi^*(\mathcal{S}; \mathcal{W})}{\langle \Psi | \Psi \rangle} \frac{\langle \mathcal{S} | \hat{A} | \Psi \rangle}{\Psi(\mathcal{S}; \mathcal{W})}, \tag{4.18}$$

which can be computed using a Markov Chain Monte Carlo (MCMC) random walk in the configuration space of spin configurations $\mathcal{S}$ and treating $\frac{|\Psi(\mathcal{S};\mathcal{W})|^2}{\langle \Psi | \Psi \rangle}$ as the probability distribution and the quantity $A(\mathcal{S}; \Psi) \equiv \frac{\langle \mathcal{S} | \hat{A} | \Psi \rangle}{\Psi(\mathcal{S};\mathcal{W})}$ as the quantity that one measures. The random walk is constructed using the Metropolis-Hastings algorithm [63]. This algorithm consists of the following steps.

- Start with an initial state $\mathcal{S}^0$.
- Until a predefined number of steps N_{steps} is reached, propose a new state $\mathcal{S}'$ by performing an operation on the previous state $\mathcal{S}^{i-1}$ (e.g. by flipping a spin).
- Calculate the ratio r of the probability of $\mathcal{S}'$ and $\mathcal{S}^{i-1}$, i.e. $r = \frac{|\Psi(\mathcal{S}';\mathcal{W})|^2}{|\Psi(\mathcal{S}^{i-1};\mathcal{W})|^2}$, and define the acceptance ratio A as $A = \min(1, r)$.
- Draw a uniform random number u between 0 and 1. If $u < r$, accept the state, i.e. $\mathcal{S}^i = \mathcal{S}'$. If $u \leq r$, reject the state, i.e. $\mathcal{S}^i = \mathcal{S}^{i-1}$.

In our case, the proposed updates are consecutive spin flips with respect to the previous state $\mathcal{S}^{i-1}$. In this fashion, we run over the lattice $N_{MC} \equiv N_{steps}/N_v$ times. One run over the lattice is called a sweep. We don't calculate the quantity $A(\mathcal{S}; \Psi)$ at every step in the Metropolis-Hastings algorithm, but rather once every sweep. The reason is that the calculation of $A(\mathcal{S}; \Psi)$ is rather expensive and calculating it every step brings little extra accuracy due to the autocorrelation time of $A(\mathcal{S}; \Psi)$. The mean of the set of values $\{A(\mathcal{S}^0; \Psi), A(\mathcal{S}^{N_v}; \Psi), ..., A(\mathcal{S}^{N_{MC}N_v}; \Psi)\}$ is an approximation of $\langle \hat{A} \rangle$. In practice, the initial state $\mathcal{S}^0$ in the Metropolis-Hastings algorithm is determined by performing a thermalization run of the Metropolis-Hastings algorithm with a random initial state and taking the final state of this thermalization run as the initial state for the actual production run.

Having calculated the gradient g_w, one can make use of gradient descent techniques to update the parameters, so that it decreases the energy. The different techniques that we have adopted are described below. We denote with w^t the value of an unspecified weight w at iteration step t and with g_w^t the gradient of Eq. (4.17) at iteration step t.

Gradient descent

The simple gradient descent consists of small updates of the parameters in the direction of decreasing energy:

$$w^t = w^{t-1} - l g_w^{t-1}. \tag{4.19}$$

Here, l is the learning rate, a parameter which determines the rate of change of the parameter updates. This simple scheme suffers a number of very important limitations. The first limitation is the fact that the learning rate l is the same for every parameter. If it happens that the gradients for some parameters are very small, and the gradients of other parameters are very large, one needs to adapt the learning rate to the large gradients because otherwise the method would become unstable. The cost function varies the most in the dimensions with large gradients, which means that the updates in these dimensions should be small as to not overshoot the minimum. However, a small learning rate introduces very small updates of the parameters with small gradients which slows down convergence. The second limitation is the fact that this method might converge to a local minimum, rather than the global minimum. For general machine learning problems, this limitation is of less importance because a local minimum is often sufficient. Here, the real ground state corresponds to the global minimum in parameter space. In general, one uses an annealing scheme, i.e. one lowers the learning rate during the iteration process to increasingly improve on the localization of the (local) minimum of the cost function.

Adagrad (adaptive gradient algorithm)

The adaptive gradient algorithm (Adagrad) extends gradient descent with a learning rate adapted to each parameter separately [64]:

$$w^t = w^{t-1} - l_w^{t-1} g_w^{t-1}, \tag{4.20}$$

where l_w^{t-1} is calculated as follows:

$$l_w^{t-1} = \frac{l}{\sqrt{\sum_{t'=0}^{t-1} g_w^{t'2}} + \varepsilon}, \tag{4.21}$$

where $\varepsilon > 0$ is a small number rather so as to avoid division by zero, and l is a parameter which tunes the overall magnitude of the different learning rates. This method assigns a large learning rate l_w^t to the parameters corresponding with small gradients and a small learning rate to parameters with large gradients. A nice byproduct is that the learning rate diminishes upon further iteration (this can also be a disadvantage for some applications because the learning rate decays quite strongly). The Adagrad method overcomes the problem of gradient descent that the gradients can be different in size, but not that the parameter updates stop when

a local minimum is reached. Another disadvantage is that the adaptations to the learning rates take into account the gradients of all the iteration steps, while it may be possible that the behaviour of the gradients changes during the iterations. This problem is overcome in other, similar techniques, such as Adadelta (which takes into account only a certain window of the previous gradients) [65].

Adam (adaptive moment estimation)

Adaptive moment estimation (Adam) uses the concept of momentum to avoid converging to local minima [66]. Momentum means that the parameter updates use information from previous updates to construct the current update. On top of that, an Adagrad-like parameter-specific learning rate is used. The updates are described as follows:

$$w^t = w^{t-1} - \frac{l}{\sqrt{v_w^{t-1}} + \varepsilon} m_w^{t-1}. \tag{4.22}$$

In this case, v_w^t and m_w^t are found as follows:

$$\begin{aligned} m_w^t &= \frac{\beta_1}{1-\beta_1} m_w^{t-1} + g_w^t, \\ v_w^t &= \frac{\beta_2}{1-\beta_2} v_w^{t-1} + {g_w^t}^2. \end{aligned} \tag{4.23}$$

Here, β_1 and β_2 are two parameters which govern the magnitude of influence of the previous gradients and l is a parameter governing the global magnitude of the updates. One can see from these equations that on a given step, the previous gradients influence the present update step, but with a weight exponentially decaying in iteration time. The learning rate is adapted in an Adagrad-like fashion, with an exponentially decaying influence of previous steps. This method is widely used in machine learning. Although the method is a major improvement of the naive gradient descent, it can still be possible that the convergence fails. For example, pathological behaviour of the cost function can be detrimental to this method. The adaptation is gradual, meaning that the adaptation to sudden changes in the structure of the cost function can take some iteration steps. For example, the update steps stay large when a sudden increase in the second derivative of the cost function arises. Another problematic case are sharp bends in the structure of the cost function (see [67] for a visual representation of the concept of momentum).

Stochastic reconfiguration

We now turn to a method which is more suitable for variational Monte Carlo problems [68]. It is well known that the repeated application of the time-evolution operator $\exp\left(-i\hat{H}t\right)$ in imaginary time $t \to -i\tau$,

$$\exp\left(-\hat{H}\tau\right), \tag{4.24}$$

on a random quantum state $|\Psi\rangle$ (which is not orthogonal to the ground state) yields the ground state of the system with Hamiltonian $\hat{H}$. This can be seen by expanding the random state $|\Psi\rangle$ in the eigenbasis of the Hamiltonian $\{|\psi_n\rangle\}$ with energies $\{E_n\}$

$$|\Psi\rangle = \sum_n c_n |\psi_n\rangle , \tag{4.25}$$

where c_n are the expansion coefficients. Applying operator (4.24) on the state $|\Psi\rangle$ yields

$$\exp(-\hat{H}\tau) |\Psi\rangle = \sum_n \exp(-E_n\tau) c_n |\psi_n\rangle . \tag{4.26}$$

We see that the eigenstates with higher energy are multiplied with exponentially lower prefactors, and vanish approximately compared with the ground state when τ becomes large.

While this method works in theory, in practice one needs to resort to approximation methods to this scheme. Indeed, there is no known technique to perform the operation in Eq. (4.26) because the explicit operator is not known. One approach is to approximate the exponential operator of Eq. (4.24) for small τ to first order via a Taylor expansion

$$\exp\left(-\hat{H}\tau\right) \approx 1 - \hat{H}\tau. \tag{4.27}$$

Suppose we have a normalized quantum state $|\Phi^t\rangle$ on a given iteration step t

$$|\Phi^t\rangle = \frac{|\Psi^t\rangle}{\sqrt{\langle\Psi^t|\Psi^t\rangle}}, \tag{4.28}$$

depending on the variational parameters $\{a_i, b_i, w_{ij}\}$. The derivatives of the normalized wave function to an arbitrary variational parameter $w \in \{a_i, b_i, w_{ij}\}$ is

$$\frac{\partial |\Phi^t\rangle}{\partial w} = \frac{1}{\sqrt{\langle\Psi^t|\Psi^t\rangle}} \left|\frac{\partial \Psi^t}{\partial w}\right\rangle - \frac{\left\langle \Psi^t \middle| \frac{\partial \Psi^t}{\partial w} \right\rangle}{\langle\Psi^t|\Psi^t\rangle} \frac{|\Psi^t\rangle}{\sqrt{\langle\Psi^t|\Psi^t\rangle}}. \tag{4.29}$$

We can now write down the Taylor expansion of a quantum state for some perturbation $\boldsymbol{\delta}$ around some parameterset $\mathbf{w} = \{\mathcal{W}\}$:

$$|\Phi^t(\mathbf{w} + \boldsymbol{\delta})\rangle = |\Phi^t(\mathbf{w})\rangle + \sum_{i=1}^{N_p} \delta_i \left|\frac{\partial \Phi^t(\mathbf{w})}{\partial w_i}\right\rangle, \tag{4.30}$$

where the number of parameters is denoted as N_p. We now apply the first-order expansion in Eq. (4.27) of the imaginary-time evolution operator to the normalized state of Eq. (4.28). The result of this operation is written as a first-order Taylor

expansion of the quantum state as in Eq. (4.30). This yields the following equation

$$\exp\left(-\tau\hat{H}\right)|\Phi^t\rangle \approx |\Phi^t\rangle - \tau\hat{H}\,|\Phi^t\rangle = |\Phi^{t+1}\rangle = |\Phi^t\rangle + \sum_{i=1}^{N_p}\delta_i\left|\frac{\partial\Phi^t}{\partial w_i}\right\rangle, \tag{4.31}$$

i.e. applying the imaginary-time evolution operator to the normalized state $|\Phi^t\rangle$ is equivalent to updating the variational parameters $\mathbf{w}$ with the vector $\boldsymbol{\delta}$. By projecting Eq. (4.31) on the derivatives of the wave function, we get

$$-\tau\left\langle\frac{\partial\Phi^t}{\partial w_j}\middle|\hat{H}\middle|\Phi^t\right\rangle = \sum_{i=1}^{N_p}\delta_i\left\langle\frac{\partial\Phi^t}{\partial w_j}\middle|\frac{\partial\Phi^t}{\partial w_i}\right\rangle. \tag{4.32}$$

By casting the indices to vectors, we can write this more concisely as

$$\boldsymbol{\delta} = -\tau\mathbf{S}^{-1}\mathbf{g}, \tag{4.33}$$

where $\boldsymbol{\delta}$ is the vector of Taylor expansion coefficients δ_i, $\mathbf{S}$ is the matrix defined by the elements

$$S_{ij} = \left\langle\frac{\partial\Phi^t}{\partial w_i}\middle|\frac{\partial\Phi^t}{\partial w_j}\right\rangle, \tag{4.34}$$

and $\mathbf{g}$ is the vector containing

$$g_j = \left\langle\frac{\partial\Phi^t}{\partial w_j}\middle|\hat{H}\middle|\Phi^t\right\rangle. \tag{4.35}$$

One observes that the updates of the variational parameters are

$$\mathbf{w}^{t+1} = \mathbf{w}^t + \boldsymbol{\delta}, \tag{4.36}$$

where $\boldsymbol{\delta}$ is found by calculating $\mathbf{S}$ and $\mathbf{g}$ and then inverting $\mathbf{S}$.

The vector $\mathbf{g}$ is calculated by using the expression of Eq. (4.28) for the normalized state $|\Phi^t\rangle$ in terms of the unnormalized state $|\Psi^t\rangle$ and the expression of Eq. (4.29) for the derivatives of the state $|\Phi^t\rangle$ with respect to a parameter w:

$$g_w = \left\langle\frac{\partial\Phi^t}{\partial w}\middle|\hat{H}\middle|\Phi^t\right\rangle = \frac{1}{\langle\Psi^t|\Psi^t\rangle}\left\langle\frac{\partial\Psi^t}{\partial w}\middle|\hat{H}\middle|\Psi^t\right\rangle - \frac{\left\langle\frac{\partial\Psi^t}{\partial w}\middle|\Psi^t\right\rangle}{\langle\Psi^t|\Psi^t\rangle}\frac{\langle\Psi^t|\hat{H}|\Psi^t\rangle}{\langle\Psi^t|\Psi^t\rangle}. \tag{4.37}$$

Calculating the derivatives of the wave functions in the σ_z-basis yields:

$$\left|\frac{\partial\Psi^t}{\partial w}\right\rangle = \sum_{\mathcal{S}}\frac{\partial\Psi^t(\mathcal{S};\mathcal{W})}{\partial w}\,|\mathcal{S}\rangle \tag{4.38}$$

Inserting this in the expression for g_w of Eq. (4.37), we get

$$g_w = \langle\mathcal{O}_w^*\hat{H}\rangle - \langle\mathcal{O}_w^*\rangle\langle\hat{H}\rangle. \tag{4.39}$$

The expression for the matrix $\mathbf{S}$ can be calculated in the same way

$$\begin{aligned} S_{ij} &= \langle \frac{\partial \Phi^t}{\partial w_i} | \frac{\partial \Phi^t}{\partial w_j} \rangle \\ &= \left(\frac{\left\langle \frac{\partial \Psi^t}{\partial w_i} \right|}{\sqrt{\langle \Psi^t | \Psi^t \rangle}} - \frac{\langle \frac{\partial \Psi^t}{\partial w_i} | \Psi^t \rangle}{\langle \Psi^t | \Psi^t \rangle} \frac{\langle \Psi^t |}{\sqrt{\langle \Psi^t | \Psi^t \rangle}} \right) \left(\frac{\left| \frac{\partial \Psi^t}{\partial w_j} \right\rangle}{\sqrt{\langle \Psi^t | \Psi^t \rangle}} - \frac{\langle \Psi^t | \frac{\partial \Psi^t}{\partial w_j} \rangle}{\langle \Psi^t | \Psi^t \rangle} \frac{| \Psi^t \rangle}{\sqrt{\langle \Psi^t | \Psi^t \rangle}} \right). \end{aligned} \tag{4.40}$$

This equation reduces to the following form

$$\begin{aligned} S_{ij} &= \frac{\langle \frac{\partial \Psi^t}{\partial w_i} | \frac{\partial \Psi^t}{\partial w_j} \rangle}{\langle \Psi^t | \Psi^t \rangle} - \frac{\langle \frac{\partial \Psi^t}{\partial w_i} | \Psi^t \rangle}{\langle \Psi^t | \Psi^t \rangle} \frac{\langle \Psi^t | \frac{\partial \Psi^t}{\partial w_j} \rangle}{\langle \Psi^t | \Psi^t \rangle} \\ &= \langle \mathcal{O}^*_{w_i} \mathcal{O}_{w_j} \rangle - \langle \mathcal{O}^*_{w_i} \rangle \langle \mathcal{O}_{w_j} \rangle \end{aligned} \tag{4.41}$$

The parameter updates can thus be calculated by evaluating the expectation values in Eqs. (4.39) and (4.41), which can be done with an MCMC algorithm which samples the quadratic wave function, as described in the beginning of this section. Note the apparent difference in computational complexity between the methods which only depend on the gradients of the wave function (gradient descent, Adagrad and Adam) and the stochastic reconfiguration method. The methods depending on the gradient essentially scale as $\mathcal{O}(N_p)$, where N_p is the number of variational parameters. For the stochastic reconfiguration method, constructing the matrix $\mathbf{S}$ scales as $\mathcal{O}(N_p^2)$ and inverting it scales with $\mathcal{O}(N_p^3)$. This seems to render the stochastic reconfiguration method intractable because the cost of the updates increases rapidly with increasing number of parameters. The problem of inverting the matrix can be circumvented by using more advanced methods to invert matrices. In this work, we use the MINRES-QLP method to invert $\mathbf{S}$ [69]. We use the word inversion here for the real inverse or the pseudo-inverse, depending on whether the matrix is singular or not. The MINRES-QLP method is a method which solves the problem $\mathbf{Ax} = \mathbf{b}$ by finding $\mathbf{x}$ such that $\mathbf{x}$ minimizes $\|\mathbf{Ax} - \mathbf{b}\|_2$, where $\|\mathbf{v}\|_2$ denotes the 2-norm of the vector $\mathbf{v}$, i.e.

$$\|\mathbf{v}\|_2 = \sqrt{\sum_{i=1}^{n} v_i^2}, \tag{4.42}$$

where n is the dimensionality of the vector $\mathbf{v}$. The MINRES-QLP algorithm consists of applying a Lanczos algorithm to the matrix $\mathbf{A}$ to cast it to a tridiagonal form. This algorithm applies the given matrix a number of times (typically lower than the number of rows/columns of the matrix) to a given testvector. This operation has a computational complexity of $\mathcal{O}(k \times N_p^2)$, where k is the number of iterations needed to reduce $\mathbf{A}$ to a tridiagonal form. After the matrix $\mathbf{A}$ has been converted to a tridiagonal matrix, this matrix is used to minimize the norm $\|\mathbf{Ax} - \mathbf{b}\|_2$ in what is called the MINRES-QLP phase of the algorithm. This phase not only minimizes the norm, but als protects for numerical instabilities in the problem. The authors

of the MINRES-QLP algorithm claim that the complexity of this phase is between $\mathcal{O}(9N_p)$ and $\mathcal{O}(14N_p)$. The dominant cost of the whole algorithm is thus the tridiagonalization of the given matrix: $\mathcal{O}(k \times N_p^2)$. We rewrite the matrix $\mathbf{S}$ of Eq. (4.41) as follows

$$\begin{aligned} S_{ij} = \langle \mathcal{O}_{w_i}^* \mathcal{O}_{w_j} \rangle - \langle \mathcal{O}_{w_i}^* \rangle \langle \mathcal{O}_{w_j} \rangle &= \langle (\mathcal{O}_{w_i} - \langle \mathcal{O}_{w_i} \rangle)^* \left(\mathcal{O}_{w_j} - \langle \mathcal{O}_{w_j} \rangle \right) \rangle \\ &= \frac{1}{N_{MC}} \sum_{MC samples} (\mathcal{O}_{w_i} - \langle \mathcal{O}_{w_i} \rangle)^* \left(\mathcal{O}_{w_j} - \langle \mathcal{O}_{w_j} \rangle \right) \\ &= \frac{1}{N_{MC}} \sum_{MC samples} \left[\tilde{\mathbf{o}}^* \tilde{\mathbf{o}}^T \right]_{ij}, \end{aligned} \tag{4.43}$$

where the vector $\tilde{\mathbf{o}}$ is a vector containing the list of $\mathcal{O}_{w_i} - \langle \mathcal{O}_{w_i} \rangle$ measured in the MCMC algorithm. By using this form of the matrix $\mathbf{S}$, the multiplication of matrix and testvector in the Lanczos algorithm can be reduced to the multiplication of the vector $\tilde{\mathbf{o}}$ and the testvector of the Lanczos algorithm, and a multiplication of the resulting scalar with the vector $\tilde{\mathbf{o}}^*$:

$$\mathbf{S}\mathbf{x} = \frac{1}{N_{MC}} \sum_{MC samples} \tilde{\mathbf{o}}^* (\tilde{\mathbf{o}} \cdot \mathbf{x}). \tag{4.44}$$

This process scales as $\mathcal{O}(2N_p)$, i.e. linear in the number of parameters.

By using these considerations, the complexity of the Stochastic reconfiguration scheme is reduced to a process of $\mathcal{O}(N_p)$, in line with the methods using the gradients only.

4.2.3 Convergence criteria

The updates of the parameters, as described in section 4.2.2, are stopped after a suitable number of iteration steps. We stop the updates of the parameters when the wave function converges to the ground state. For this, different criteria can be used. First, a stopping criterium based on the energy can be used. The expectation value of the energy is calculated in every iteration step to construct the estimator of the gradient in Eq. (4.17). This energy can be compared to a reference energy, for example the exact energy of the ground state (if known) or the energy obtained via alternate techniques. When the relative energy error ϵ_E at an iteration step drops below a predefined tolerance value ϵ_E^{tol} (e.g. $\epsilon_E^{tol} < 10^{-3}$), the iterations can be stopped. A second stopping criterium is based on the moving average of the energy

not changing appreciably during some predefined number of iteration steps. A third criterium is using the energy fluctuations, defined as

$$(\Delta E)^2 = \langle \hat{H}^2 \rangle - \langle \hat{H} \rangle^2. \tag{4.45}$$

For the ground state with energy E_{gs} (and any other eigenstate of the Hamiltonian) Eq. (4.45) reduces to

$$(\Delta E)^2 = E_{gs}^2 - E_{gs}^2 = 0. \tag{4.46}$$

When the energy fluctuations ΔE drop below a predefined tolerance value ΔE^{tol} (e.g. $\Delta E^{tol} < 10^{-4}$), the iterations can be stopped. However, this stopping criterion is disfavoured compared to the other two because an additional expectation value needs to be calculated in every iteration, increasing the computing time.

4.2.4 Summary of the algorithm used to determine ground states

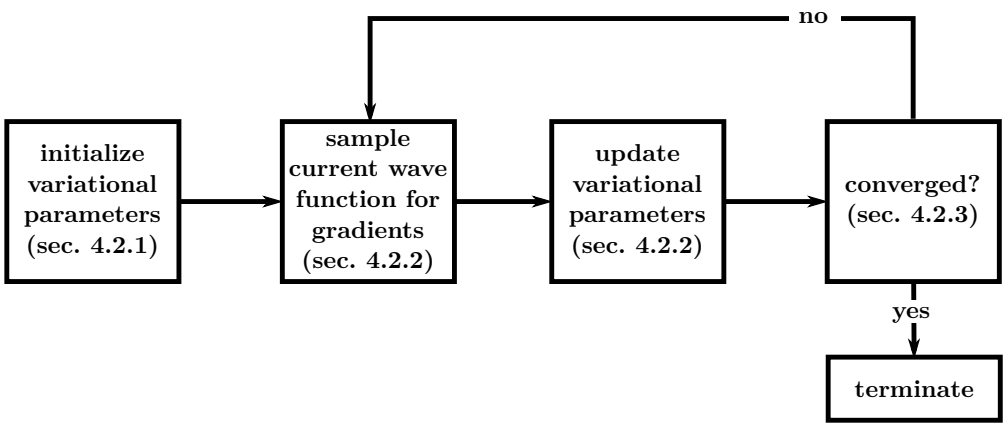

Fig. 4.2: Schematic depiction of the algorithm used to find the ground state of quantum spin systems, as described in section 4.2.

We will now summarize the program flow of the algorithm used to find ground states. This is also shown in figure 4.2.

1. Randomly initialize a state $|\Psi\rangle$ according to the ansatz of Eq. (4.3) which has the right symmetry for the Hamiltonian we try to find the ground state for. This is done by initializing the weights to some small, non-zero value as described in section 4.2.1.

2. Generate a set of N_{MC} spin states $\{\mathcal{S}\}$ by sampling the probability distribution $\frac{|\Psi(\mathcal{S};\mathcal{W})|^2}{\sum_{\mathcal{S}} |\Psi(\mathcal{S};\mathcal{W})|^2}$ using an MCMC algorithm. This is described in section 4.2.2.

3. Use the generated samples of spin states to calculate the expectation values of Eq. (4.17) (and Eq. (4.41) if stochastic reconfiguration is used), and construct the gradients.

4. Update the parameters of the RBM with a chosen scheme (for example gradient descent, Adagrad, Adam or stochastic reconfiguration). This is outlined in section 4.2.2

5. Go to step 2 unless convergence is achieved. This is described in section 4.2.3.

4.3 Theoretical properties and other methods

4.3.1 Theoretical properties

We now investigate the RBM approach for modelling quantum many-body systems from a theoretical point of view by examining the available literature. We first treat the entanglement entropy of many-body wave functions represented by RBMs. The entanglement entropy S_e is defined as

$$S_e(\hat{\rho}_A) = -\operatorname{Tr}(\hat{\rho}_A \log(\hat{\rho}_A)) = -\operatorname{Tr}(\hat{\rho}_B \log(\hat{\rho}_B)), \tag{4.47}$$

where A is a subsystem of the system and B is its complement. Furthermore, $\hat{\rho}_A$ is the density matrix associated with the subsystem A, defined as

$$\hat{\rho}_A = \operatorname{Tr}_B(\hat{\rho}), \tag{4.48}$$

where $\hat{\rho}$ is the density matrix of the system (see Eq. (1.17)) and Tr_B is a trace over the degrees of freedom associated with the subsystem B. The entanglement entropy is a measure for the (quantum) correlations between subsystems A and B [70]. It can be associated with the information gain: when measuring the state of subsystem A, the entanglement entropy determines how much information can be gained about the subsystem B. For quantum states arising from local gapped Hamiltonians (i.e. Hamiltonians for which the ground and first excited states have a difference in energy) it is believed that the entanglement entropy of a subsystem A of size L^d scales (in leading order) as

$$S_e(\hat{\rho}_A) = cL^{d-1}, \tag{4.49}$$

where d is the dimension of the system. This has been proven for 1D systems [71], but remains an open question for systems of more than one dimension. The entanglement entropy is thus proportional to the surface of the subsystem and not to

the volume of the subsystem, as would be expected from the extensivity of (classical) entropy. This is the so-called area-law of entanglement. Because entanglement entropy is a measure for correlations, it follows that the correlations between A and B are present at the surface of the subsystem, i.e. they are short-range.

The entanglement entropy for RBM states was investigated in [72]. They proved that the wave function as described in section 4.1 obeys a volume-law entanglement, i.e. it is capable of modelling quantum states with entanglement entropy scaling as $S_e(\hat{\rho}_A) = cL^d$. When the RBM is restricted to short range correlations (i.e. the weights originating from a hidden node are only non-zero for a finite range of visible spins (cfr. convolutional neural networks)), the states follow an area-law entanglement.

The RBM wave function has also been used to represent quantum states exactly. The first example is the 1D symmetry protected cluster state [73], i.e. the ground state of the following Hamiltonian

$$\hat{H}_{cluster} = -\sum_{i}^{N_v} \sigma_z^{i-1}\sigma_x^i\sigma_z^{i+1}, \tag{4.50}$$

where periodic boundary conditions are implied. The second example are the ground states of the 2D Kiteav Toric code [74] with Hamiltonian

$$\hat{H}_{KTC} = -\sum_{+}\prod_{i\in+}\sigma_z^i - \sum_{\square}\prod_{i\in\square}\sigma_x^i. \tag{4.51}$$

Here, the spins are placed on the edges of a square lattice. The symbol $+$ denotes the four spins neighbouring a vertex of the lattice, i.e. the sum $\sum_+$ runs over all lattice points. The symbol $\square$ denotes a square of the square lattice, i.e. four edges in a square. The sum $\sum_\square$ runs over all squares of the lattice. Also for this Hamiltonian periodic boundary conditions are implied. Determining the wave functions exactly with an RBM entails finding the weights and biases $\{a_i, b_i, w_{ij}\}$ such that Eq. (4.1), with $\Psi(\mathcal{S};\mathcal{W})$ determined by Eq. (4.2), represents the state exactly.

4.3.2 Other methods

Apart from the (very) limited amount of exactly solved quantum many-body problems (such as the transverse field Ising model in section 4.4.1), there exists a wealth of approximation methods of which we briefly mention a selection of important ones here. Most of the methods have the aim to compress the quantum states in a formalism which is exponentially cheaper than the general formalism described in section 1.2, i.e. one wants to go from a formalism with $\mathcal{O}(\exp(N))$ free parameters to a

formalism with a number of free parameters which scales for example polynomially with system size.

The method which is closest to the original formulation of the problem (i.e. diagonalizing the Hamiltonian matrix) is the *exact diagonalization* technique. The exact diagonalization method for finding ground states of quantum systems entails a diagonalization of the Hamiltonian matrix in a subspace of the full Hilbert space. One of the possible methods to find this subspace is the Lanczos algorithm [75]:

1. Start with a random normalized state $|\Psi^0\rangle$, which is not orthogonal to the ground state.

2. If we would apply one step of the gradient descent algorithm (described in section 4.2), the new state will lie in the space spanned by $|\Psi^0\rangle$ and $|\Psi_g^1\rangle$, where $|\Psi_g^1\rangle$ is the normalized state which is proportional to the functional derivative of the energy functional to the current state $|\Psi^0\rangle$

$$|\Psi_g^1\rangle = \left.\frac{\delta \langle\Psi|\hat{H}|\Psi\rangle}{\delta |\Psi\rangle}\right|_{|\Psi_0\rangle}. \tag{4.52}$$

3. If we would apply the gradient descent algorithm n times, the resulting state will lie in the space spanned by $\{|\Psi^0\rangle, |\Psi_g^1\rangle, |\Psi_g^2\rangle, ..., |\Psi_g^n\rangle\}$, which is often called the Krylov space. An orthonormal basis for this subspace can be constructed via an iterative algorithm. In this basis, the Hamilonian matrix is tridiagonal, which can be efficiently diagonalized (in $\mathcal{O}(n\log(n))$ computational time [76]). The eigenstate with the lowest eigenvalue is then found to be an approximation of the ground state.

This approach thus avoids the curse of dimensionality of the matrix diagonalization. However, the problem with this approach is that the different states need to be stored exactly, which requires an exponential amount (as a function of system size) of expansion coefficiens (see section 1.2). This method is thus only feasible for small systems.

Another successful method to find ground states of many-body system is the *density matrix renormalization group* (DMRG) [77]. This method provides the ground state of a system of N degrees of freedom by iteratively constructing it. The construction follows the following steps (we employ the example of a spin system as an illustration) [78]:

1. Start with two spins and find the eigenspectrum of the system using diagonalization of the Hamiltonian matrix.

2. Until the desired system size N is reached, suppose we have at some step $2l$ degrees of freedom. These are divided in two blocks A and B of size l. Two new degrees of freedom are added to the system by placing them between A and B. The resulting (arbitrary) state can then be written as

$$|\Psi^{2l+2}\rangle = \sum_{\theta_A,\theta_B,\lambda_A,\lambda_B} \Psi(\theta_A,\theta_B,\lambda_A,\lambda_B)\,|\theta_A\rangle\,|\theta_B\rangle\,|\lambda_A\rangle\,|\lambda_B\rangle\,, \tag{4.53}$$

   where $|\theta_A\rangle$ ($|\theta_B\rangle$) is the basis state associated with block A (B) and $|\lambda_A\rangle$ ($|\lambda_B\rangle$) is the basis state associated with the new degree of freedom next to block A (B).

3. The eigenspectrum of the new system is found and a new basis is constructed for the new blocks $A+$ ($B+$) (formed by appending the new spin next to A (B) to the block A (B)). This new basis for $A+$ ($B+$) is defined by constructing the density matrix $\hat{\rho}_{\Psi_{gs}^{2l+2}}$ of the ground state and taking the reduced density matrix $\hat{\rho}_{A+}$ ($\hat{\rho}_{B+}$) of block $A+$ ($B+$) defined as

$$\hat{\rho}_{A+} = \mathrm{Tr}_{B+}(\hat{\rho}_{\Psi_{gs}^{2l+2}}), \tag{4.54}$$

   after which one diagonlizes $\hat{\rho}_{A+}$ ($\hat{\rho}_{B+}$) and keeps only the D eigenstates corresponding to the D highest eigenvalues. In this way, the number of basis states of the blocks does not grow during the iterations, but stays constant. This reduces the cost of representing a state compared to keeping all the basis states, in which case the number of basis states of the blocks grows exponentially with block size.

The DMRG procedure naturally results in the concept of *tensor networks* (TN). It can be proven that the states resulting from a DMRG algorithm are states which fall in the class of matrix product states (MPS) [79]. MPS are states which can be written as a product of matrices, i.e. every degree of freedom corresponds to a unique matrix. The matrices are contracted with each other to yield the expansion coefficients of a quantum state. This can be written down as

$$|\Psi\rangle = \sum_{\{\lambda^i\}} \mathrm{Tr}(\mathbf{A}(\lambda^1)\mathbf{A}(\lambda^2)...\mathbf{A}(\lambda^N))\,|\lambda^1\lambda^2...\lambda^N\rangle\,, \tag{4.55}$$

where λ^i are the different basis states associated with the i-th degree of freedom (e.g. the σ_z-basis states of Eq. (1.11)) and $\mathbf{A}(\lambda^i)$ is a matrix associated with basis state λ^i. The trace reflects the periodic boundary conditions. Loosely speaking, the

contraction of the matrices represents the interaction between two neighbouring degrees of freedom. The matrices $\mathbf{A}$ have a small dimension compared to the total number of basis states. This dimension is called the bond dimension and determines how many parameters are available to describe the quantum states. The bond dimension is intimately related to the entanglement entropy S_e of Eq. (4.47) of the state. The bond dimension provides a bound on the maximal entanglement entropy of an MPS [80]. The MPS can be generalised to a general tensor network, where tensors with an arbitrary number of indices are contracted in a specific way. This makes it possible to for example define tensor networks for systems in more than one dimension. It can be proven that tensor networks fall in the class of models which can describe states with area-law entanglement [80]. Finding the parameters of tensor networks can be done for example by using the variational monte carlo approach, as described in section 4.2 [81] or by a DMRG-like algorithm, where one iteratively optimizes every tensor via a diagonalization procedure [82].

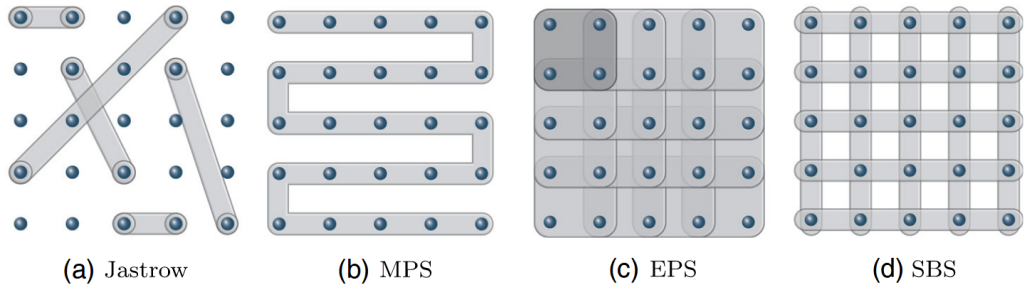

Fig. 4.3: Schematic depiction of the ansätze described in Ref. [61]. The shaded areas depict how the different degrees of freedom on a 2D lattice functionally depend on each other in the ansatz. The different ansätze are (a) the Jastrow ansatz, (b) matrix product states, (c) entangled plaquette states and (d) string bond states. Figure adapted from [61].

The RBM states presented in this chapter can be related to other variational ansätze for many-body quantum states. It has been proven that the RBM states can be related to tensor network states [74]. Spefically, Ref. [74] provided constructive algorithms to write MPSs as RBMs and vice versa. Because RBMs can represent states with volume-law entanglement (see section 4.3.1), which is only possible for MPSs when the bond dimension grows exponentially with system size, the RBMs can represent these states with an exponentially lower amount of resources. However, this does not mean that RBMs are better as a variational ansatz: volume-law entanglement is by far not always required (see section 4.3.1) and the success not only depends on the representational power but also on the algorithms to actually find the states. Another related class of states [61] is the Jastrow ansatz, which can be written as

$$|\Psi\rangle_{Jastrow} = \sum_{\mathcal{S}} \prod_{s_z^i < s_z^j} f_{i,j}(s_z^i, s_z^j)\, |s_z^1 s_z^2 ... s_z^{N_v}\rangle\,, \tag{4.56}$$

where $f_{i,j}(s_z^i, s_z^j)$ is a function representing two-body interactions between spin i and spin j. The RBM can represent any Jastrow function with $N_h = (N_v - 1)N_v/2$ hidden units. Ref. [61] also showed that RBMs can be written as string-bond states (SBS), which are a product of different MPSs snaking over a 2D lattice:

$$|\Psi\rangle_{SBS} = \sum_{\mathcal{S}} \prod_{j} \mathrm{Tr}(\prod_{i \in j} A_{i,j}(s_z^i)) \, |s_z^1 s_z^2 ... s_z^{N_v}\rangle \,, \tag{4.57}$$

where j is the index identifying the different snakes over the lattice and i is the index identifying the lattice vertices the snakes run over. Finally, Ref. [61] showed that short-range RBMs can be written as entangled plaquette states (EPS), defined as

$$|\Psi\rangle_{EPS} = \sum_{\mathcal{S}} \prod_{i=0}^{M} P_i(\mathcal{S}) \, |s_z^1 s_z^2 ... s_z^{N_v}\rangle \,, \tag{4.58}$$

where i identifies local subsets of spins (plaquettes) and $P_i(\mathcal{S})$ is a function of the spins in these subsets. The different ansätze are schematically depicted in figure 4.3.

4.4 Quantum spin systems

While the RBM approach outlined in sections 4.1 and 4.2 works for any finite (discrete) quantum system, we will use the method to find ground states and properties of quantum spin systems. These spin systems describe models for magnetism. While this is interesting in itself, the most interesting feature of quantum spin systems is that they serve as an excellent prototypical quantum system, exhibiting features also found in other systems. These features include phase transitions, superposition, entanglement, frustration effects, etc. The fact that these models are discrete in nature make them attractive from a computational point of view.

4.4.1 Transverse field Ising model

The transverse field Ising model (TFI) is defined by the Hamiltonian of Eq. (1.14). The TFI model has an exact solution in one dimension. This was given by Pfeuty in 1970 [83] by using the Jordan-Wigner transformation and solving the resulting eigenvalue problem. The result is a ground-state energy with magnitude

$$E_{gs} = -j \sum_{k} \Lambda_k, \tag{4.59}$$

where Λ_k is given by

$$\Lambda_k = \sqrt{1 + 2g\cos(k) + (g)^2}. \tag{4.60}$$

The variable k runs over the discrete set of possible wavevectors in the first brillouin zone, i.e. $k \in [-\pi, \pi]$, where k can have values (for even N_v, which we will always choose)

$$k = -\pi(N_v - 1)/N_v, -\pi(N_v - 3)/N_v, ..., \pi(N_v - 3)/N_v, \pi(N_v - 1)/N_v. \quad (4.61)$$

This exact solution is extremely useful with the eye on benchmarking numerical calculations.

As stated in the introduction, the TFI model undergoes a phase transition between a disordered state (with zero absolute magnetization $\langle|\hat{s}_z|\rangle$) and an ordered state (with non-zero absolute magnetization $\langle|\hat{s}_z|\rangle$). Absolute magnetization is defined as

$$\langle|\hat{s}_z|\rangle = \frac{\left\langle \left| \sum_{i=1}^{N_v} \sigma_z^i \right| \right\rangle}{N_v}, \quad (4.62)$$

which is the quantum counterpart of the classical magnetization of spin systems defined in Eq. (2.3).[1] In one dimension the phase transition occurs at $j = h = 1$, or $g_c = 1$. In two-dimensions, the TFI model undergoes a phase transition at $j = 1, h = 3.05266...$, or $g_c = 3.05266...$. For $h \gg j$, the system is in a disordered state, whereas for $j \gg h$ an ordered state with non-vanishing magnetization is found. One can prove that the phase transition is second order [84]. This means that the response function of the thermodynamic variables, e.g. the specific heat or the susceptibility, diverge at the critical point. The 1D TFI belongs to the same universality class as the 2D classical Ising model described in section 1.1 [85]. A universality class is defined as a collection of models which share the same large-scale behaviour at a second order critical point [3]. One of the consequences is that they share the same critical exponents.

The TFI model has a spin flip symmetry (Z_2-symmetry) meaning that the Hamiltonian is invariant under the operator $\hat{F}$, defined in Eq. (4.11). In the magnetically ordered phase ($h < 1$ for one-dimensional systems), this symmetry is broken and there exist two degenerate ground states with opposite total spin.

With the eye on the discussion of chapter 5, we mention the characteristics of the first excited states of the 1D TFI model [84]. For ordered states, the first excited states correspond to states with two domain walls[2], i.e. a collection of neighbouring spins with arbitrary size $0 < l < N_v$ is flipped with respect to the ground state. States with domain walls of size l and of size $N_v - l$ have opposite magnetization. This means that a linear combination of these two kinds of states (with equal weights) has vanishing

[1] We use the *absolute* magnetization because the ground state in the ordered phase is doubly degenerate with opposite magnetizations.

[2] There are two domain walls due to the periodic boundary conditions. With open boundary conditions, the first excited state would correspond to states with one domain wall.

magnetization. Because the states represented by RBMs imply spin flip symmetry, these states indeed have equal weight, resulting in a vanishing magnetization when measured. For disordered states, the first excited state is obtained by flipping a spin in the ground state.

The TFI model is often used as a benchmark for new algorithms that are designed to solve the quantum many-body problem. Indeed, the TFI model is one of the simplest models for spins on a lattice, while non-trivial in terms of its interactions.

4.4.2 Antiferromagnetic Heisenberg model

The antiferromagnetic Heisenberg model (AFH) is defined by the Hamiltonian

$$\hat{H}_{AFH} = \sum_{\langle i,j \rangle} \sigma_x^i \sigma_x^j + \sigma_y^i \sigma_y^j + \sigma_z^i \sigma_z^j, \tag{4.63}$$

where $\sum_{\langle i,j \rangle}$ denotes a sum over all pairs of nearest neighbours and σ_x^i, σ_y^i, σ_z^i are the Pauli matrices defined in Eq. (1.8), acting on the spin with index i.

The AFH model is invariant under the total spin operator,

$$\hat{\mathbf{S}} = \sum_{i=1} N_v \begin{pmatrix} \hat{s}_x^i \\ \hat{s}_y^i \\ \hat{s}_z^i \end{pmatrix}. \tag{4.64}$$

Indeed, the Hamiltonian commutes with $\hat{\mathbf{S}}$,

$$[\hat{H}_{AFH}, \hat{\mathbf{S}}] = 0. \tag{4.65}$$

This can be seen by using the commutation and anticommutation relations of the Pauli matrices of Eqs. (1.9) and (1.10). The AFH model is also invariant under spin rotations (i.e. $SU(2)$ invariance)

$$[\hat{H}_{AFH}, \hat{R}(\boldsymbol{\theta})] = 0. \tag{4.66}$$

The spin rotation operator $\hat{R}(\boldsymbol{\theta})$ is defined in Eq. (4.14). To prove Eq. (4.66), the exponential in the spin rotation operator can be expanded in a Taylor series and rewritten in terms of its real and complex parts, yielding

$$\hat{R}(\boldsymbol{\theta}) = \cos(\theta)\hat{I} - i\boldsymbol{\sigma} \cdot \boldsymbol{e}_{\boldsymbol{\theta}} \sin(\theta). \tag{4.67}$$

The first term commutes with the Hamiltonian because it is proportional to the unit operator. The second term commutes because it is proportional to the total spin operator, which commutes with the Hamiltonian (see Eq. (4.65)).

The one-dimensional AFH model has a disordered ground state. This property follows directly from the Mermin-Wagner theorem, which states that a continuous symmetry of the system (in this case the $SU(2)$-symmetry) cannot be broken in classical systems of dimension $d < 3$ [86]. For quantum systems, we can make use of the relation between d-dimensional quantum systems in the ground state and $(d+1)$-dimensional classical systems at finite temperature (see section 1.4) to find that a continuous symmetry cannot be broken in a system of dimension $d < 2$. The SU(2)-symmetry of the AFH model can thus be broken in the two-dimensional system. This is the case for the AFH model, which exhibits a non-zero staggered magnetization (Néel order) in two dimensions [87].

Both the one and two dimensional AFH models exhibit long-range correlations in their ground state, decaying polynomially with distance [87, 88].

4.5 Results

4.5.1 Ground-state energy

Tab. 4.1: Learning rates l used for figures 4.4, 4.5, 4.6 and 4.7 (see section 4.2.2 for their definition). These learning rates are the ones which lead to the lowest energy.

	1D TFI	1D AFH	2D TFI	2D AFH
gradient descent	0.0005	0.0005	0.0005	5×10^{-5}
Adam	0.007	0.003	0.005	0.0005
Adagrad	0.05	0.05	0.04	0.005
stochastic reconfiguration	0.4	1.0	0.1	0.2

We will start with the situation $N_h = N_v$ and reproduce the results of [57]. We first compare the different minimization methods on small systems. For all the different runs, we used 1000 Monte Carlo samples (from 1000 sweeps over the lattice, see section 4.2.2) per iteration step to estimate the expectation values of Eqs. (4.41) and (4.17). This number is a balance between accuracy and computation time per iteration step: when the number of MC samples is increased, the accuracy of the gradients increases, but this also increases the computation time per step. It is well-known in machine learning that noisy estimators of gradients perform comparably well in finding the minimum of a cost function compared with more accurate estimates [89]. However, noisy estimators come with an increase in the

number of iteration steps needed for convergence. This technique is called Stochastic Gradient Descent (SGD). The learning rates l are summarized in table 4.1. These learning rates yield the lowest energy compared to other learning rates. When the Adam method in Eq. (4.22) is used, we take $\beta_1 = 0.9$, $\beta_2 = 0.99$ in Eq. (4.23).

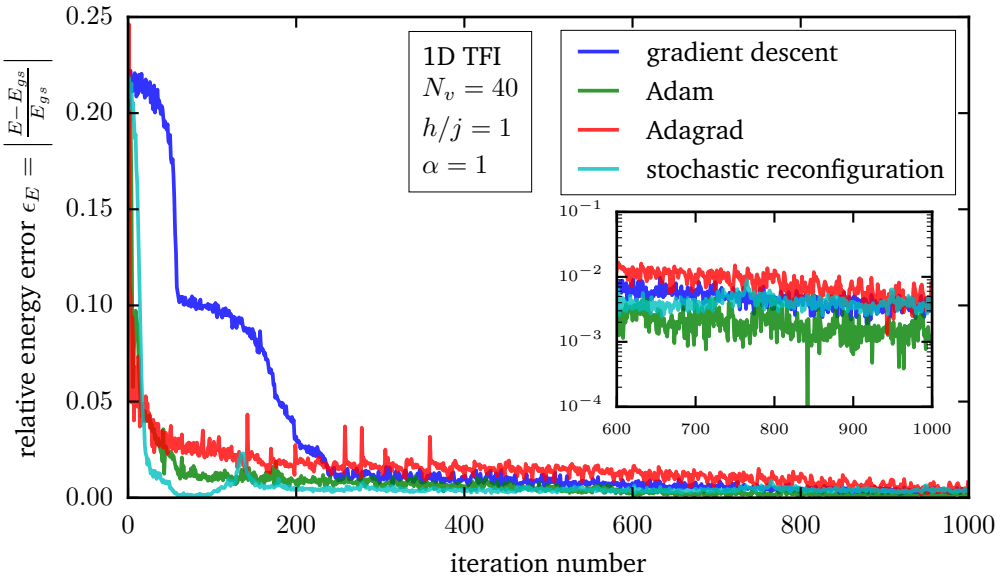

Fig. 4.4: Convergence of the energy of the 1D TFI system ($j/h = 1$, $N_v = 40$ and $\alpha = 1$) as a function of the number of iteration steps for different learning algorithms. The energy error is with respect to the exact ground-state energy E_{gs} of Eq. (4.59) ($E_{gs}/N_v = -1.2736j$). The inset shows the last 400 steps on a logarithmic scale in order to illustrate the magnitude of the fluctuations of the energy once convergence is reached.

Figure 4.4 shows the convergence of the energy for the one-dimensional TFI model at $g = h/j = 1$ relative to the exact result described in section 4.4.1 (see Eq. (4.59)). We see that all the different learning algorithms perform well in terms of energy convergence. Differences can be seen in the convergence speed[3], where the stochastic reconfiguration scheme converges the fastest and the gradient descent method the slowest. Although faster, the stochastic reconfiguration does not appear to be the optimal method in terms of attaining the lowest energy value. In this case, the Adam method is capable of attaining the lowest energy values. Note, however, that the stochastic reconfiguration method reaches its lowest energy value after approximately 100 iteration steps, after which the energy increases slightly and starts to converge again but more slowly. This can point in the direction that a better fine-tuning of the learning rate would result in a more monotonous decrease of the energy. All learning schemes perform well, and result in an accuracy at the subpercent level after 1000 iterations.

[3]We use the term *speed* to denote how fast some quantity converges in terms of iteration steps. The term *computation time* denotes the time it takes in seconds for a certain computation to be performed.

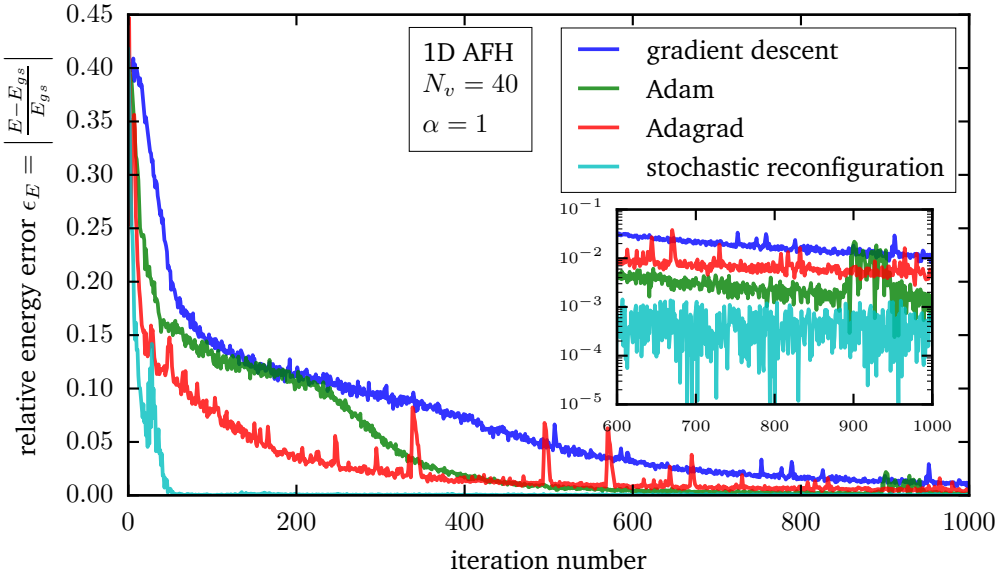

Fig. 4.5: Convergence of the energy of the 1D AFH system ($N_v = 40$ and $\alpha = 1$) as a function of the number of iteration steps for different learning algorithms. The energy error is with respect to the ground-state energy E_{gs} obtained with quantum Monte Carlo simulations using the ALPS software [90] ($E_{gs}/N_v = -1.7746$). The inset shows the last 400 steps on a logarithmic scale in order to illustrate the magnitude of the fluctuations of the energy once convergence is reached.

Figure 4.5 shows the energy convergence of the 1D AFH model. The energy is relative to the energy of quantum Monte Carlo simulations of the AFH at finite temperature extrapolated to zero temperature [90]. In figure 4.5, the power of the stochastic reconfiguration method is more clear than in the one-dimensional TFI case. Here, we see that the speed of convergence is much faster than all the other methods, especially than the gradient descent method. Stochastic reconfiguration also reaches the lowest energy compared to the other methods.

Figure 4.6 shows the energy convergence of the 2D TFI model at $j/h = g = 3.05266$ relative to the energy obtained via exact diagnonalization [91]. We see that all optimization methods converge fast and approximate E_{gs} well. The final energy of the stochastic reconfiguration method is again the lowest, in this case comparable with the Adam method.

Figure 4.7 shows the energy convergence for the 2D AFH model relative to quantum Monte Carlo simulations [92]. This figure provides the most compelling information on the performance of the learning methods. Here, we see that only the stochastic reconfiguration method reaches acceptable energy values, whereas the Adagrad and gradient descent methods are barely able to reach 10 percent of the true ground-state energy in an acceptable amount of iteration steps. For the Adam method, matters

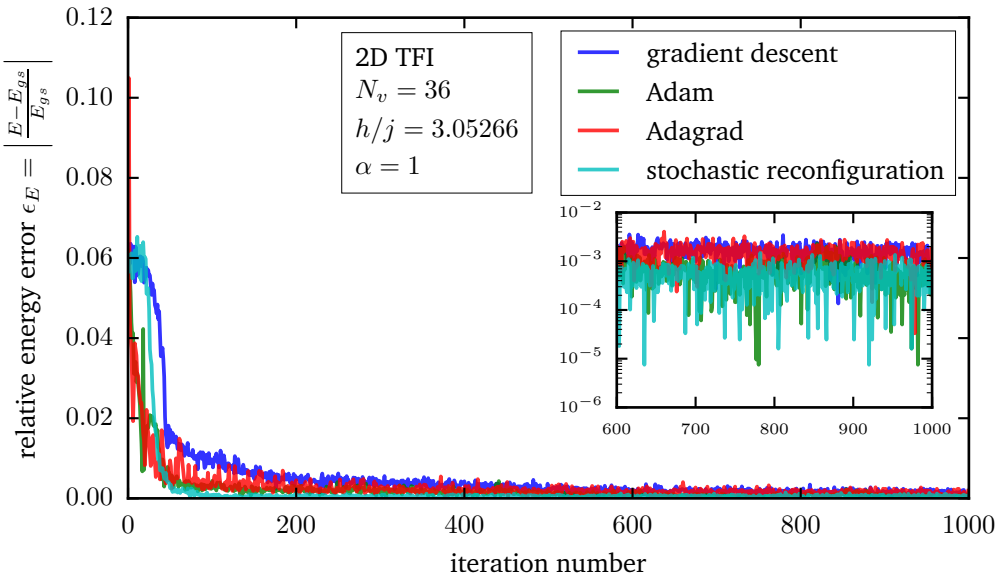

Fig. 4.6: Convergence of the energy of the 2D TFI system ($j/h = 0.32758$, $N_v = 36$ and $\alpha = 1$) as a function of the number of iteration steps for different learning algorithms. The energy error is with respect to the ground-state energy E_{gs} obtained via exact diagonalization results [91] ($E_{gs}/N_v = -3.2473j$). The inset shows the last 400 steps on a logarithmic scale in order to illustrate the magnitude of the fluctuations of the energy once convergence is reached.

are even worse because the energy diverges and settles at an error of 30 percent. This behaviour is most likely due to large variations in the Hessian matrix $\mathbf{C}$ of the energy surface, defined as

$$\mathbf{C} = \begin{pmatrix} \frac{\partial^2 E}{\partial w_1 \partial w_1} & \cdots & \frac{\partial^2 E}{\partial w_1 \partial w_{N_p}} \\ \vdots & \ddots & \\ \frac{\partial^2 E}{\partial w_{N_p} \partial w_1} & & \frac{\partial^2 E}{\partial w_{N_p} \partial w_{N_p}} \end{pmatrix}, \tag{4.68}$$

where N_p is the number of parameters in the model. This is the matrix describing the curvature of the energy surface. Large values in the Hessian may cause divergent behaviour for methods where momentum is involved (for a nice interactive account of these effects, see [67]).

In conclusion, using an RBM representation of the ground-state wave function, the energy converges to within 1% of the known value after 1000 iterations or less for all the considered models, even with a low amount of variational parameters. The stochastic reconfiguration technique performs systematically better in terms of convergence speed. With the stochastic reconfiguration method, initial fluctuations are present, after which the energy drops fast. The other minimization techniques require a larger amount of iteration steps to converge to a low energy value. Also in

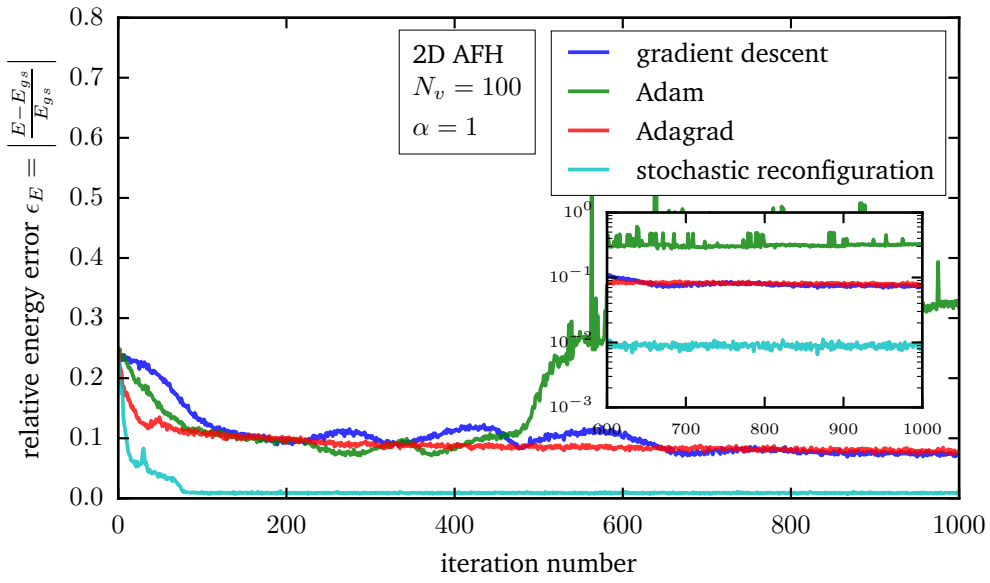

Fig. 4.7: Convergence of the energy of the 2D AFH system ($N_v = 100$ and $\alpha = 1$) as a function of the number of iteration steps for different learning algorithms. The energy error is with respect to the ground-state energy E_{gs} obtained with quantum Monte Carlo simulations of [92] ($E_{gs} = -2.6862$). The inset shows the last 400 steps on a logarithmic scale in order to illustrate the magnitude of the fluctuations of the energy once convergence is reached.

terms of energy minimization, the stochastic reconfiguration technique converges to the lowest energy for all but the 1D TFI system. From these considerations, it is clear that the stochastic reconfiguration method is the most favorable for wave function optimization. This comes with the cost of slightly increased computation times due to the matrix inversion in the method, as shown in Eq. (4.33). In the remainder of this work, the stochastic reconfiguration method will be used to perform minimization of variational wave functions, unless stated otherwise.

4.5.2 Scaling

We now investigate how the procedure to find variational wave functions with RBMs scales with the system size N_v and with the number of variational parameters N_h. For this, we investigate the 1D TFI model at $g = h/j = 1$. Figure 4.8 displays the scaling of the proposed algorithm with system size. We simulate system sizes from 10 to 100 spins. We choose $\alpha = 4$ (see Eq. (4.9)) as a compromise between accuracy and computation time per step. The CPU-times are measured and a power law is fitted to these data points. The CPU time scales approximately quadratically with system size: $t_{CPU} = 0.00673 N_v^{1.929} - 0.217$. This can be attributed to the fact that the size of the weight matrix of Eq. (4.8) scales quadratically with system size. This matrix is used in the evaluation of the expectation values of Eqs. (4.17) and

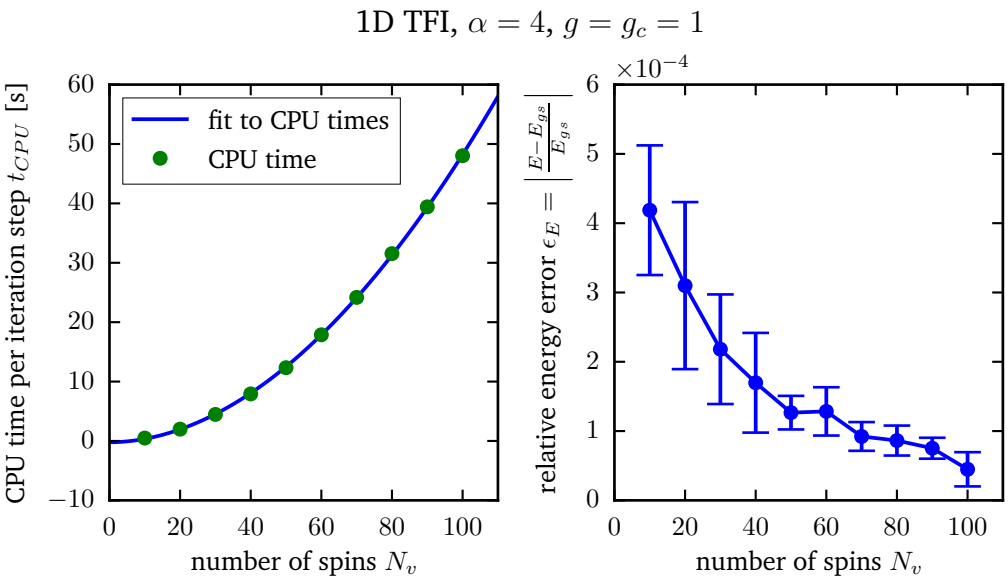

Fig. 4.8: Scaling as a function of system size for the 1D TFI model. Left: scaling of CPU time t_{CPU} per iteration step as a function of system size. The green line is a power law ($t_{CPU} = 0.00673 N_v^{1.929} - 0.217$) fit to the CPU-times. Right: scaling of energy convergence ϵ_E (with standard error) with system size.

(4.41) required to perform the optimization steps. From figure 4.8, we also see that the energy error becomes smaller with increasing system size. This can be appreciated by noting that larger system sizes induce more variational parameters ($N_h = \alpha N_v$). The variational parameters need to describe the correlations between the spins. These correlations decay with distance. Effectively, this means that the number of parameters grows faster than the correlation in the system, facilitating the description of these correlations for larger systems.

The scaling with the number of variational parameters is depicted in figure 4.9. We simulated the TFI model with $N_v = 40$ for values of α between 1 and 16. We see that the CPU-times per step scale polynomially with system size ($t_{CPU} = 1.147\alpha^{1.283} + 0.908$). The scaling is close to linear in the number of parameters. This is to be expected from the MINRES-QLP matrix inversion, which scales linearly in the number of variational parameters (see Eq. (4.43) and its discussion). The energy error decreases monotonously with increasing number of variational parameters. This is expected from the variational principle, but is still reassuring as the method is non-linear in the variational parameters and its success does not only depend on theoretical bounds, but also on the optimization algorithm. The optimization of the variational parameters can fail, even when it is theoretically feasible. This property is well-known in the machine learning community and especially arises in the field of deep learning [62]. Because RBMs are shallow networks, this problem can still be present but should be under control.

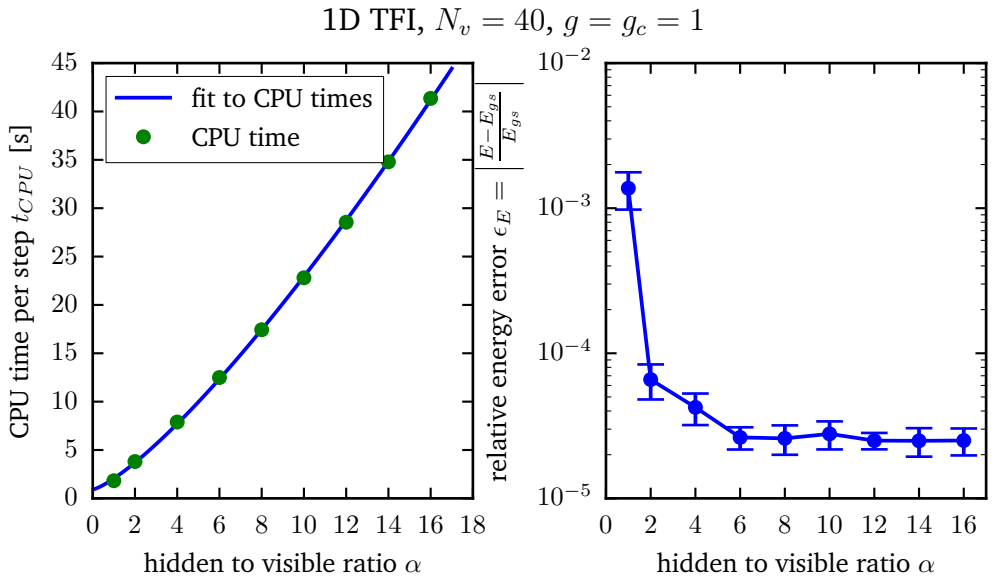

Fig. 4.9: Scaling as a function of the ratio of the number of hidden variables to the number of visible variables α. Left: scaling of CPU time t_{CPU} per iteration step as a function of the number of parameters. The green line is a power law ($t_{CPU} = 1.147\alpha^{1.283} + 0.908$) fit to the measured CPU times. Right: scaling of energy convergence ϵ_E (with standard error) with the number of parameters.

4.5.3 Energy fluctuations

In section 4.2.3, we showed that the relative energy fluctuations $\Delta E/E$ defined in Eq. (4.45) provide a good way to verify if the variational wave function converged to the ground state. A proper representation of the ground state should have a low value for $\Delta E/E$ (see Eq. (4.46)). The relative energy fluctuations are depicted in figure 4.10. We see a systematic improvement to lower values of the relative energy fluctuations when increasing the number of variational parameters.

4.5.4 Comparison with literature

We compare the relative energy error on the ground state for the 2D TFI and AFH models with results in the available literature. We optimize the 2D TFI ground state on a square lattice with size 6×6 at the critical point ($g_c = 3.05265897...$) and the 2D AFH model on a square lattice with size 10×10. The energy convergence is measured for different values of α (see Eq. (4.9)), relative to the values of [91] (TFI) and [92] (AFH), which we also used in section 4.5.1. The results are shown in figure 4.11. For the TFI model, we compare with correlator product states (CPS) [93] and tree tensor networks (TTN, see tensor networks in section 4.3.2) [94]. We see that the relative energy error is lower than that obtained with CPS and approaches

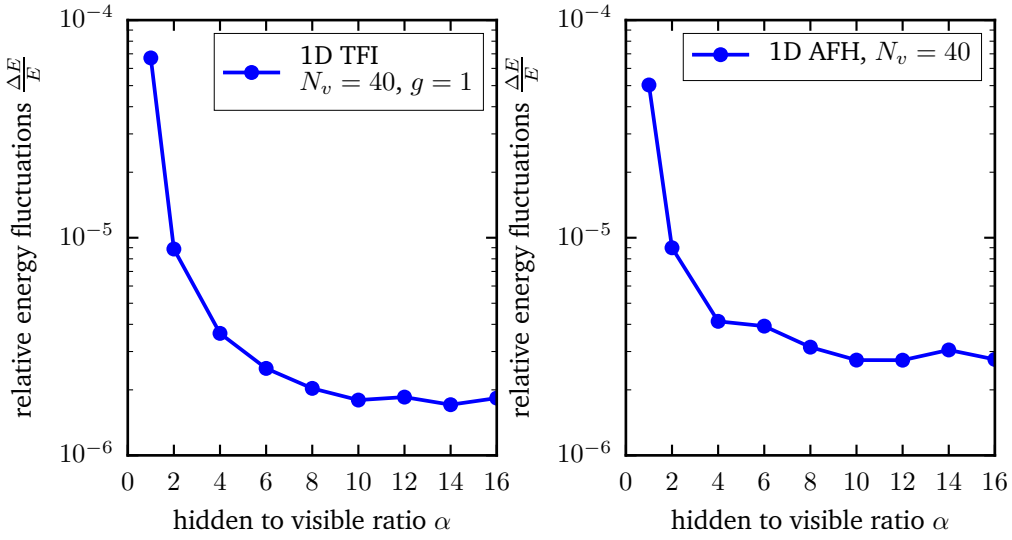

Fig. 4.10: Relative energy fluctuations $\Delta E/E$ as a function of the ratio of the number of hidden variables to the number of visible variables α for the 1D TFI and the 1D AFH models. This measure is important to determine the quality of the resulting ground state, where the energy fluctuations should vanish.

the value quoted for TTN. It is interesting to note that the CPS and TTN ansätze have 65536 and $\mathcal{O}(10^8)$ parameters respectively, whereas for $\alpha = 16$ we have 20736 parameters in the weight matrix of Eq. (4.8), of which 576 are unique. For the AFH model, we compare with entangled plaquette states (EPS, see section 4.3.2) [95] and projected entangled pair states (PEPS, see tensor networks in section 4.3.2) [96]. We see that our RBM ansatz does not achieve the accuracy of these two methods. The relative energy error drops with increasing α, but rises again[4] for $\alpha = 16$. However, Refs. [57, 58] reported lower relative energy errors with the same RBM ansatz. It is unclear why this discrepancy between this work and other RBM implementations is present, but one possibility is that the hyperparameters (i.e. the learning rate, the learning rate annealing, the number of MC samples, ...) are better tuned in the works of [57, 58] compared to our work.

[4] This may be due to too little runs. For $\alpha = 16$ we only did 3 different runs because of the higher computational cost. We did of the order of 20 runs for the other values of α.

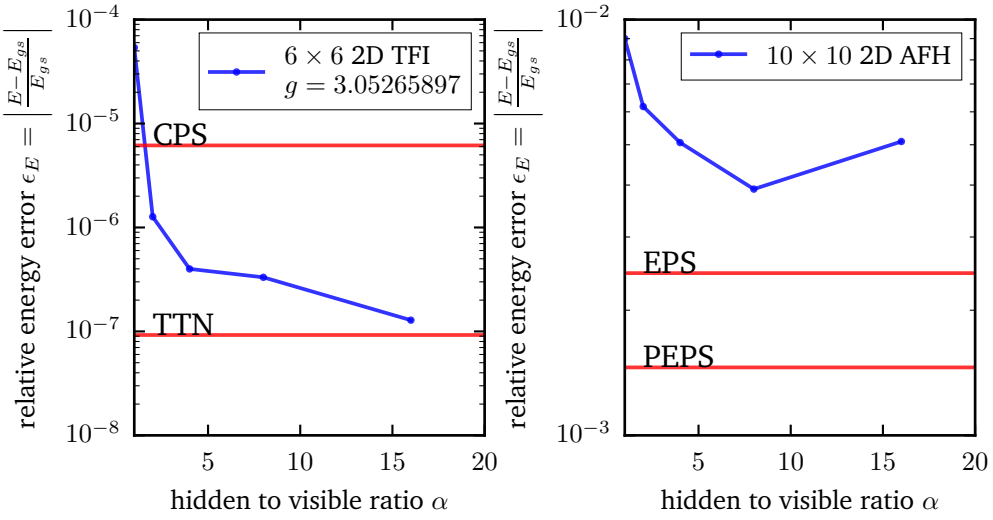

Fig. 4.11: Comparison of relative energy errors on the ground states of the 6×6 critical 2D TFI model and the 10×10 2D AFH model with other variational ansätze. The energies are relative to the energy obtained with exact diagonalization [91] for the TFI and with quantum Monte Carlo [92] for the AFH. We compare our results with correlator product states (CPS) and tree tensor networks (TTN) for the TFI and with entangled plaquette states (EPS) and projected entangled pair states (PEPS) for the AFH.

4.5.5 RBM representation as a function of iteration step

An interesting aspect of the RBM representation of quantum states is how exactly they represent the quantum states. This information is encoded in the weight vectors of Eq. (4.10). The weight vectors encode the correlations present in the wave function. This is done, however, in a complicated way judging from Eq. (4.10). The weight vector defines a linear mapping from the configuration to a scalar value, where every spin is multiplied with its unique weight after which they are added together. To this scalar value one adds the bias b_j and feeds the result to a cosine hyperbolic. The complexity lies in the product of cosine hyperbolics: every cosine hyperbolic contains the scalar product of the wave vector and the spin configuration, but with the wave vector periodically translated by a unique amount of indices (see Eq. (4.7)). The result is that some spin configurations are favoured by the weight vector, but disfavoured when the weight vector is periodically translated. However, because the minimal value of the cosine hyperbolic is one, a spin configuration which is strongly favoured by the weight vector still results in a high expansion coefficient $\Psi(\mathcal{S};\mathcal{W})$ because it is the product of at least one high value and numbers which are greater than one. This implies that the spin configuration for which the scalar product with the weight vector is the highest is an example of an important spin

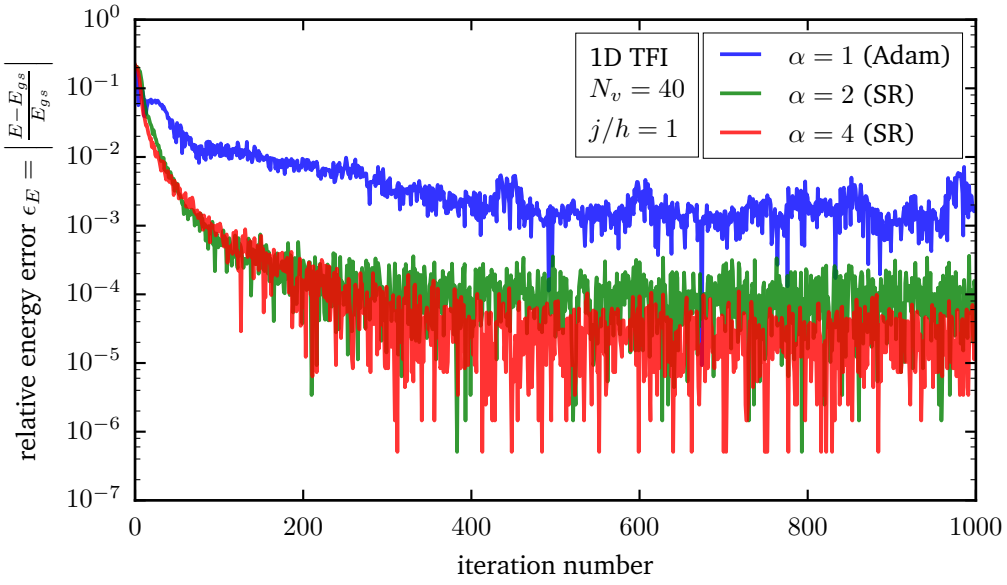

Fig. 4.12: The relative energy error as a function of iteration number for the wave functions used to make figures 4.13, 4.14 and 4.15. The wave functions are of the 1D TFI model with $N_v = 40$ at the critical point ($g = 1$) for different values of α. For $\alpha = 1$ the Adam method is used, while for $\alpha = 2$ and $\alpha = 4$ the stochastic reconfiguration method is used.

configuration in the expansion of the wave function. This spin configuration can thus be identified as a peculiar feature of the system.

We investigate the wave vectors of the RBM representation of the ground state of the 1D TFI model at $g = g_c = 1$ with $N_v = 40$. Figures 4.13, 4.14 and 4.15 show the evolution of the weight vectors as a function of iteration steps for different values of α. For figure 4.13, the Adam method of Eq. (4.22) is used for the parameter updates because lower energy values are obtained with this method compared to stochastic reconfiguration (see also figure 4.4). For the other figures, stochastic reconfiguration obtained the lowest energies. Figure 4.12 shows the energy convergence of these optimization runs for reference.

Figure 4.13 ($\alpha = 1$) shows the convergence to a weight vector where 3 weights centered around one spin are large and have the same sign, while all the other weights are small. This favours spin configurations where three neighbouring spins are in the same state, while all the others are random. This reminds of the $\hat{H}_z$-part of the Hamiltonian in Eq. (1.14), where every term favours the same spin state for two neighbouring spins. Spins other than these three spins are multiplied with zero weight, i.e. they have the same probability of being up or down. These spins thus favour the eigenstate of a term in the $\hat{H}_x$-part of Eq. (1.14). This shows

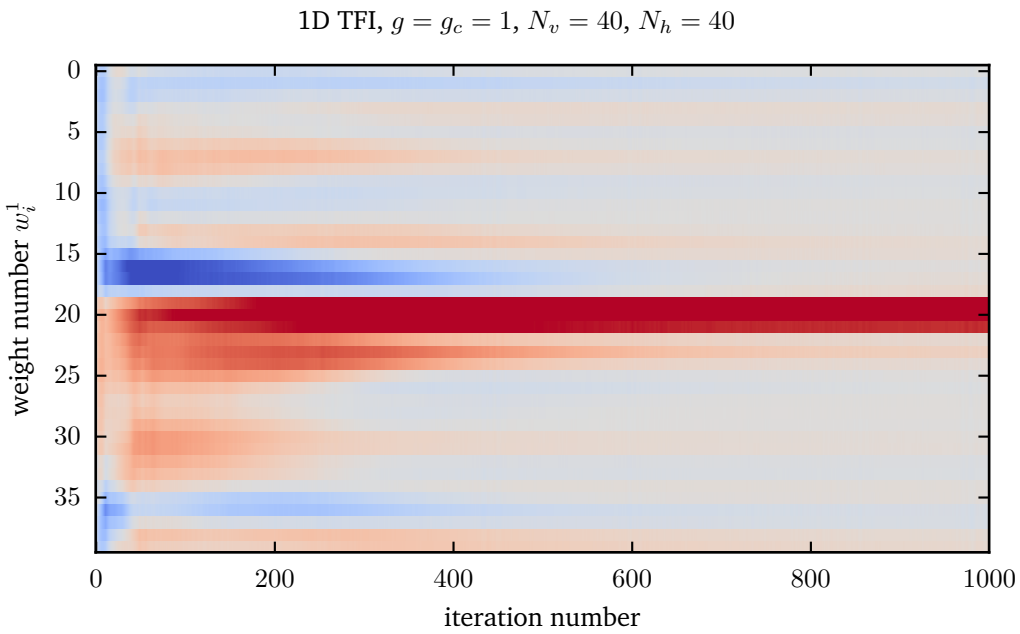

Fig. 4.13: The weights in the weight vector as a function of iteration step for the 1D TFI with $N_v = 40$, $\alpha = 1$ and $g = 1$. A red color indicates a high positive weight and a blue color a low negative weight. The weight vector is periodically translated such that the highest weight in absolute value is centered on weight 20. The weight vector converges to one feature centered around a few neighbouring spins. See figure 4.12 for the energy convergence as a function of the iteration step.

that the representation captures both the short range features (the $\hat{H}_z$-part of the Hamiltonian) and the long-range features (the $\hat{H}_x$-part of the Hamiltonian).

Figure 4.14 ($\alpha = 2$) shows the same qualitative weight vectors in both filters. An additional property of these filters is that they show small, but non-zero, weights in a broad range around the center spin (approximately 20 spins). This is what we would intuitively expect for a system at the critical point ($g_c = 1$), where correlations have a long range. However, it is to be expected that the correlations appear with the same sign, which is not the case in the left panel of figure 4.14. It needs to be pointed out that the presence of more than one filter partially complicates the interpretability of the weight vector, because new factors in the product of cosine hyperbolics of Eq. (4.10) appear. In this case both filters can be used to favour a given spin configuration because they can be independently translated.

Figure 4.15 ($\alpha = 4$) shows the same behaviour as figure 4.14 for two filters (filter 1 and 3), but new behaviour for filters 2 and 4. These weight vectors favour spin configurations where two neighbouring spins have opposite spin. This feature is not readily clear from the separate parts of the Hamiltonian of Eq. (1.14). However, the existence of new features is to be expected because the two parts of the Hamiltonian do not commute. Furthermore, the same caveat holds for the case of two filters: it is

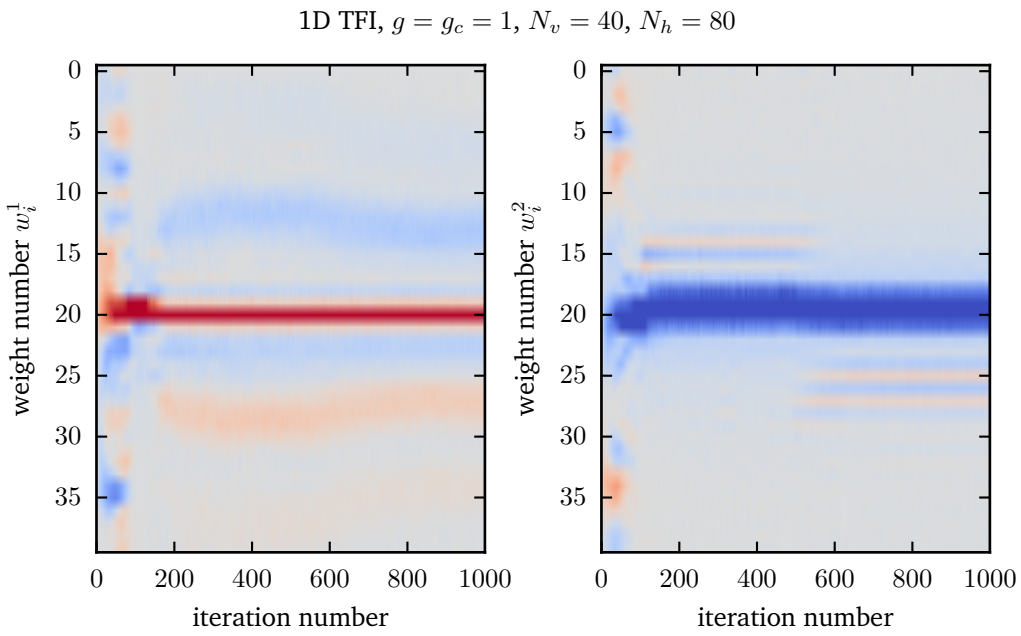

Fig. 4.14: The weights in the weight vectors as a function of iteration step for the 1D TFI with $N_v = 40$, $\alpha = 2$ and $g = 1$. A red color indicates a high positive weight and a blue color a low negative weight. The weight vectors are periodically translated such that the highest weight in absolute value is centered on weight 20. Both filters learn a different feature of the wave function. See figure 4.12 for the energy convergence as a function of the iteration step.

possible to favour a spin configuration by the combination of all filters, which may have other effects than the filters independently.

It is interesting to note from figure 4.12 that the features only become apparent when the energy is already low. Before the relative energy error reaches approximately 1%, the weight vectors are featureless with a fluctuating distribution of weights over the spins. When the energy is low enough, the weight vectors converge to a sharply peaked distribution around a few spins, which monotonously decreases for spins further away from the peak. It is interesting that this change in structure happens very suddenly and when the energy has already significantly converged. Further, remember that the stochastic reconfiguration algorithm is based in imaginary-time evolution, which means that during the optimization, the excited states present in the initial random state are multiplied with a weight c_n relative to the ground state, which is proportional to

$$c_n \propto \exp(-t(E_n - E_{gs})), \tag{4.69}$$

where E_n is the energy of the excited state and t is a measure for the number of iteration steps. It is thus clear that the first excited state is the state which remains present for the largest amount of iteration steps (excluding the ground state) in the variational wave function.

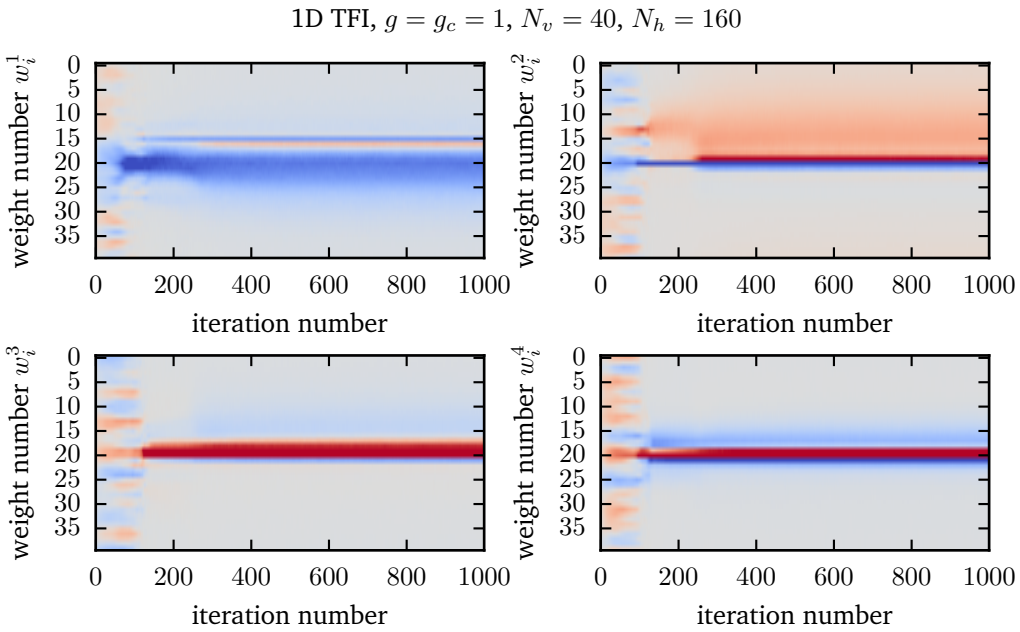

Fig. 4.15: The weights in the weight vectors as a function of iteration step for the 1D TFI with $N_v = 40$, $\alpha = 4$ and $g = 1$. A red color indicates a high positive weight and a blue color a low negative weight. The weight vectors are periodically translated such that the highest weight in absolute value is centered on weight 20. Different filters learn a different feature of the wave function. See figure 4.12 for the energy convergence as a function of the iteration step.

4.5.6 Weight histograms

The weight histograms for different values of $g = h/j$ of the 1D TFI model with $N_v = 40$ are shown in figure 4.16. For every subpanel, we used $\alpha = 4$ and plotted the frequency of occurrence of weights with specific values for 10 separately trained wave functions. The weight histograms give information about how many parameters are needed to represent the wave function. Broad histograms indicate that many parameters are non-vanishing and are reminiscent of highly correlated systems. The plots in figure 4.16 show a broadening of the histogram when going from high to low values of g. This can be appreciated physically. For $g \gg 1$, the ground state will lie close to the eigenstate with smallest eigenvalue of the $\hat{H}_x$-contribution to the Hamiltonian of Eq. (1.14)

$$\hat{H}_x = -h \sum_{i=1}^{N_v} \sigma_x^i. \tag{4.70}$$

Because this part does not contain interactions between spins, the eigenstate of this operator with minimal eigenvalue can be found by taking the direct product of the eigenstates with minimal eigenvalue of every term in Eq. (4.70) separately. Here, we will use the notation $|\uparrow_{x,y,z}\rangle$ ($|\downarrow_{x,y,z}\rangle$) for the eigenstates $\left|s_{x,y,z} = \frac{1}{2}\right\rangle$ ($\left|s_{x,y,z} = -\frac{1}{2}\right\rangle$)

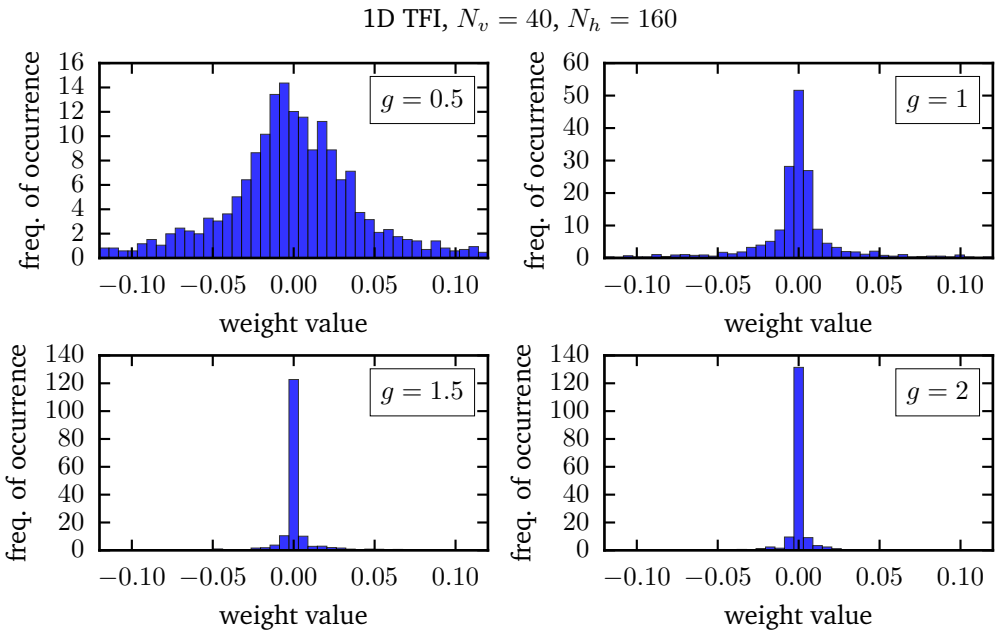

Fig. 4.16: Histogram of the weight values for the 1D TFI model with $N_v = 40$ and $\alpha = 4$ for different values of the coupling g. The histograms are composed of all the weights of 10 independently trained wave functions. Broad histograms indicate that the wave function is more difficult to train because many variational parameters contribute non-trivially to the wave function.

of the operator $\sigma_{x,y,z}$ in order to make the notations lighter. Using Eqs. (1.7) and (1.8), the eigenstate $|\uparrow_x\rangle$ minimizing $-h\sigma_x$ is

$$|\uparrow_x\rangle = |\uparrow_z\rangle + |\downarrow_z\rangle \quad \rightarrow \quad -h\sigma_x |\uparrow_x\rangle = -h |\uparrow_x\rangle . \tag{4.71}$$

The direct product of these states for every spin yields the wave function $|\Psi_x\rangle$ minimizing Eq. (4.70):

$$|\Psi_x\rangle = \sum_{\mathcal{S}} |s_z^1 s_z^2 ... s_z^{N_v}\rangle \quad \rightarrow \quad -h \sum_i \sigma_x^i |\Psi_x\rangle = -hN_v |\Psi_x\rangle . \tag{4.72}$$

This is the state where every $\Psi(\mathcal{S};\mathcal{W})$ is equal to one. It is clear from Eq. (4.10) that this is the state where every weight is zero.

The other extreme situation, $g \ll 1$, corresponds with the eigenstate with smallest eigenvalue of the $\hat{H}_z$-part of the Hamiltonian of Eq. (1.14)

$$\hat{H}_z = -j \sum_{\langle i,k \rangle} \sigma_z^i \sigma_z^k . \tag{4.73}$$

The eigenstate with minimal eigenvalue of one term $-j\sigma_z^1\sigma_z^2$ in the sum is either $\left|\uparrow_z^1\uparrow_z^2\right\rangle$ or $\left|\downarrow_z^1\downarrow_z^2\right\rangle$ as can be seen from

$$-j\sigma_z^1\sigma_z^2\left|\uparrow_z^1\uparrow_z^2\right\rangle = -j\left|\uparrow_z^1\uparrow_z^2\right\rangle \quad \text{and} \quad -j\sigma_z^1\sigma_z^2\left|\downarrow_z^1\downarrow_z^2\right\rangle = -j\left|\downarrow_z^1\downarrow_z^2\right\rangle. \tag{4.74}$$

The eigenstates with minimal eigenvalue of Eq. (4.73) are thus $\left|\uparrow_z^1\uparrow_z^2 \ldots \uparrow_z^{N_v}\right\rangle$ and $\left|\downarrow_z^1\downarrow_z^2 \ldots \downarrow_z^{N_v}\right\rangle$ or a linear combination of them. The eigenstate $|\Psi_z\rangle$ which is invariant under the spin flip operator of Eq. (4.11) is

$$|\Psi_z\rangle = \left|\uparrow_z^1\uparrow_z^2 \ldots \uparrow_z^{N_v}\right\rangle + \left|\downarrow_z^1\downarrow_z^2 \ldots \downarrow_z^{N_v}\right\rangle \quad \rightarrow \quad -j\sum_{\langle i,k\rangle}\sigma_z^i\sigma_z^k|\Psi_z\rangle = -j\frac{N_v(N_v-1)}{2}|\Psi_z\rangle. \tag{4.75}$$

The weights in Eq. (4.10) should now attain values such that $\Psi(\mathcal{S};\mathcal{W})$ is one for the configurations where all the spins are up or all the spins are down in the σ_z-basis and zero for all the other configurations. One solution is that all the weights have the same (high) value. However, multiple (non-trivial) solutions can exist. It is clear however that, in order to enforce that all spin have approximately the same value, all the weights should contribute a non-zero value.

4.5.7 Correlations

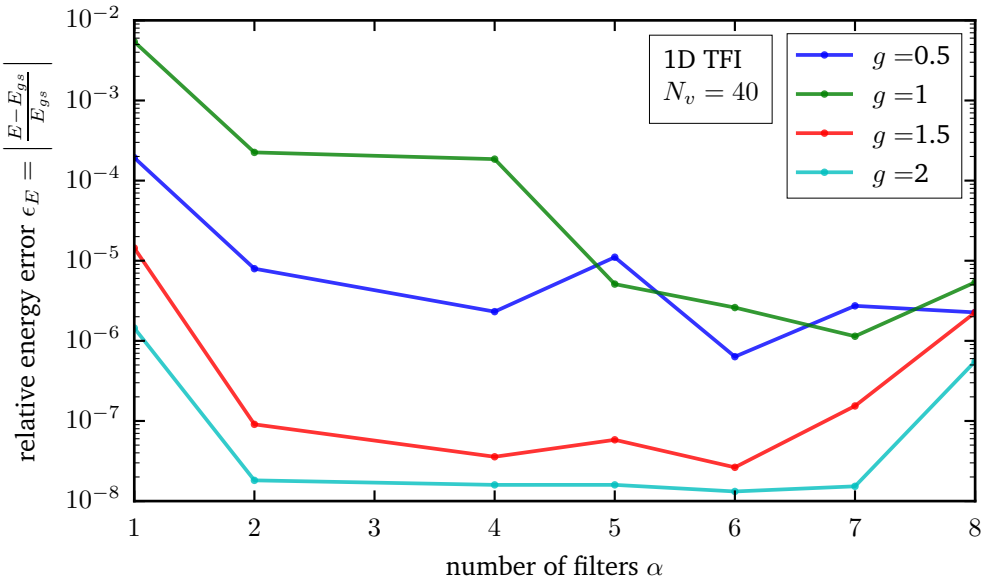

Fig. 4.17: The relative energy errors corresponding to the RBM states in figure 4.18. The model is a 1D TFI with $N_v = 40$. The errors are shown as a function of the number of filters α and the coupling g.

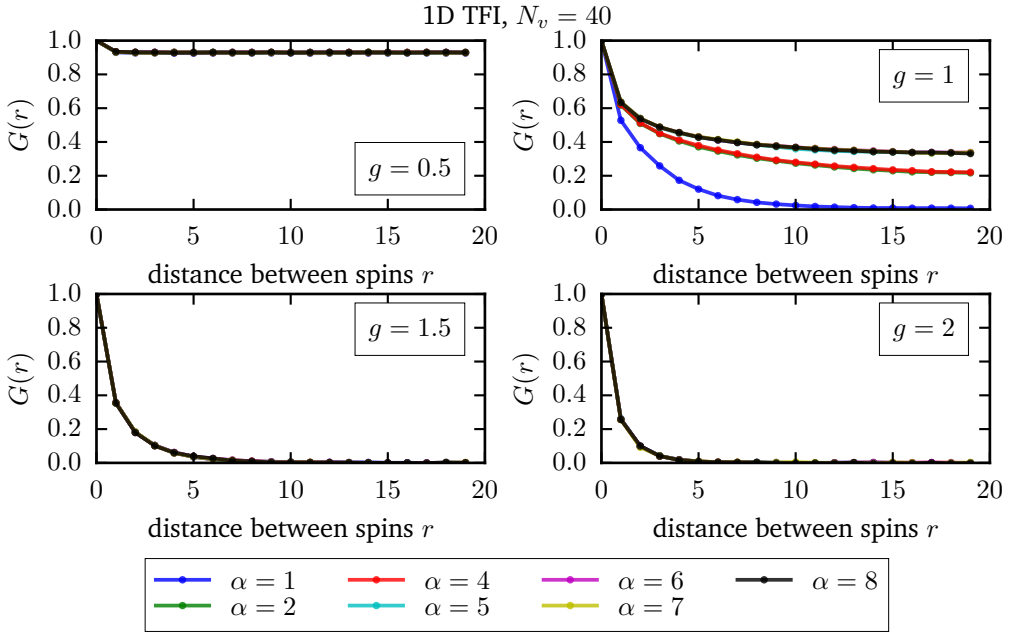

Fig. 4.18: The correlation function as defined in Eq. (4.76) for different values of g and α in a 1D TFI model with $N_v = 40$. This figure shows the long-range order for $g < 1$, the non-trivial long-range correlations for $g = 1$ and the disorder for $g > 1$. States with a large amount of correlations (either long-range order or critical correlations) are more difficult to represent with RBM states. The non-visible lines are due to overlap with other lines. See figure 5.7 for an investigation of the correlation function close to the critical point.

After convergence, the wave functions can be used to calculate expectation values of physical observables. One central observable is the spin-spin correlation function $G(r)$, defined as

$$G(r) = \langle \sigma_z^i \sigma_z^{i+r} \rangle. \tag{4.76}$$

Note that this observable does not depend on the index i due to translation invariance. The correlation function describes the probability that two spins which are r indices apart are aligned. When long-range order is present in the system, the correlation function for $r \to \infty$ converges to

$$G(r) \overset{r\to\infty}{=} \langle \sigma_z^i \rangle \langle \sigma_z^{i+\infty} \rangle = \langle \sigma_z^i \rangle^2, \tag{4.77}$$

where we used the fact that $\langle \sigma_z^i \rangle = \langle \sigma_z^j \rangle$ for every j due to translational invariance. That this relation holds can be appreciated by the fact that spins which are infinitely far apart attain values independently of each other because there is no direct interaction between them. The correlation function can be corrected for this long-range behaviour, yielding the *connected* spin-spin correlation function $C(r)$

$$C(r) = G(r) - \langle \sigma_z^i \rangle^2 = \langle (\sigma_z^i - \langle \sigma_z^i \rangle)(\sigma_z^{i+r} - \langle \sigma_z^i \rangle) \rangle. \tag{4.78}$$

Eq. (4.76) and Eq. (4.78) are both valid measures to measure the correlations present in a system. Eq. (4.76) provides both the non-trivial correlations (arising from direct interactions) and the systematic correlations due to long-range order. Eq. (4.76) only provides the non-trivial part of the correlations. We will argue below that Eq. (4.76) is the most natural way of providing the correlation function in the context of RBM quantum states.

The correlation functions $G(r)$ of Eq. (4.76) for the TFI are displayed in figure 4.18. The corresponding relative energy errors are depicted in figure 4.17 for reference. For $g = 0.5$, $g = 1.5$ and $g = 2$, we see that the correlation function does not change appreciably with increasing α. The energy errors are low for every value of α, compared to $g = 1$. This supports the results of section 4.5.6: ground states with a disordered character are easy to represent with a RBM wave function. For $g = 0.5$, the correlations are for the most part of a long-range character. This is harder to represent than no correlations at all, but the RBM proves to be able to represent them with one filter. For $g = 1$, we see that the correlation function is not well-represented for $\alpha < 5$. This behaviour is logical considering that $g = 1$ is the critical point, where the non-trivial correlations are the largest.

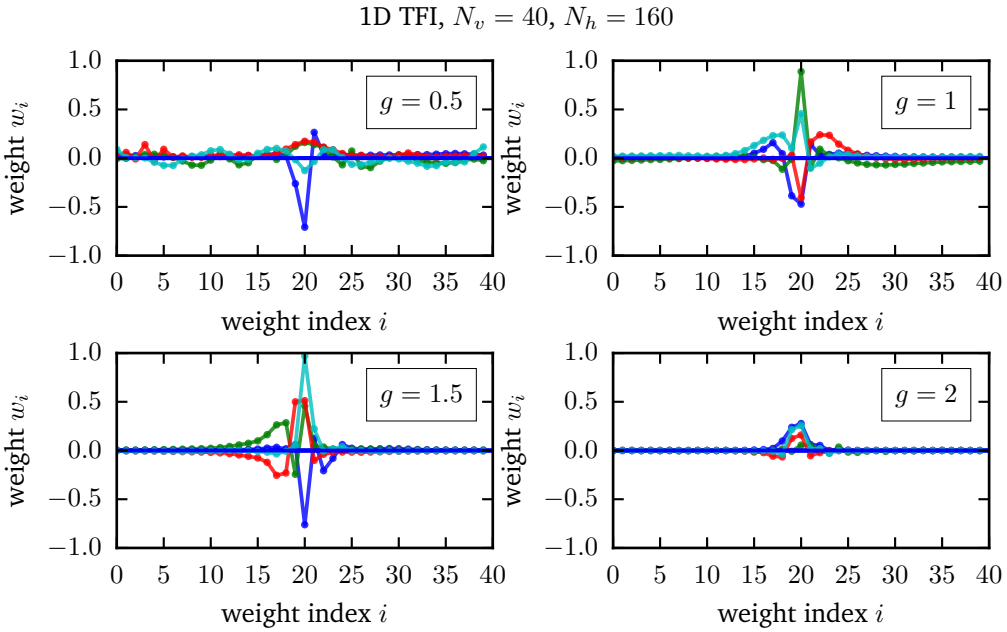

Fig. 4.19: The weight vectors for different values of g for a 1D TFI with $N_v = 40$ and $\alpha = 4$. The weight vectors are centered such that the highest weight is in the middle of the weight vector. A horizontal line at $w_i = 0$ is added for clarity. Note the qualitative relation between the weight vectors and the correlation functions in figure 4.18.

Figure 4.19 shows the four weight vectors of an RBM with $\alpha = 4$ for the same values of g as in figure 4.18. The highest value of a weight vector can be seen as a reference value: because the weight vector is multiplied with the spin configuration via the

inner product, the spin corresponding to the highest weight is enforced in a certain state (up or down), and the other weights determine the behaviour of the other spins with respect to this reference spin. It is expected that for disordered states, the spins are fairly independent, resulting in a weight vector with a short range of non-zero weights around the reference weight. This behaviour is seen in figure 4.19. For the state at $g = 1$ we see that the weights decay to zero with increasing distance from the highest weight, although much slower than when $g > 1$. This reflects the long-range correlations present at criticality. Finally, for $g = 0.5$, we see that the weights decrease for increasing distance from the highest weight, but converge to a non-zero value. This reflects the long-range order in the ground state of systems with $g < 1$: the spins far away from the reference spin should still have a tendency to align with the reference spin.

We thus find that the qualitative behaviour of the weight vector is the same as the behaviour of the spin-spin correlation function defined in Eq. (4.76). Although a qualitative connection between the weight vectors and the correlation function can be made, the weight vectors show behaviour which is more complex than the correlation function. For example, in the case where $g = 0.5$, the weight vectors are globally non-zero, but the weight vectors show slowly varying fluctuations around zero. Another feature are the fluctuations between positive and negative weights around the highest weight. These features arise from the fact that the expansion coefficient of the wave function in Eq. (4.3) is a complex product of cosine hyperbolics, which can lead for example to the cancellation of effects found in the weight vectors.

5 Finite-size scaling for the Transverse Field Ising model

In chapter 4, the quality of the ground-state wave functions was evaluated by means of energy related measures. For instance, the relative energy error of the resolved ground states with respect to the (quasi)-exact ground state energies (section 4.5.1) and the energy fluctuations of the determined states (section 4.5.3) were measured. Energy-related measures have their limitations to fully determine the quality of a resolved ground state, especially near criticality. This can be understood by considering the energy gap Δ_E, defined as the energy difference between the ground and the first excited state

$$\Delta_E = E_1 - E_{gs}, \tag{5.1}$$

where E_1 is the energy of the first excited state and E_{gs} is the energy of the ground state. The energy gap tends to zero for systems approaching the critical point [84]. This makes the use of the energy as a convergence measure questionable. In a finite system with linear dimension L, it can be proven that the energy gap $\Delta_E(L)$ is non-zero, and scales as [88]:

$$\Delta_E(L) \propto L^{-z}, \tag{5.2}$$

with z the dynamical critical exponent. This fact is the motivation to look beyond energy-related measures. Other properties of the ground state give complementary information about the trustworthiness of the numerical results. For this reason, we will study the phase transition in the ground state of the one-dimensional transverse field Ising system. We determine properties of the ground states with the aid of the method described in chapter 4. By studying the finite-size scaling of the phase transition in the TFI model, we can get information about the critical exponents, observables and response functions of the system. This information is relevant as it studies the most correlated wave functions attainable by the TFI model, which are expected to be most difficult to describe.

5.1 The finite-size scaling method

Finite-size scaling originates from statistical physics, where it is used for studying classical phase transitions. It can be directly translated to quantum mechanics, by

the analogy between the nature of the phase transition in classical and quantum systems [13]. In this section, we will introduce the finite-size scaling method.

It is well known that selected observables of many-body physical systems display power law behaviour near the critical point [3, 84]. This behaviour can be captured by the so-called critical exponents of the system: these are the exponents of the power laws describing the observables. We will use the notation of [97], where g stands for the control variable of the model, i.e. the parameter of the system which induces the phase transition. In the TFI model $g = h/j$. Furthermore, g_c is the value of the control variable at which the phase transition occurs. The definitions of the observables and the critical exponents are as follows:

- Correlation length ξ (critical exponent ν):

$$\xi \propto |g - g_c|^{-\nu} \quad \text{(for } g \approx g_c\text{)}. \tag{5.3}$$

  The correlation length ξ defines the length scale of the connected spin-spin correlation function defined in Eq. (4.78):

$$C(r) = \frac{\exp(-r/\xi)}{r^{d-2+\eta}}, \tag{5.4}$$

  with η another critical exponent (often referred to as *anomalous exponent*) and d the dimension of the system.

- Order parameter $\langle|\hat{s}_z|\rangle$ (critical exponent β):

$$\langle|\hat{s}_z|\rangle \propto (g_c - g)^{\beta} \quad \text{(for } g \lesssim g_c\text{)}. \tag{5.5}$$

  the quantity $\langle|\hat{s}_z|\rangle$ is defined in Eq. (4.62).

- Order parameter susceptibility χ_s (critical exponent γ):

$$\chi_s = \left(\frac{\partial\langle|\hat{s}_z|\rangle}{\partial h_{\parallel}}\right)_{h_{\parallel}\to 0} \propto |g - g_c|^{-\gamma} \quad \text{(for } g \approx g_c\text{)}, \tag{5.6}$$

  where $h_{\parallel}$ is an external magnetic field in the z-direction. We refrain from simulating the TFI model with an external magnetic field in the z-direction, i.e. $h_{\parallel} = 0$. However, χ_s can be obtained from the spin-spin correlation function $C(r)$ of Eq. (4.78) at $h_{\parallel} = 0$ by integrating it over all the spin sites (see for example [98]):

$$\chi_s \propto \sum_{i=1}^{N_v} C(i). \tag{5.7}$$

Phase transitions happen in principle only in the thermodynamic limit, i.e. for $N_v \to \infty$. In simulations, one is restricted to systems with a finite size. We thus need a way to connect the observables in finite systems to those in the thermodynamic limit. A convenient way to do this is via the finite-size scaling method. This method assumes that data has been collected for systems in a range of values of the control parameter (including the transition point) and in a range of system sizes. The first step in the finite-size scaling analysis is writing a certain observable near the critical point, for example the order parameter, in terms of the correlation length:

$$\langle|\hat{s}_z|\rangle \propto \xi^{-\beta/\nu}, \tag{5.8}$$

where we make use of Eqs. (5.5) and (5.3). For infinite systems, the correlation length ξ diverges to infinity near criticality. For finite systems however, the correlation length is bounded by the size of the system. This implies that the order parameter cannot diverge polynomially near g_c but will go smoothly to zero when approaching g_c. To describe this behaviour, the order parameter of Eq. (5.8) is expressed as follows:

$$\langle|\hat{s}_z|\rangle \propto \xi^{-\beta/\nu} s_z^0(L/\xi), \tag{5.9}$$

where ξ is the correlation length the system would have in the thermodynamic limit and s_z^0 is a function which is constant when $L \gg \xi$ (such that the scaling relation of Eq. (5.8) holds) and is equal to $(L/\xi)^{-\beta/\nu}$ when $L < \xi$. Eq. (5.9) now explicitly describes how $\langle|\hat{s}_z|\rangle$ scales with system size. However, it depends on the unknown correlation length. To eliminate the correlation length (and bring back the dependence on the deviation from the critical point) from Eq. (5.9), we define the following new scaling function $\tilde{s}_z(x)$

$$\tilde{s}_z(x) = x^\beta s_z^0(x^\nu). \tag{5.10}$$

Using expression (5.10) in Eq. (5.9), we get

$$\langle|\hat{s}_z|\rangle \propto L^{-\beta/\nu} \tilde{s}_z(L^{1/\nu}(g - g_c)), \tag{5.11}$$

which now explicitly depends on the system size L and the control variable g. We can make the following observations. First, when we measure $\langle|\hat{s}_z|\rangle L^{\beta/\nu}$ as a function of g, the curves for the different system sizes all cross at the same value g_c because the argument of $\tilde{s}_z$ is zero. This makes it possible to determine g_c, given the critical exponents are known. Second, the critical exponents (and g_c) can be determined. This can be done by noting that when plotting $\langle|\hat{s}_z|\rangle L^{\beta/\nu}$ versus $L^{1/\nu}(g - g_c)$, the curves for all the different system sizes should coincide. This property is called *data collapse*. Note that this only holds in the vicinity of the critical point, as the scaling analysis is only valid in this region. By varying the critical exponents and the critical transverse field, the appropriate values of the exponents and the critical point are

found when the different curves collapse. The above analysis can be carried out for the other observables in Eqs. (5.3) and Eq. (5.6), yielding similar relationships.

We now wish to define a quantitative measure to evaluate the quality of the data collapse. To perform the data collapse and find the critical exponents and their errors, we use the AutoScale.py routine from [99]. This routine defines a measure Q for the quality of the data collapse as in [100]:

$$Q = \frac{1}{N} \sum_{g,L} \frac{(y(g,L) - Y(g,L))^2}{(dy(g,L))^2 + (dY(g,L))^2}. \tag{5.12}$$

Here, N is the number of (g,L)-pairs in the dataset. Furthermore, $y(g,L)$ is the measured scaled value of the observable for a (g,L)-pair (e.g. in the case of the order parameter $y(g,L) = \langle|\hat{s}_z|\rangle L^{\beta/\nu}$) and $dy(g,L)$ is the measured standard deviation on $y(g,L)$. Lastly, $Y(g,L)$ is the scaling function at (g,L) (e.g. $Y(g,L) = \tilde{s}_z(L^{1/\nu}(g-g_c))$ in the case of the order parameter (see Eq. (5.11))), and $dY(g,L)$ is the standard deviation on it.

Because the scaling function $Y(g,L)$ is unknown, it is estimated from the data points $y(g,L)$ in the following way. Given (g,L), those data points (labeled by (g',L')) are selected for which $L'^{1/\nu}(g'-g_c)$ lies within a predefined range from $L^{1/\nu}(g-g_c)$. With these values, a linear least squares fit is performed. Then, $Y(g,L)$ is the function value of this linear fit at the point labeled by (g,L). The standard deviation $dY(g,L)$ is determined from the error on the fit. When the scaling is performed with the wrong critical exponents, there is no data collapse and the linear fit will be poor. This will result in a high value of the quantity Q. When the data does collapse on a single curve, $Y(g,L)$ will be close to $y(g,L)$ and Q will be small. The correct critical exponents can thus be extracted by minimizing Q. This is done by using the iterative optimization method of Nelder and Mead [101]. Since Q measures the deviation from the curve Y divided by the errors on measurement and curve, it resembles a χ^2 test. This implies that Q should be close to one when the correct critical exponents are used. The error on the critical exponents is found by the $Q+1$ method, which means that the error is defined as the range of values the critical exponents could have such that Q does not deviate more than one unit from the minimal value. This measure is motivated by the resemblance with the χ^2 test, where the error is determined by the surface of χ^2+1-values in the parameter space. This can be brought back to the fact that the deviations in the definition of χ^2 (or in this case Q) are normalized by the error on it.

Of course, the data collapse only holds in a specific range around the critical value of the control variable. In turn, the collapse is only good (as determined by $Q \approx 1$) for this small range of control variables. Because there is no good way to determine

this range, we calculate the critical exponents for different ranges with the above described algorithm, and the range for which the value of Q is lowest is selected to carry out the optimization.

5.2 Finite-size scaling results with restricted Boltzmann machines

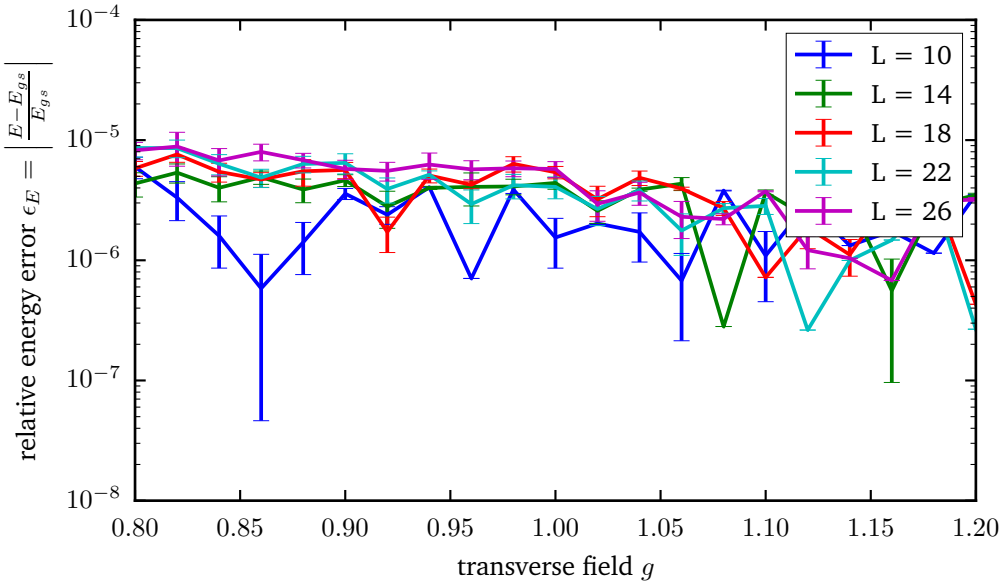

Fig. 5.1: Convergence of the energy of the one-dimensional TFI system in the vicinity of the transition point for different system sizes. The energy is obtained from the state represented by a RBM with $\alpha = 4$ (see Eq. (4.9)). The error bars denote the standard deviation of 15 different runs.

We investigate the phase transition in the one-dimensional transverse field Ising model. We simulate systems of sizes $L \in \{10, 14, 18, 22, 26\}$ for a range of transverse fields $g \in \{0.8, 0.82, ..., 1.18, 1.20\}$. For this, we use a restricted Boltzmann machine (RBM) with $\alpha = 4$ (see Eq. (4.9)). There are two sources of errors on the measured observables. The first source stems from the stochastic importance sampling of the observables (see section 4.2.2). This error can in general be kept low by sampling a large amount of configurations (which is computationally cheap for the system sizes $L \leq 26$ considered). The second source is the imperfect optimization of the wave functions. Due to the stochastic nature of the training process, the quality of the obtained representation will differ between two trained wave functions (for example defined in terms of the relative energy error with respect to exact values of the energy). To alleviate this error, we train 15 wave functions for every (L, g)-combination, and take the average of the measured observables as the observable for given (L, g), with the standard deviation on the mean as the error. The energy

convergence of the different (L, g)-combinations is depicted in figure 5.1. We see that the relative energy error of all the (L, g)-combinations is lower than 10^{-5}, implying that the determined wave function can be expected to be close the ground state.

5.2.1 Magnetization histograms

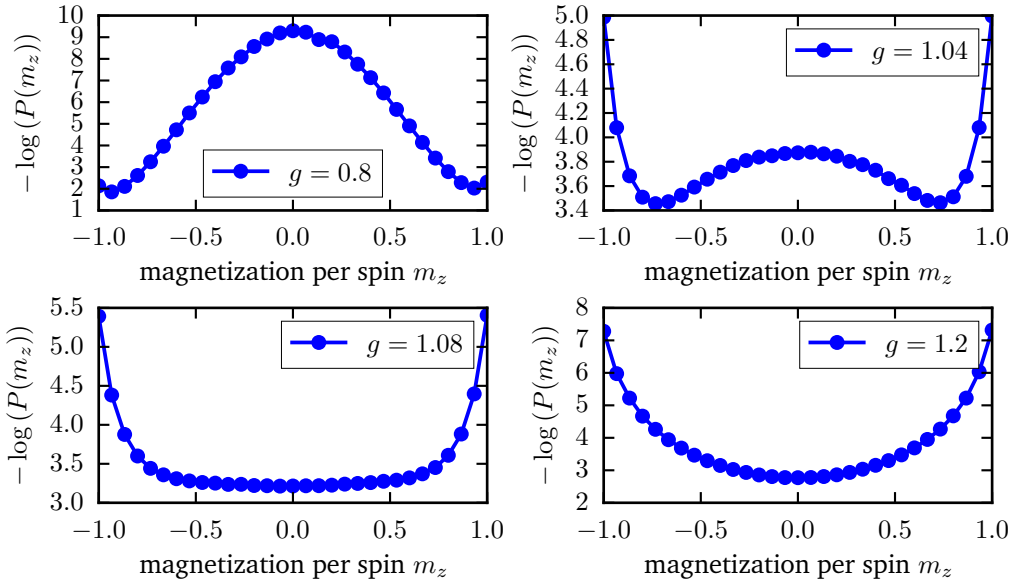

Fig. 5.2: The negative logarithm of the distribution of the magnetization in the 1D TFI model for different transverse field strengths g and $L = 30$. The change from a double well to a single well is clearly visible and resembles the behaviour of the classical two-dimensional Ising model.

As a first measure, we can investigate the symmetry breaking of the phase transition. For brevity, we will denote $\hat{s}_z \equiv \frac{\sum_{i=1}^{N_v} \sigma_z^i}{N_v}$ from now on. We measure the probability of the different possible magnetization values for fixed system size and different transverse fields. The probability is defined as

$$P(m_z) = \sum_{\{\mathcal{S}\}} \delta\left(m_z - \langle \mathcal{S} \,|\, \hat{s}_z \,|\, \mathcal{S} \rangle\right) \frac{\Psi(\mathcal{S}; \mathcal{W})^* \Psi(\mathcal{S}; \mathcal{W})}{\langle \Psi \,|\, \Psi \rangle}, \tag{5.13}$$

where $|\Psi\rangle$ is the state for which the probability distribution is defined. Because the σ_z-basis of Eq. (1.11) is an eigenbasis of the operator $\hat{s}_z$, $\langle \mathcal{S} \,|\, \hat{s}_z \,|\, \mathcal{S} \rangle$ will yield an eigenvalue of $\hat{s}_z$. Thus, m_z is a variable which can take on all possible eigenvalues of $\hat{s}_z$.

To make the connection with the classical two-dimensional Ising model, we calculate the negative logarithm of the probability distribution. This should represent a free energy-like function, as it is defined in the same way as the free energy (as a function of magnetization) is defined in classical statistical physics. The measured values for

system size $L = 30$ for different transverse field strengths g are depicted in figure 5.2. We see the same qualitative behaviour as the classical Ising model. For $h \geq j$, the transverse field part of the Hamiltonian dominates and the ground state in the σ_z-basis converges to a superposition of disordered states (see also section 4.5.6). This corresponds to the disordered states found in the high-temperature range of the classical Ising model, i.e. the supercritical states. The negative logarithm of the magnetization distribution corresponds to the one found in the classical case, i.e. a single well centered around zero. For $h \leq j$, the Ising part of the Hamiltonian dominates and the ground state converges to the ground state of the Ising part $\hat{H}_z$ of the Hamiltonian in Eq. (1.14) (i.e. comparable with the subcritical behaviour of the classical Ising model). The negative logarithm of the magnetization now shows a double well, with the two lowest points at a non-zero magnetization differing from each other only in the sign. At $g \approx 1$, there is a cross-over between the two different profiles. This shows the symmetry breaking in the low transverse field region, also present in the classical Ising model at $T < T_c$. Figure 5.2 also shows the finite-size effects of the magnetization: the magnetization stays non-zero when g is slightly larger than 1. The consequence is that the magnetization converges smoothly to zero for $g \to \infty$ in finite systems. In infinite systems, the magnetization is always zero in the supercritical region ($g > g_c$) (see for example [3]).

5.2.2 Binder cumulant

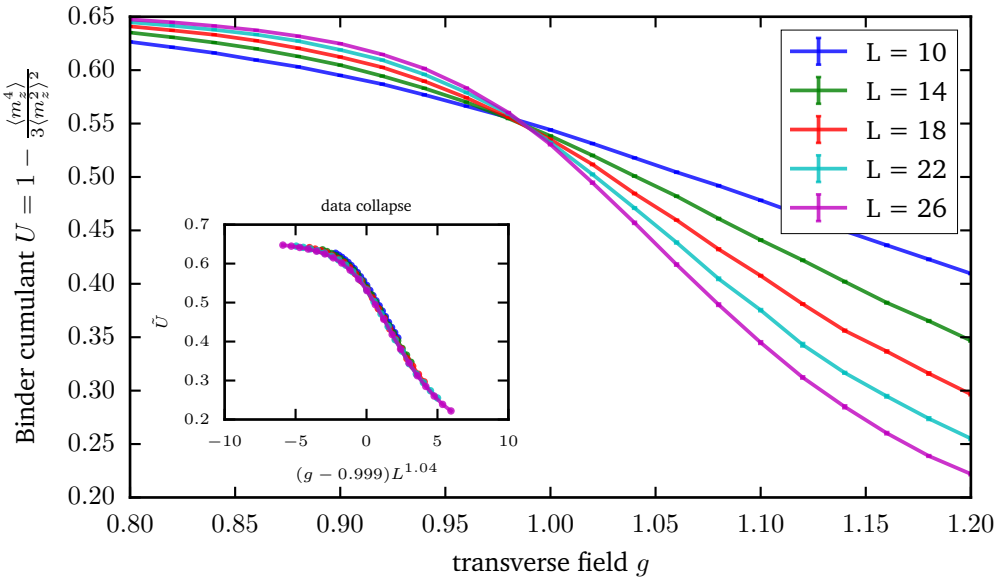

Fig. 5.3: The Binder cumulant for the TFI model. The inset shows the scaled Binder cumulant according to the found critical exponents.

The Binder cumulant U quantifies the deviation of the order parameter probability distribution $P(m_z)$ of Eq. (5.13) from a Gaussian probability distribution [102]. It is

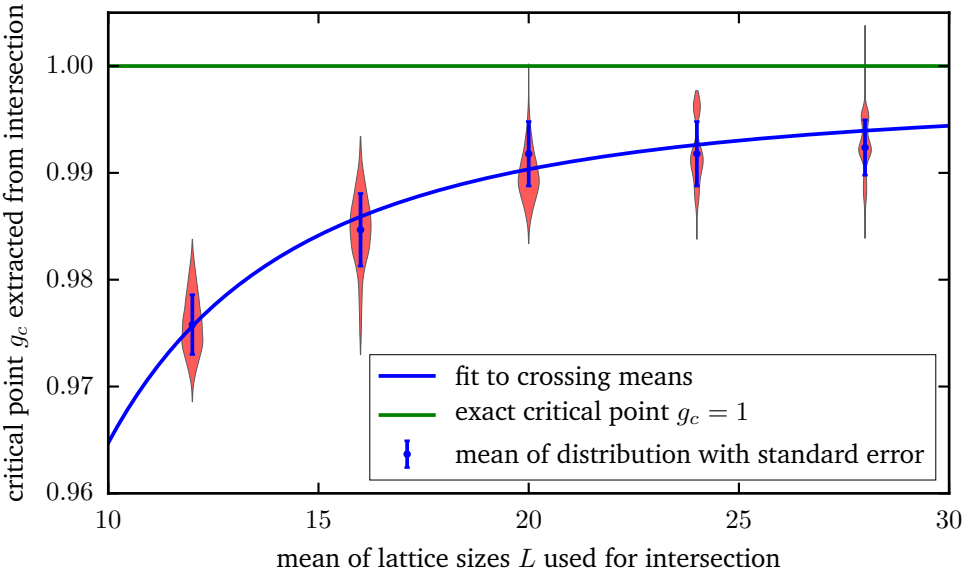

Fig. 5.4: Violin plot of the crossing of the Binder cumulant between two subsequent system sizes. The violins denote the probability distribution of the crossing from our bootstrap sample obtained via kernel density estimation. The x-axis denotes the average of the system sizes used to determine each crossing ($(L_2 - L_1)/2$). The curve is a power law least squares fit to the means of the distributions, resulting in $g_c^{cross} = -6.65L^{-2.31} + 0.997$.

defined as the ratio of the fourth-order cumulant Q_4 and the second-order cumulant Q_2 of the magnetization distribution $P(m_z)$ times $-1/3$:

$$U = -\frac{Q_4}{3Q_2} = 1 - \frac{\langle m_z^4 \rangle}{3\langle m_z^2 \rangle^2} \tag{5.14}$$

The distribution of the magnetization $P(m_z)$ scales with L as:

$$P(m_z, L) = L^y \tilde{P}(m_z L^x, \xi/L), \tag{5.15}$$

for general x, y and with $\tilde{P}(m_z L^x, \xi/L)$ a dimensionless scaling function. The moments $\langle m_z^k \rangle$ of this distribution scale as follows

$$\begin{aligned} \langle m_z^k \rangle &= \int_{-1}^{1} m_z^k P(m_z, L) dm_z \\ &= \int_{-1}^{1} m_z^k L^y \tilde{P}(m_z L^x, \xi/L) dm_z \\ &= L^{x(-k-1)} L^y \int_{-L^x}^{L^x} (m_z L^x)^k \tilde{P}(m_z L^x, \xi/L) d(m_z L^x) \\ &= L^{x(-k-1)} L^y \tilde{F}_k(\xi/L). \end{aligned} \tag{5.16}$$

The function $\tilde{F}_k(\xi/L)$ now only depends on ξ/L because the dependence on m_z is integrated out. The normalization ($k = 0$) ensures that $x = y$ because it cannot depend on L. Using this form for the Binder cumulant, we get the following scaling relation (for $g \approx g_c$) [103]:

$$U = 1 - \frac{L^{-4x}\tilde{F}_4(\xi/L)}{3L^{-4x}\tilde{F}_2(\xi/L)^2} = \tilde{U}((g - g_c)L^{1/\nu}), \tag{5.17}$$

where $\tilde{U}((g - g_c)L^{1/\nu})$ is a new scaling function defined from the scaling functions of the moments of the magnetization distribution. One can see from Eq. (5.17) that the Binder cumulant does not explicitly scale with system size. For $g = g_c$, the Binder cumulant for different system sizes coincides. This makes the Binder cumulant popular for the determination of the critical point g_c as it can be found without any scaling optimization and independently of the critical exponents. The Binder cumulant for different (g, L)-combinations is shown in figure 5.3. We see that the curves cross at a single point, determining the critical point g_c. To find the exact intersections, an interpolation is performed for every curve on the plot. From this interpolation, the crossing points between the curves of two subsequent values of L can be determined. To determine the errors on the intersection, a bootstrap method is performed. For every (g, L)-pair, the different measurements of the Binder cumulant are sampled with replacement to generate a subset of the measurements (recall that we have 15 measurements per (g, L)-pair). From these subsets, the curves are determined and the intersections measured. This procedure is repeated with different subsets of measurements and yields a distribution of the crossing points. The results of this analysis are shown in figure 5.4. This plot shows the distributions of the crossings of the curves of two subsequent system sizes. The distributions are obtained from the bootstrapping procedure estimated via kernel density estimation [104]. To obtain the crossings of the last two points, we performed extra simulations for $L \in \{22, 26, 30\}$ and $g \in \{0.98, 0.985, ..., 1.015\}$ to obtain a denser sampling of points in the vicinity of the critical point, which increases the quality of the interpolations needed to find the crossing points. This is required due to the sensitivity of the crossing point to the interpolation. From the measured distribution, we determine the mean and the standard deviation. From the analysis, we see that the extracted critical points are slightly underestimated, although a systematic convergence towards the real critical point may be possible judging from the plot (this behaviour is typical for the crossing points of the Binder cumulant [105]). A power law function can be fitted to these measurements via the least squares method. We get the following functional form for the crossings g_c^{cross}

$$g_c^{cross}(L) = -6.65L^{-2.31} + 0.997. \tag{5.18}$$

In the thermodynamic limit ($L \to \infty$), we find that the function $g_c^{cross}(L)$ approaches the value $0.997(4)$ where the error is measured from the covariance matrix obtained

with the least squares method. Our values for the crossings are thus consistent with a function which has the expected asymptotic behaviour ($g_c^{cross} \overset{L\to\infty}{=} 1$).

The Binder cumulant can also be used to determine the critical exponent ν (and g_c) with a scaling analysis. Doing this with our data and the method desribed in section 5.1, we get $g_c = 0.999(2)$ and $\nu = 1.04(4)$ with $Q = 1.770$. The Binder cumulant and data collapse as a function of $(g - g_c)L^{1/\nu}$ is shown in figure 5.3.

5.2.3 Order parameter

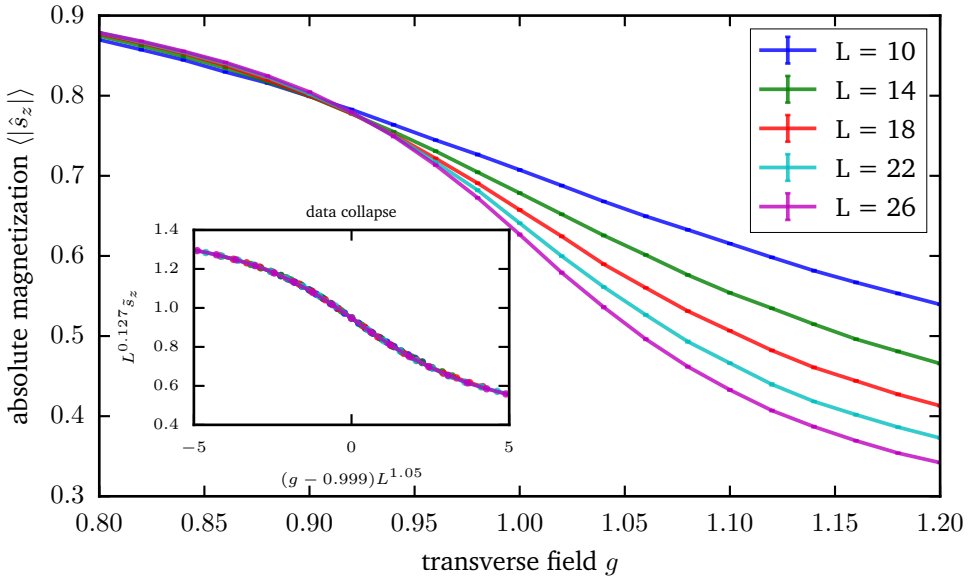

Fig. 5.5: The absolute magnetization $\langle|\hat{s}_z|\rangle$ for the 1D TFI model. The inset shows the scaled absolute magnetization according to the critical exponents found with a scaling analysis. Error bars are shown but are too small to see.

As stated in section 5.1 (see Eq. (5.11)), the order parameter scales as

$$\langle|\hat{s}_z|\rangle \propto L^{-\beta/\nu}\tilde{s}_z(L^{1/\nu}(g - g_c)) \quad (\text{for } g \lesssim g_c). \tag{5.19}$$

The absolute magnetization is depicted in figure 5.5. We see clearly the symmetry breaking between the ordered phase ($g < 1$, high $\langle|\hat{s}_z|\rangle$) and the disordered phase ($g > 1$, low $\langle|\hat{s}_z|\rangle$). We also see the convergence to the expected non-smooth behaviour (i.e. discontinuous first order derivative) near $g = g_c$ with growing system sizes. By using the algorithm described in section 5.1, we can determine the exponents β and ν and the critical transverse field g_c. We observe that qualitatively better results can be obtained when excluding system sizes with $L \leq 10$. This can be attributed to the fact that small systems contain large finite-size effects (other than, of course, the scaling effects). We find $\beta = 0.127(4)$, $\nu = 1.05(5)$ and

$g_c = 0.999(2)$. The quality of the collapse as measured by Q (see Eq. (5.12)) is in this case $Q = 1.133$. The collapse is shown in the inset of figure 5.5.

The fact that the order parameter tends to 1 for $g \to 0$ is additional evidence for the fact that we indeed obtain the ground-state wave function because the magnetization vanishes for the excited states. The *absolute* magnetization never completely vanishes, even not when the magnetization does, but for a linear combination of all excited states it will never raise above 0.5.

5.2.4 Order parameter susceptibility

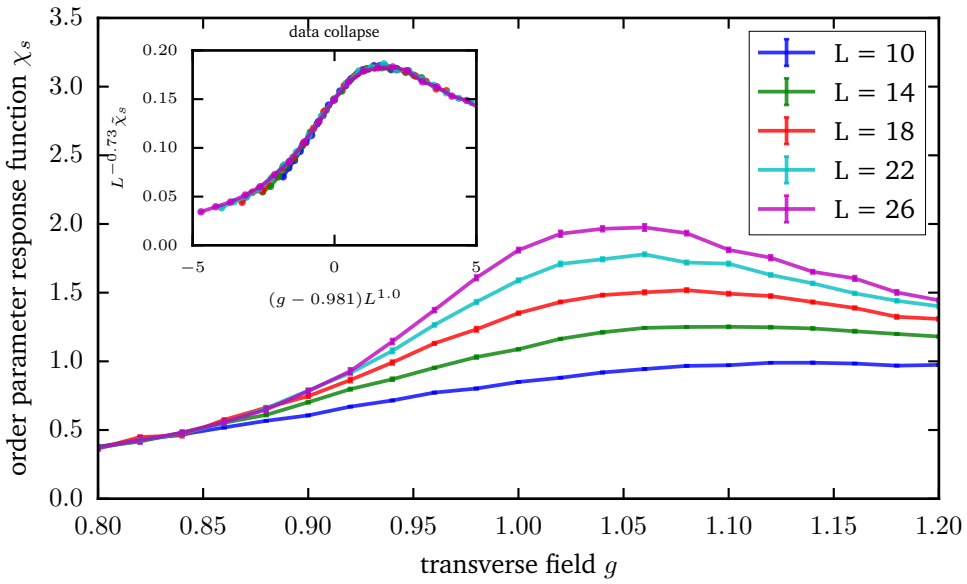

Fig. 5.6: The susceptibility of the order parameter χ_s for the 1D TFI model. The inset shows the scaled order parameter susceptibility according to the extracted critical exponents.

The susceptibility of the order parameter scales as

$$\chi_s = \frac{\partial \langle |\hat{s}_z| \rangle}{\partial h} \propto L^{\gamma/\nu} \tilde{\chi}_s (L^{1/\nu}(g - g_c)) \quad \text{(for } g \approx g_c\text{)}. \tag{5.20}$$

The order parameter susceptibility is displayed as a function of the transverse field g in figure 5.6. Also these curves follow the expected behaviour of a response function. The response is low for values of the control variable far from the critical point. At $g = g_c$, a peak in the response function is observed, reflecting the large amount of fluctuations at this point. The finite-size behaviour is seen in the growth of the peak value of the susceptibility, which should diverge in the thermodynamic limit. The finite size scaling analysis yields $\gamma = 0.73(3)$, $\nu = 1.0(3)$ and $g_c = 0.981(7)$. The quality of the proposed data collapse as determined by Eq. (5.12) is $Q = 0.555$. It is

interesting that the error on ν is quite large compared to the other errors. However, when cross-checking with the value of ν determined in the scaling analyses of the order parameter and the Binder cumulant, all are consistent with each other. The value of g_c, on the other hand, shows a deviation compared with the value obtained in the scaling of the order parameter. However, the values are consistent within three times the error. We stress again that the error estimate as determined by Eq. (5.12) is scheme dependent and might be underestimated.

5.2.5 Integral of correlation function

The correlation function at the critical point is defined as

$$G(r) \propto \frac{1}{r^{d-2+\eta}}, \tag{5.21}$$

as can be seen from Eq. (4.76) with $\xi \to \infty$. The integral of the correlation function should thus scale as

$$\int_0^L G(r)dr \propto L^{2-\eta}. \tag{5.22}$$

Since the order parameter susceptibility is defined as the integral of the correlation function (see Eq. (5.7)), we see that the following relation holds

$$2 - \eta = \gamma/\nu, \tag{5.23}$$

which is one of the scaling relations of the critical exponents [3]. Although Eq. (5.23) determines η from the knowledge of ν and γ, it is still interesting to compute η from Eq. (5.22) as a cross-check. Furthermore, Eq. (5.22) only depends on η. This means that deriving η does not depend on the values of the other critical exponents, as is the case for the scaling analysis described in section 5.1, where two critical exponents and g_c are involved in the optimization. The exponent η is determined by fitting with the least squares method a polynomial function $y = ax^b - c$ to the values of the correlation function integral at the critical point. The exponent in the integral of the correlation function is found to be $2 - \eta = 0.753(4)$, which yields $\eta = 0.247(4)$. We also see that the scaling relation (5.23) is consistent in our analysis.

5.2.6 Comparison with other work

The finite-size scaling analysis of the 1D TFI model has been analysed in the context of exact diagonalization (ED) in Ref. [106] and matrix product states (MPS) in Ref. [107]. In the case of ED, the critical exponents were not explicitly determined via a scaling analysis. Rather, the behaviour of the observables when scaled with the critical exponents of the 2D classical Ising model was investigated in a qualitative

way. However no quantitative results can be compared, their results show that the data collapse for the magnetization (figure 5.5) is similar to the one in this work. Also the magnetization histograms (figure 5.2) share the same behaviour.

Tab. 5.1: Critical exponents of the transverse field Ising model in one dimension found via the finite-size scaling method, and exact critical exponents of the two dimensional classical Ising model

Critical exponent	RBM (this work)	MPS ([107])	2D Ising
g_c / T_c	g_c = 0.999(1)	g_c = 1.000(2)	$T_c = 2.27\frac{J}{k}$
β	0.127(4)	0.125(5)	1/8
γ	0.73(3)	0.75(1)	3/4
η	0.247(4)	0.25(1)	1/4
ν	1.04(4)	1.00(5)	1

The analysis with MPS of Ref. [107] provides quantitative results for the critical exponents. Their results for the critical exponents are summarized and compared to the results of the analysis in this work and of the 2D classical Ising model in table 5.1. We see that the exponents and the accuracy are comparable. Some comments on the analysis of [107] are in place:

- The order parameter susceptibilty is defined as the integral of the unconnected correlation function (see Eq. (4.76)), rather than the integral of the connected correlation function (see Eq. (4.78)). The unconnected and connected correlation functions coincide in the supercritical regime, where the magnetization is zero. However, they do not coincide in the subcritical regime. This explains why the susceptibility is not peaked around $g \approx g_c$ and increases monotonically when going from high values of g to low values of g in [107].

- Their order parameter is defined as the root of the unconnected correlation function $G(r)$ at site $L/2$. This is motivated by the fact that $G(r)$ converges to the square of the magnetization when $r \to \infty$. However, finite systems are investigated, which introduces an error on this measure. In this analysis, the absolute value of the magnetization was used as an order parameter. This measure is however also not free from error as the magnetization is always non-zero, whereas it should be zero when the spin flip symmetry is not broken. It is a known problem [2] in finite systems that the average magnetization always vanishes, even in the symmetry-broken phase (due to transitions between the positive and negative magnetized states). Therefore, other measures need to be used, of which the one in the analysis of [107] and the one in our analysis are both valid examples.

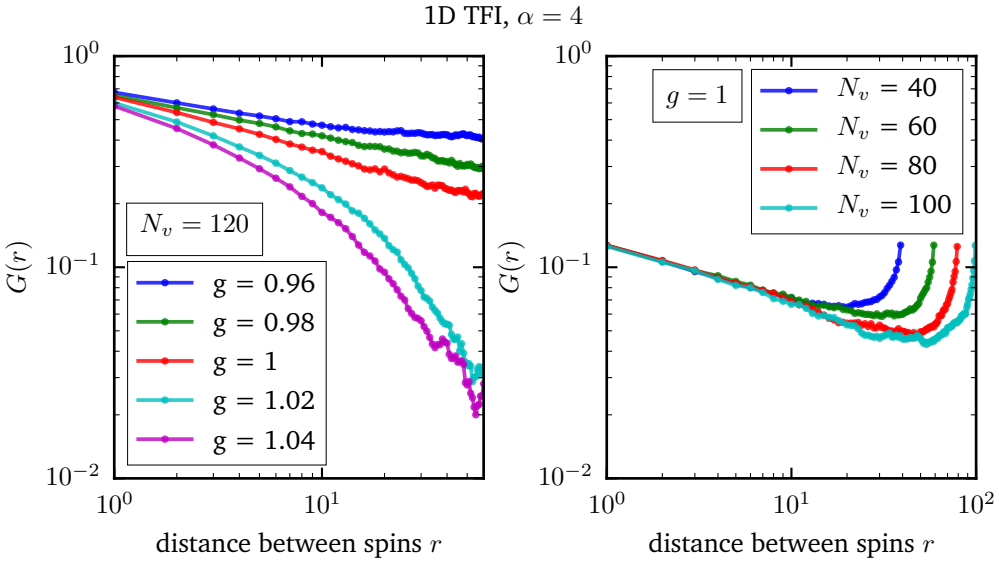

Fig. 5.7: (Unconnected) correlation functions $G(r)$ of the TFI in the vicinity of the critical point determined from RBM wave functions with $\alpha = 4$. Left: the correlation function for $N_v = 120$ around the critical point. Right: the correlation function at the critical point for different system sizes.

Special attention is given to the behaviour of correlations in [107]. More specifically, they show the correlation function for $g \in \{0.96, 0.98, ..., 1.04\}$ and $L = 120$ and for $g = g_c = 1$ and $L \in \{20, 40, ..., 120\}$. For comparison, we plot the same functions obtained with the RBM approach in figure 5.7. The same behaviour as in [107] is observed. We see from this figure the expected behaviour of the correlation function $G(r)$. For $g < 1$ the correlation function levels off due to long-range order for which $G(r)$ is not corrected. For $g > 1$ the correlation function curves downwards, denoting the exponential decrease to zero. At $g = 1$ the correlation function is a straight line for $r < L/2$, denoting the power law behaviour of the correlation function at criticality (see Eq. (5.4)).

Furthermore, [107] calculates the correlation length as defined by the second moment correlation length ξ_2

$$\xi_2 = \frac{1}{q_1}\left(2\frac{S(0)}{S(q_1)} - 2\right)^{1/2}, \tag{5.24}$$

where q_1 is the smallest allowed wave vector (i.e. $q_1 = \frac{2\pi}{L}$) and $S(q)$ is the static structure factor, or the fourier transform of the connected correlation function

$$S(q) = \sum_{r=1}^{N_v} \exp(iqr)C(r). \tag{5.25}$$

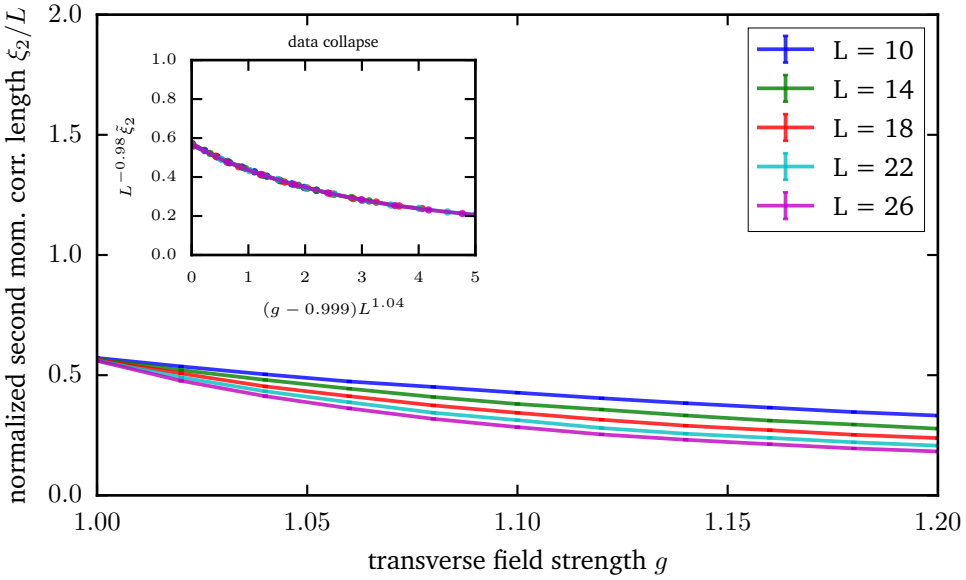

Fig. 5.8: The second moment correlation length ξ_2 of the 1D TFI for different system sizes L and transverse fields g. The inset shows the data collapse according to the measured critical exponents. Error bars are shown but are too small to see.

This definition is proportional to the second moment of the connected correlation function:

$$\xi_2 = \frac{1}{q_1}\left(2\frac{S(0)}{S(q_1)} - 2\right)^{1/2} \propto \frac{\sum_{r=1}^{N_v} r^2 C(r)}{\sum_{r=1}^{N_v} C(r)}. \tag{5.26}$$

Note that Ref. [107] claims that the second moment correlation function is equal to

$$\xi_2 = \frac{1}{q_1}\left(2\frac{S(0)}{S(q_1)} - 2\right)^{1/(2-z-\eta)}, \tag{5.27}$$

where z is the dynamical critical exponent (see introduction to this chapter). However, other literature [88] shows that Eq. (5.24) is the correct form. Note that the second moment correlation length is only well-defined when the correlation function goes to zero when $r \to \infty$ (hence the use of the connected correlation function). Because only the unconnected correlation function is easily accessible in finite systems, we calculate the second moment correlation length only in the supercritical regime (where $G(r)$ coincides with $C(r)$). The second moment correlation function is shown in figure 5.8. We see that the second moment correlation length normalized to the system size grows when approaching $g_c = 1$ and attains the same value, independent of system size, at $g = 1$. Scaling can be performed according to the ansatz

$$\xi_2 = L\tilde{\xi}_2((g - g_c)L^{1/\nu}), \tag{5.28}$$

where $\tilde{\xi}_2$ is a dimensionless scaling function. From this ansatz, the exponent ν and the critical point g_c can be determined. We find from our data $\nu = 1.04(8)$ and $g_c = 0.999(2)$ with the quality $Q = 0.807$ (see Eq. (5.12)).

The fact that the correlation function at $g = 1$ decays polynomially with distance (see figure 5.7), and, equivalently, that the correlation length approaches the system size at $g = 1$ (see figure 5.8) provides the most compelling evidence that the obtained states are indeed the ground state. It can be proven that states which are a linear combination of the ground state and the excited states always have a finite correlation length, depending on the weight of the excited states in the linear combination [84].

5.2.7 Conclusion

The results of our finite-size scaling analysis are summarized in table 5.1. For the critical exponents which are measured more than once, we reported the one with the smallest error. We already saw that these critical exponents are consistent with each other. The transverse field Ising model in one dimension belongs to the same universality class as the two-dimensional classical Ising model (see section 4.4.1) and thus should have the same critical exponents. This is found (within the errorbars) in our analysis. The fact that the correlation function decays polynomially with distance and the fact that the magnetization is non-vanishing provides further evidence that the obtained states are indeed the ground states.

It is clear that the restricted Boltzmann representation of the wave functions of quantum spin systems is able to represent the ground states at quantum critical points. This proves that the correlations induced by the hidden nodes are well suitable to describe the high degree of correlations in quantum systems at a critical point, with a reasonable amount of computational resources.

6 Conclusion and outlook

6.1 Conclusion

This thesis explored how machine learning can be used to solve problems in many-body physics. Chapters 1 and 2 introduced many-body physics and machine learning respectively. Examining the concepts introduced in these two chapters resulted in chapter 3, where we outlined the connection between these two seemingly unrelated fields. Specifically, high-dimensional systems and correlations are features on which these two fields heavily hinge. This provided the motivation to investigate how these fields can benefit from each other. Chapter 3 further provided examples of how concepts from many-body physics can be used in machine learning and how machine learning techniques can be used in many-body physics.

In chapter 4 we investigated in detail how restricted Boltzmann machines (RBM) (introduced in section 2.3.3) can be used as an ansatz for the wave function of strongly-correlated quantum spin systems. For this, we followed closely the work of [57]. After describing the model in the context of quantum spin systems in section 4.1, we describe how to optimize the model to represent the ground state of quantum spin systems in section 4.2. Theoretical results and the relation to other methods used to find the ground state and energy of quantum spin systems are described in section 4.3. The ansatz and the optimization procedure described in sections 4.1 and 4.2 were implemented in a computer code and the results were investigated in section 4.5. The different learning schemes described in section 4.2.2 were compared on different models (the transverse field Ising and antiferromagnetic Heisenberg models, described in section 4.4). It was found that the Adam method [66] and the stochastic reconfiguration method [68] are capable of attaining the lowest energies. However, from the 2D antiferromagnetic Heisenberg model, it was found that the stochastic reconfiguration method surpassed the Adam method in terms of energy minimization. In sections 4.5.2 and 4.5.3 we investigated how the quality of the ground state (measured in terms of relative energy error and relative energy fluctuations) depends on the number of parameters (or hidden nodes) in the RBM ansatz. The quality of the ground state increases with increasing number of parameters, in line with the variational principle and the universal approximation theorem (see section 2.3.3). In section 4.5.2, we also investigated the quality of the ground state with increasing number of degrees of freedom. The RBM is capable of

representing states of systems with an arbitrary number of degrees of freedom, even improving on the relative energy error for larger system sizes. In sections 4.5.5 and 4.5.6, we investigated whether the internal representation (i.e. the weights of the RBM) can shed light on the physics of the ground state we model. For this, we used the 1D transverse field Ising model. It is found that the filters of the RBM represent to some extent what we would expect from the Hamiltonian of which we model the ground state, although some features are present which are not to be expected from the Hamiltonian. The weight histograms of the RBM at different magnitudes of the transverse field shed light on the correlations present in the ground state. It is found that ground states with long-range order have significantly more non-zero weights than ground states which do not exhibit long-range order. Finally, in section 4.5.7 we calculate explicitly the correlation functions for the obtained ground states at different magnitudes of the transverse field and different numbers of filters. It is found that the ground state at criticality is the hardest to represent, which is to be expected from the fact that the correlations become scale-free.

In chapter 5 we investigated the quality of the ground states obtained with the RBM approach in more detail. Instead of only using the energy as a measure for the quality of the ground state, other observables should also be investigated to be able to claim that the RBM is a good representation of the ground state. Especially at criticality this is important due to the fact that the first excited states have an energy close to the ground-state energy. We investigated the quality of the ground state of the 1D TFI near criticality by performing a finite-size scaling analysis to determine the critical exponents β, γ, η and ν (see section 5.1 for their definitions) and the critical point g_c. In section 5.2.2 we calculate the Binder cumulant and determine the critical point g_c from the crossing of the curves of the Binder cumulant as a function of g for different system sizes. The crossings show a systematic increase towards the exact critical point, and are consistent with a power law behaviour which asymptotically converges to $g_c = 1$. Performing a data collapse yields a critical point and critical exponent ν which are consistent (within the errors) with the exact values. Data collapse of the order-parameter (section 5.2.3) and the order-parameter susceptibility (section 5.2.4) yield a critical point and critical exponents β, γ and ν which are also consistent (within the errors) with the exact values. The integral of the correlation function (section 5.2.5) yields the critical exponent η, again consistent with the exact value. The results are compared with results obtained with other methods (exact diagonalization [106] and matrix product states [107]) in section 5.2.6. Both qualitatively and quantitatively comparable results were found.

6.2 Outlook

As for the analysis of this thesis specifically, we can formulate the following improvements and optimizations. First, a more thorough study of the optimal hyperparameters (i.e. the parameters associated with the learning scheme, such as the learning rate) of the optimization schemes can be performed and compared between learning schemes. For example, is there an optimization scheme for which the results only weakly depend on the choice of hyperparameters compared to other schemes? This question is relevant as the true representational power of the RBM can only be accessed via a well-performing optimization procedure. Another question is whether algorithms which do not depend on the gradient of the energy but only on energy values (such as the method of Nelder and Mead [101] or genetic algorithms) work for the determination of ground state-wave functions. If this is the case, the computational cost of the RBM method would be significantly reduced as the determination of the gradients dominate the computational cost of the optimization algorithm (see section 4.2). It was recently shown [108] that the success of gradient-free methods diminishes with the number of parameters to be optimized. This makes the success of the approach questionable for large systems. However, the approach can still be feasible for small systems.

Second, the RBM approach could be extended to include parameters describing long-range order. Section 4.5.6 described how long-range order is encoded in the RBM by setting many weights to a non-zero value. However, long-range order is a systematic effect, for which it is not natural to model with explicit correlations between the spins described by the weights of the RBM. If the systematic effect can be described by a separate mechanism, the weights of the RBM would serve for correlations not described by this systematic effect, and would more naturally reflect the connected correlation function.

Third, the critical exponents which we did not measure in our finite-size scaling analysis could be investigated. These are the exponent of the control variable susceptibility α, the dynamical critical exponent z, and the critical exponent δ of the dependence of the magnetization on the longitudinal magnetic field $h_{\parallel}$. The exponent α could have been readily measured in our analysis, however it requires the determination of a derivative via finite-difference methods, which holds large statistical errors. The determination of the dynamical critical exponent requires time-dependent wave functions and the exponent δ requires the inclusion of a longitudinal magnetic field in the simulations. Both features were not considered, but the information extracted from them can give further insight in the validity of the obtained wave functions.

Promising results were found in chapters 4 and 5. However, due to the novel character of the field, there remain a lot of challenges for the RBM method to solve problems in quantum many-body physics. Some examples are

- To what extent can the internal representation be used to extract information on properties of the system?
- Can RBMs represent all classes of systems faithfully (e.g. topological states)?
- Can the RBM ansatz be used to represent the excited states of Hamiltonians?
- Can the RBM ansatz include different symmetries of Hamiltonians?
- Are there other learning schemes which outperform stochastic reconfiguration?
- How exactly does the RBM approach relate to other methods (in construction and in accuracy)?
- Is there an extension to the RBM approach such that it can describe infinite systems?

A somewhat more general question is whether other machine learning models can represent quantum many-body wave functions and how they compare to the RBM method. Examples are other types of neural networks such as convolutional neural networks or generative adversarial networks (see section 2.3.4) or deep Boltzmann machines (which is the deep variant of Boltzmann machines, i.e. a stack of multiple Boltzmann machines). Preliminary studies have already been caried out [59, 60], but a thorough investigation is needed to establish the optimal model to approximate wave functions of quantum many-body systems.

Nederlandse samenvatting

Systemen met veel vrijheidsgraden zijn een van de meest complexe fysische systemen. Een manier om inzicht te krijgen in deze veeldeeltjessystemen is door de connectie te maken tussen de microscopische vrijheidsgraden en de macroscopische variabelen. Hoe deze connectie formeel gemaakt kan worden, wordt beschreven in de theorie van statistische fysica en de kwantumveeldeeltjestheorie. Hoewel deze theorieën een exacte beschrijving geven van de connectie tussen de microscopie en de macroscopie van een systeem, blijkt het in de praktijk echter vaak nodig om benaderingen en numerieke simulaties uit te voeren. Recente ontwikkelingen brachten een nieuwe manier naar voren die dergelijke benaderingen kan maken: machinaal leren. Technieken uit machinaal leren hebben de eigenschap dat ze de connectie kunnen maken tussen microscopische variabelen (bijvoorbeeld pixels in een foto) en macroscopische concepten (bijvoorbeeld het object dat op een foto staat). Deze eigenschap maakt technieken uit machinaal leren interessant voor toepassingen in veeldeeltjestheorie omdat het net die connectie is die we willen maken in fysische systemen.

In deze thesis gaan we dieper in op het gebruik van machinaal leren voor veeldeeltjessystemen. In hoofdstuk 1 en 2 worden respectievelijk veeldeeltjesfysica en machinaal leren geïntroduceerd. De concepten besproken in deze twee hoofdstukken worden gecombineerd in hoofdstuk 3, waar beschreven wordt hoe machinaal leren gebruikt kan worden in veeldeeltjesfysica, maar ook hoe veeldeeltjesfysica gebruikt kan worden in machinaal leren. In hoofdstuk 4 gaan we dieper in op een van de toepassingen van machinaal leren in veeldeeltjesfysica: het gebruik van restricted Boltzmann machines (RBM) als ansatz voor de golffunctie van sterk gecorreleerde kwantum spin systemen. In eerste instantie volgen we het werk van [57]. De resultaten die we bekomen zijn in lijn met de resultaten van [57]. Vervolgens gaan we verder in op de interne representatie van de RBM en gaan we na of dit strookt met de fysische realiteit. We vinden kwalitatieve verbanden tussen de interne representatie van de RBM en de fysische fenomenen in het systeem, zoals lange dracht orde en korte dracht interacties. In hoofdstuk 5 gaan we dieper in op de kwaliteit van de RBM-representatie van grondtoestanden van kwantum spin systemen. In hoofdstuk 4 werd de energie gebruikt (in de vorm van relatieve fout op exacte waarden of fluctuaties) als maat voor de kwaliteit van de gevonden grondtoestand. Vooral in

de buurt van kritische punten worden deze maten minder betekenisvol omdat het energieverschil tussen de eerste aangeslagen toestand en de grondtoestand klein wordt. Verder is de regio rond het kritisch punt moeilijk te beschrijven door de grote hoeveelheid aan correlaties. Als test voor de kwaliteit van RBM-representaties doen we een finite-size scaling analyse van het eendimensionaal transvers veld Ising model. Hiermee vinden we de theoretisch voorspelde resultaten voor de kritische exponenten β, γ, η en ν en voor het kritisch punt g_c. Dit betekent dat de RBM-representatie wel degelijk de juiste grondtoestanden representeert. We vergelijken onze analyse met de beschikbare analyses in de literatuur en vinden dat onze resultaten consistent zijn met die bekomen aan de hand van andere methodes.

Science popularization

To popularize science, I did a short talk about how machine learning can be used to find the connection between microscopic degrees of freedom and macroscopic properties of many-body systems. This talk was brought by me on a public event, organised by VVN (Vereniging voor Natuurkunde). The aim of the event was to provide a stage for master's students doing scientific studies to present the work they did for their master's thesis to the general public in an accessible way. The talk is in Dutch and is aimed at high school students in their final year. The talk can be found at `https://www.youtube.com/watch?v=hYqYae3JTAE`.

Bibliography

[1] R. J. Baxter, *Exactly solved models in statistical mechanics* (Elsevier, 2016) (cit. on p. 2).

[2] J. Thijssen, *Computational physics* (Cambridge university press, 2007) (cit. on pp. 3, 4, 93).

[3] H. Gould and J. Tobochnik, *Statistical and thermal physics: with computer applications* (Princeton University Press, 2010) (cit. on pp. 3, 60, 82, 87, 92).

[4] A. M. Belaza, K. Hoefman, J. Ryckebusch, et al., "Statistical physics of balance theory", PLoS one **12**, e0183696 (2017) (cit. on p. 3).

[5] T. Vicsek, A. Czirók, E. Ben-Jacob, I. Cohen, and O. Shochet, "Novel type of phase transition in a system of self-driven particles", Physical Review Letters **75**, 1226 (1995) (cit. on p. 3).

[6] Z. Wang, C. T. Bauch, S. Bhattacharyya, et al., "Statistical physics of vaccination", Physics Reports **664**, 1–113 (2016) (cit. on p. 3).

[7] A. Engel, *Statistical mechanics of learning* (Cambridge University Press, 2001) (cit. on p. 3).

[8] B. H. Bransden, *Quantum mechanics* (Pearson Education, 2000) (cit. on pp. 3, 4, 45).

[9] S. Sirca and M. Horvat, *Computational methods for physicists: compendium for students* (Springer Science & Business Media, 2012) (cit. on p. 4).

[10] M. P. Nightingale and C. J. Umrigar, *Quantum Monte Carlo methods in physics and chemistry*, 525 (Springer Science & Business Media, 1998) (cit. on p. 4).

[11] M. A. Nielsen and I. L. Chuang, *Quantum computation and quantum information* (Cambridge university press, 2010) (cit. on p. 8).

[12] H. de Raedt and A. Lagendijk, "Monte Carlo simulation of quantum statistical lattice models", Physics Reports **127**, 233–307 (1985) (cit. on pp. 8, 9).

[13] M. Suzuki, "Relationship between d-Dimensional Quantal Spin Systems and (d+1)-Dimensional Ising SystemsEquivalence, Critical Exponents and Systematic Approximants of the Partition Function and Spin Correlations", Progress of Theoretical Physics **56**, 1454–1469 (1976) (cit. on pp. 9, 11, 82).

[14] M. Suzuki, "Generalized Trotter's formula and systematic approximants of exponential operators and inner derivations with applications to many-body problems", Communications in Mathematical Physics **51**, 183–190 (1976) (cit. on p. 9).

[15] T Mitchell, *Machine Learning* (McGraw Hill, 1997), p. 2 (cit. on p. 13).

[16]S. Lloyd, "Least squares quantization in PCM", IEEE Transactions on Information Theory **28**, 129–137 (1982) (cit. on p. 13).

[17]P. A. Gagniuc, *Markov Chains: From Theory to Implementation and Experimentation* (John Wiley & Sons, 2017) (cit. on p. 13).

[18]D. Silver, J. Schrittwieser, K. Simonyan, et al., "Mastering the game of go without human knowledge", Nature **550**, 354–359 (2017) (cit. on p. 13).

[19]G. Hinton, L. Deng, D. Yu, et al., "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups", IEEE Signal Processing Magazine **29**, 82–97 (2012) (cit. on p. 13).

[20]S. J. Park, B. Bae, J. Kim, and M. Swaminathan, "Application of machine learning for optimization of 3-D integrated circuits and systems", IEEE Transactions on Very Large Scale Integration (VLSI) Systems **25**, 1856–1865 (2017) (cit. on p. 13).

[21]I. Kononenko, "Machine learning for medical diagnosis: history, state of the art and perspective", Artificial Intelligence in medicine **23**, 89–109 (2001) (cit. on p. 13).

[22]R. J. Bolton and D. J. Hand, "Statistical fraud detection: A review", Statistical Science, 235–249 (2002) (cit. on p. 13).

[23]W. Huang, Y. Nakamori, and S.-Y. Wang, "Forecasting stock market movement direction with support vector machine", Computers & Operations Research **32**, 2513–2522 (2005) (cit. on p. 13).

[24]Y. LeCun, C. Cortes, and C. Burges, "MNIST handwritten digit database", AT&T Labs [Online]. Available: http://yann. lecun. com/exdb/mnist (2010) (cit. on p. 15).

[25]I. Goodfellow, Y. Bengio, A. Courville, and Y. Bengio, *Deep learning*, Vol. 1 (MIT press Cambridge, 2016) (cit. on p. 16).

[26]K. Pearson, "LIII. On lines and planes of closest fit to systems of points in space", The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science **2**, 559–572 (1901) (cit. on p. 17).

[27]L. v. d. Maaten and G. Hinton, "Visualizing data using t-SNE", Journal of Machine Learning Research **9**, 2579–2605 (2008) (cit. on p. 18).

[28]C. Cortes and V. Vapnik, "Support-vector networks", Machine Learning **20**, 273–297 (1995) (cit. on p. 18).

[29]M. A. Aizerman, "Theoretical foundations of the potential function method in pattern recognition learning", Automation and Remote Control **25**, 821–837 (1964) (cit. on p. 19).

[30]J. Carrasquilla and R. G. Melko, "Machine learning phases of matter", Nature Physics **13**, 431–434 (2017) (cit. on pp. 21, 35).

[31]K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators", Neural Networks **2**, 359–366 (1989) (cit. on p. 22).

[32]J. J. Hopfield, "Neural networks and physical systems with emergent collective computational abilities", Proceedings of the National Academy of Sciences **79**, 2554–2558 (1982) (cit. on p. 28).

[33]D. O. Hebb, *The organization of behavior: A neuropsychological theory* (Psychology Press, 2005) (cit. on p. 28).

[34]D. H. Ackley, G. E. Hinton, and T. J. Sejnowski, "A learning algorithm for Boltzmann machines", in *Readings in computer vision* (Elsevier, 1987), pp. 522–533 (cit. on p. 28).

[35]L. P. Kadanoff, "Notes on Migdal's recursion formulas", Annals of Physics **100**, 359–394 (1976) (cit. on p. 29).

[36]P. Mehta and D. J. Schwab, "An exact mapping between the variational renormalization group and deep learning", arXiv preprint arXiv:1410.3831 (2014) (cit. on p. 29).

[37]H. W. Lin, M. Tegmark, and D. Rolnick, "Why does deep and cheap learning work so well?", Journal of Statistical Physics **168**, 1223–1247 (2017) (cit. on p. 30).

[38]D. Liu, S.-J. Ran, P. Wittek, et al., "Machine learning by two-dimensional hierarchical tensor networks: A quantum information theoretic perspective on deep architectures", arXiv preprint arXiv:1710.04833 (2017) (cit. on p. 30).

[39]Y. Levine, D. Yakira, N. Cohen, and A. Shashua, "Deep Learning and Quantum Entanglement: Fundamental Connections with Implications to Network Design.", arXiv preprint arXiv:1704.01552 (2017) (cit. on p. 31).

[40]P. Baldi, P. Sadowski, and D. Whiteson, "Searching for exotic particles in high-energy physics with deep learning", Nature Communications **5** (2014) (cit. on p. 31).

[41]J. VanderPlas, A. J. Connolly, Ž. Ivezić, and A. Gray, "Introduction to astroML: Machine learning for astrophysics", in Intelligent data understanding (cidu), 2012 conference on (IEEE, 2012), pp. 47–54 (cit. on p. 31).

[42]Y. Fujimoto, K. Fukushima, and K. Murase, "Methodology study of machine learning for the neutron star equation of state", arXiv preprint arXiv:1711.06748 (2017) (cit. on p. 32).

[43]R. Utama, W.-C. Chen, and J. Piekarewicz, "Nuclear charge radii: density functional theory meets Bayesian neural networks", Journal of Physics G: Nuclear and Particle Physics **43**, 114002 (2016) (cit. on p. 32).

[44]L.-G. Pang, K. Zhou, N. Su, et al., "An equation-of-state-meter of quantum chromodynamics transition from deep learning", Nature Communications **9**, 210 (2018) (cit. on p. 32).

[45]R. H. Swendsen and J.-S. Wang, "Nonuniversal critical dynamics in Monte Carlo simulations", Physical Review Letters **58**, 86–88 (1987) (cit. on p. 33).

[46]J. Liu, Y. Qi, Z. Y. Meng, and L. Fu, "Self-learning Monte Carlo method", Physical Review B **95**, 041101 (2017) (cit. on p. 33).

[47]A. Morningstar and R. G. Melko, "Deep learning the Ising model near criticality", arXiv preprint arXiv:1708.04622 (2017) (cit. on p. 34).

[48]Z. Liu, S. P. Rodrigues, and W. Cai, "Simulating the Ising Model with a Deep Convolutional Generative Adversarial Network", arXiv preprint arXiv:1710.04987 (2017) (cit. on p. 34).

[49]P. Broecker, J. Carrasquilla, R. G. Melko, and S. Trebst, "Machine learning quantum phases of matter beyond the fermion sign problem", Scientific Reports **7**, 8823 (2017) (cit. on p. 34).

[50]L. Wang, "Discovering phase transitions with unsupervised learning", Physical Review B **94**, 195105 (2016) (cit. on p. 35).

[51]K. Ch'ng, J. Carrasquilla, R. G. Melko, and E. Khatami, "Machine Learning Phases of Strongly Correlated Fermions", Physical Review X **7**, 031038 (2017) (cit. on p. 35).

[52]E. P. van Nieuwenburg, Y.-H. Liu, and S. D. Huber, "Learning phase transitions by confusion", Nature Physics **13**, 435–439 (2017) (cit. on p. 36).

[53]M. J. S. Beach, A. Golubeva, and R. G. Melko, "Machine learning vortices at the Kosterlitz-Thouless transition", Physical Review B **97**, 045207 (2018) (cit. on p. 36).

[54]A. Shrikumar, P. Greenside, and A. Kundaje, "Learning Important Features Through Propagating Activation Differences", in International conference on machine learning (2017), pp. 3145–3153 (cit. on p. 36).

[55]J. T. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller, "Striving for simplicity: The all convolutional net", arXiv preprint arXiv:1412.6806 (2014) (cit. on p. 36).

[56]P. Ponte and R. G. Melko, "Kernel methods for interpretable machine learning of order parameters", Physical Review B **96**, 205146 (2017) (cit. on p. 36).

[57]G. Carleo and M. Troyer, "Solving the quantum many-body problem with artificial neural networks", Science **355**, 602–606 (2017) (cit. on pp. 36, 37, 39, 62, 69, 97, 101).

[58]Y. Nomura, A. S. Darmawan, Y. Yamaji, and M. Imada, "Restricted Boltzmann machine learning for solving strongly correlated quantum systems", Physical Review B **96**, 205152 (2017) (cit. on pp. 37, 69).

[59]H. Saito, "Solving the Bose–Hubbard model with machine learning", Journal of the Physical Society of Japan **86**, 093001 (2017) (cit. on pp. 37, 100).

[60]H. Saito and M. Kato, "Machine learning technique to find quantum many-body ground states of bosons on a lattice", Journal of the Physical Society of Japan **87**, 014001 (2017) (cit. on pp. 37, 100).

[61]I. Glasser, N. Pancotti, M. August, I. D. Rodriguez, and J. I. Cirac, "Neural-Network Quantum States, String-Bond States, and Chiral Topological States", Physical Review X **8**, 011006 (2018) (cit. on pp. 37, 58, 59).

[62]X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks", in Proceedings of the thirteenth international conference on artificial intelligence and statistics (2010), pp. 249–256 (cit. on pp. 44, 67).

[63]W. K. Hastings, "Monte Carlo sampling methods using Markov chains and their applications", Biometrika **57**, 97–109 (1970) (cit. on p. 46).

[64]J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization", Journal of Machine Learning Research **12**, 2121–2159 (2011) (cit. on p. 47).

[65]M. D. Zeiler, "ADADELTA: an adaptive learning rate method", arXiv preprint arXiv:1212.5701 (2012) (cit. on p. 48).

[66]D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization", arXiv preprint arXiv:1412.6980 (2014) (cit. on pp. 48, 97).

[67]G. Goh, "Why Momentum Really Works", Distill (2017) `10.23915/distill.00006` (cit. on pp. 48, 65).

[68]S. Sorella, "Green Function Monte Carlo with Stochastic Reconfiguration", Physical Review Letters **80**, 4558–4561 (1998) (cit. on pp. 48, 97).

[69]S.-C. T. Choi and M. A. Saunders, "Algorithm 937: MINRES-QLP for symmetric and hermitian linear equations and least-squares problems", ACM Transactions on Mathematical Software (TOMS) **40**, 16 (2014) (cit. on p. 51).

[70]M. M. Wolf, F. Verstraete, M. B. Hastings, and J. I. Cirac, "Area laws in quantum systems: mutual information and correlations", Physical Review Letters **100**, 070502 (2008) (cit. on p. 54).

[71]M. B. Hastings, "An area law for one-dimensional quantum systems", Journal of Statistical Mechanics: Theory and Experiment **2007**, P08024 (2007) (cit. on p. 54).

[72]D.-L. Deng, "Machine Learning Bell Nonlocality in Quantum Many-body Systems", arXiv:1710.04226 (2017) (cit. on p. 55).

[73]D.-L. Deng, X. Li, and S. Das Sarma, "Machine learning topological states", Physical Review B **96**, 195145 (2017) (cit. on p. 55).

[74]J. Chen, S. Cheng, H. Xie, L. Wang, and T. Xiang, "Equivalence of restricted Boltzmann machines and tensor network states", Physical Review B **97**, 085104 (2018) (cit. on pp. 55, 58).

[75]H. Lin, J. Gubernatis, H. Gould, and J. Tobochnik, "Exact diagonalization methods for quantum systems", Computers in Physics **7**, 400–407 (1993) (cit. on p. 56).

[76]E. S. Coakley and V. Rokhlin, "A fast divide-and-conquer algorithm for computing the spectra of real symmetric tridiagonal matrices", Applied and Computational Harmonic Analysis **34**, 379–414 (2013) (cit. on p. 56).

[77]S. R. White, "Density matrix formulation for quantum renormalization groups", Physical Review Letters **69**, 2863–2866 (1992) (cit. on p. 56).

[78]U. Schollwöck, "The density-matrix renormalization group in the age of matrix product states", Annals of Physics **326**, 96–192 (2011) (cit. on p. 56).

[79]S. Rommer and S. Ostlund, "A class of ansatz wave functions for 1D spin systems and their relation to DMRG", Physical Review B **55**, 2164 (1997) (cit. on p. 57).

[80]J. Eisert, "Entanglement and tensor network states", arXiv preprint arXiv:1308.3318 (2013) (cit. on p. 58).

[81]N. Schuch, M. M. Wolf, F. Verstraete, and J. I. Cirac, "Simulation of Quantum Many-Body Systems with Strings of Operators and Monte Carlo Tensor Contractions", Physical Review Letters **100**, 040501 (2008) (cit. on p. 58).

[82]R. Orús, "A practical introduction to tensor networks: Matrix product states and projected entangled pair states", Annals of Physics **349**, 117–158 (2014) (cit. on p. 58).

[83]P. Pfeuty, "The one-dimensional Ising model with a transverse field", Annals of Physics **57**, 79–90 (1970) (cit. on p. 59).

[84]S. Sachdev, *Quantum phase transitions* (Wiley Online Library, 2007) (cit. on pp. 60, 81, 82, 96).

[85]A. Dutta, G. Aeppli, B. K. Chakrabarti, et al., "Quantum phase transitions in transverse field spin models: From statistical physics to quantum information", arXiv preprint arXiv:1012.0653 (2010) (cit. on p. 60).

[86]N. D. Mermin and H. Wagner, "Absence of ferromagnetism or antiferromagnetism in one-or two-dimensional isotropic Heisenberg models", Physical Review Letters **17**, 1133 (1966) (cit. on p. 62).

[87]E. Manousakis, "The spin-½ Heisenberg antiferromagnet on a square lattice and its application to the cuprous oxides", Reviews of Modern Physics **63**, 1–62 (1991) (cit. on p. 62).

[88]A. W. Sandvik, "Computational studies of quantum spin systems", in Aip conference proceedings, Vol. 1297, 1 (AIP, 2010), pp. 135–338 (cit. on pp. 62, 81, 95).

[89]L. Bottou, "Large-scale machine learning with stochastic gradient descent", in *Proceedings of compstat'2010* (Springer, 2010), pp. 177–186 (cit. on p. 62).

[90]B. Bauer, L. Carr, H. G. Evertz, et al., "The ALPS project release 2.0: open source software for strongly correlated systems", Journal of Statistical Mechanics: Theory and Experiment **2011**, P05001 (2011) (cit. on p. 64).

[91]C. Hamer, "Finite-size scaling in the transverse Ising model on a square lattice", Journal of Physics A: Mathematical and General **33**, 6683 (2000) (cit. on pp. 64, 65, 68, 70).

[92]A. W. Sandvik, "Finite-size scaling of the ground-state parameters of the two-dimensional Heisenberg model", Physical Review B **56**, 11678–11690 (1997) (cit. on pp. 64, 66, 68, 70).

[93]S. Al-Assam, S. R. Clark, C. J. Foot, and D. Jaksch, "Capturing long range correlations in two-dimensional quantum lattice systems using correlator product states", Physical Review B **84**, 205108 (2011) (cit. on p. 68).

[94]L. Tagliacozzo, G. Evenbly, and G. Vidal, "Simulation of two-dimensional quantum systems using a tree tensor network that exploits the entropic area law", Physical Review B **80**, 235127 (2009) (cit. on p. 68).

[95]F Mezzacapo, N Schuch, M Boninsegni, and J. I. Cirac, "Ground-state properties of quantum many-body systems: entangled-plaquette states and variational Monte Carlo", New Journal of Physics **11**, 083026 (2009) (cit. on p. 69).

[96]M. Lubasch, J. I. Cirac, and M.-C. Bañuls, "Algorithms for finite projected entangled pair states", Physical Review B **90**, 064425 (2014) (cit. on p. 69).

[97]T. R. Kirkpatrick and D. Belitz, "Exponent relations at quantum phase transitions with applications to metallic quantum ferromagnets", Physical Review B **91**, 214407 (2015) (cit. on p. 82).

[98]M. E. Fisher, "The theory of equilibrium critical phenomena", Reports on Progress in Physics **30**, 615 (1967) (cit. on p. 82).

[99]O. Melchert, "Autoscale.py - A program for automatic finite-size scaling analyses: A user's guide", arXiv preprint arXiv:0910.5403 (2009) (cit. on p. 84).

[100]J. Houdayer and A. K. Hartmann, "Low-temperature behavior of two-dimensional Gaussian Ising spin glasses", Physical Review B **70**, 014418 (2004) (cit. on p. 84).

[101]J. A. Nelder and R. Mead, "A simplex method for function minimization", The Computer Journal **7**, 308–313 (1965) (cit. on pp. 84, 99).

[102]K. Binder, "Critical Properties from Monte Carlo Coarse Graining and Renormalization", Physical Review Letters **47**, 693–696 (1981) (cit. on p. 87).

[103]K. Anagnostopoulos, *Computational Physics-A Practical Introduction to Computational Physics and Scientific Computing (using C++)*, Vol. 2 (National Technical University of Athens, 2016) (cit. on p. 89).

[104]M. Rosenblatt, "Remarks on Some Nonparametric Estimates of a Density Function", Annals of Mathematical Statistics **27**, 832–837 (1956) (cit. on p. 89).

[105]C. G. West, A. Garcia-Saez, and T.-C. Wei, "Efficient evaluation of high-order moments and cumulants in tensor network states", Physical Review B **92**, 115103 (2015) (cit. on p. 89).

[106]J. Um, S.-I. Lee, and B. J. Kim, "Quantum Phase Transition and Finite-Size Scaling of the One-Dimensional Ising Model", Journal of the Korean Physical Society **50**, 285–289 (2007) (cit. on pp. 92, 98).

[107]S.-B. Park and M.-C. Cha, "Matrix product state approach to the finite-size scaling properties of the one-dimensional critical quantum Ising model", Journal of the Korean Physical Society **67**, 1619–1623 (2015) (cit. on pp. 92–95, 98).

[108]L. M. Rios and N. V. Sahinidis, "Derivative-free optimization: a review of algorithms and comparison of software implementations", Journal of Global Optimization **56**, 1247–1293 (2013) (cit. on p. 99).

List of Symbols

α	Number of hidden nodes divided by number of visible nodes of RBM
β	Critical exponent of order parameter
χ_s	Susceptibility of the order parameter
ΔE	Fluctuations on the energy for a given state
Δ_E	Energy gap of a quantum system (difference between first excited state and ground state)
ϵ_E	Relative energy error
η	Anomalous critical exponent
γ	Critical exponent of the order parameter susceptibility
$\hat{\rho}$	Density operator
$\hat{H}$	Hamiltonian of a quantum mechanical system
$\hat{I}$	Identity operator
$\hat{R}(\boldsymbol{\theta})$	Rotation operator in spin space by angle θ
$\hat{s}_{x,y,z}$	Spin-projection operator on x, y, z-axis
$\|\Psi\rangle$	State of quantum mechanical system in Hilbert space
$\hat{\mathbf{S}}$	Total spin of a many-body spin system
$\mathcal{H}$	Hilbert space associated with a quantum mechanical system
$\mathcal{O}_w$	Partial derivative of $\Psi(\mathcal{S};\mathcal{W})$ to w divided by $\Psi(\mathcal{S};\mathcal{W})$
$\mathcal{S}$	Configuration (or microstate) of a collection of degrees of freedom
$\mathcal{W}$	Set of parameters of a machine learning model
ν	Critical exponent of the correlation length
$\sigma_{x,y,z}$	Pauli matrix (or operator)
$\tilde{o}$	Vector consisting of $\mathcal{O}_{w_i} - \langle\mathcal{O}_{w_i}\rangle$
ξ	Correlation length of many-body system
ξ_2	Second moment correlation function
a_i	Bias of the i-th visible unit in an RBM
b	Bias of linear transformation between two layers in neural network
b_i	Bias of the i-th hidden unit in an RBM
C	Cost function

$C(r)$	Connected spin-spin correlation function
D	Dimension of a Hilbert space
d	Spatial dimension of a system
E	Energy
F	Free energy
$G(r)$	Unconnected spin-spin correlation function
g_w^t	Partial derivative of cost function to weight w at iteration step t
h	magnitude of an external transverse field
$h_\parallel$	Longitudinal external magnetic field
j	Magnitude of nearest-neighbour interactions
k_B	Boltzmann constant
L	Size of system in context of finite size scaling
l	Learning rate of machine learning optimizer
N	Number of degrees of freedom
n	Dimension of a vector (e.g. input of neural network)
N_h	Number of hidden units of an RBM
N_v	Number of visible units (input dimensions) in an RBM. When modeling wave functions with RBMs, the number of degrees of freedom in a physical system N will be changed to N_v to distinguish from N_h
Q	Quality of data collapse in the context of finite-size scaling
S	Entropy
$S(q)$	Static structure factor
s^i	Spin degree of freedom on site i
S_e	Entanglement entropy
$s^i_{x,y,z}$	Spin projection of spin on site i on x, y, z-axis
T	Temperature
t	Linear transformation between two layers in neural network
U	Binder cumulant
$w_{ij}^{(k)}$	Weight connecting node i in layer k with node j in layer $k+1$ in a neural network. If there is only one layer, the superscript is suppressed.
Z	Canonical partition function

List of Figures

List of Tables

Colophon

This thesis was typeset with LaTeX 2$_\varepsilon$. It uses the *Clean Thesis* style developed by Ricardo Langner. The design of the *Clean Thesis* style is inspired by user guide documents from Apple Inc.

Download the *Clean Thesis* style at `http://cleanthesis.der-ric.de/`.