Augmenting Quantum Mechanics with Artificial Intelligence

by

Giacomo Torlai

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Doctor of Philosophy
in
Physics

Waterloo, Ontario, Canada, 2018

Examining Committee Membership

The following served on the Examining Committee for this thesis. The decision of the Examining Committee is by majority vote.

External Examiner:	Norbert Schuch Professor, Max Planck Institute for Quantum Optics, Garching
Supervisor:	Roger G. Melko Professor, Dept. of Physics, University of Waterloo
Internal Member:	Anton Burkov Professor, Dept. of Physics, University of Waterloo
Internal Member:	Matteo Mariantoni Professor, Institute of Quantum Computing, Waterloo
Internal-External Member:	Pierre-Nicholas Roy Professor, Dept. of Chemistry, University of Waterloo

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

The simulation of quantum matter with classical hardware plays a central role in the discovery and development of quantum many-body systems, with far-reaching implications in condensed matter physics and quantum technologies. In general, efficient and sophisticated algorithms are required to overcome the severe challenge posed by the exponential scaling of the Hilbert space of quantum systems. In contrast, hardware built with quantum bits of information are inherently capable of efficiently finding solutions of quantum many-body problems. While a universal and scalable quantum computer is still beyond the horizon, recent advances in qubit manufacturing and coherent control of synthetic quantum matter are leading to a new generation of intermediate scale quantum hardware.

The complexity underlying quantum many-body systems closely resembles the one encountered in many problems in the world of information and technology. In both contexts, the complexity stems from a large number of interacting degrees of freedom. A powerful strategy in the latter scenario is machine learning, a subfield of artificial intelligence where large amounts of data are used to extract relevant features and patterns. In particular, artificial neural networks have been demonstrated to be capable of discovering low-dimensional representations of complex objects from high-dimensional dataset, leading to the profound technological revolution we all witness in our daily life.

In this Thesis, we envision a new paradigm for scientific discovery in quantum physics. On the one hand, we have the essentially unlimited data generated with the increasing amount of highly controllable quantum hardware. On the other hand, we have a set of powerful algorithms that efficiently capture non-trivial correlations from high-dimensional data. Therefore, we fully embrace this data-driven approach to quantum mechanics, and anticipate new exciting possibilities in the field of quantum many-body physics and quantum information science. We revive a powerful stochastic neural network called a restricted Boltzmann machine, which slowly moved out of fashion after playing a central role in the machine learning revolution of the early 2010s. We introduce a neural-network representation of quantum states based on this generative model. We propose a set of algorithms to reconstruct unknown quantum states from measurement data and numerically demonstrate their potential, with important implications for current experiments. These include the reconstruction of experimentally inaccessible properties, such as entanglement, and diagnostics to determine sources of noise. Furthermore, we introduce a machine learning framework for quantum error correction, where a neural network learns the best decoding strategy directly from data. We expect that the full integration between quantum hardware and artificial intelligence will become the gold standard, and will drive the world into the era of fault-tolerant quantum computing and large-scale quantum simulations.

Acknowledgements

I would like to thank all the people that made this Thesis possible, and more importantly, very much enjoyable. First of all, I want to thank my advisor, Roger Melko, from whom I learned a great deal over the last four years. It has been a incredible experience to explore together new uncharted territories of science. I would like to thank the rest of my academic family as well: Lori, Lauren, Pedro, Matt and Anna. A special thanks goes to Bohdan, who being also my roommate, was forced to listen to me constantly ramble about neural networks and the sign problem. Many thanks to my friends and colleagues Andres, Evert, Giuseppe, Juan and Miles, who greatly enriched my scientific experience. I thank my PhD committee members, Anton Burkov, Daniel Gottesman and Matteo Mariantoni, for the helpful discussions and advice. I would also like to thank the co-supervisor of my MSc Thesis, Gabriele De Chiara, who first pointed me to Waterloo and the Perimeter Institute, and without whom I would not have joined such vibrant scientific community. Finally, and most importantly, I thank my parents, Patrizia and Giovanni, who always believed in me.

Table of Contents

List of Figures

Chapter 1

Introduction

Emergence is a fundamental, and most intriguing, property of nature. From the microscopic scale of particles, to the macroscopic scale of human society, emergence leads to unseen collective behaviours from simple elementary laws. Given a collection of degrees of freedom, mutual interactions lead to collective phenomena, impossible to predict from the properties of the single constituents. A remarkable example of emergence at the human scale is ant colonies. This self-organizing organisms are capable of performing complicated tasks very efficiently, as well as collectively solving complex problems. Through a set of simple (chemical) interactions and without any centralized coordination, ants self-organize their division of labour according to the current need of the colony. The same type of emergent behaviour governs the synchronized flashing of fireflies, the swarming of birds during migrations, as well as the "self-organization" of modern society.

The concept of emergence plays a particularly important role also in the physics of condensed matter systems [1], where the elementary components are interacting particles. Besides traditional solid state materials, these also include synthetic quantum matter engineered in laboratories, such as ultra-cold atoms in optical traps [2] or low-dimensional quantum magnets [3, 4]. For these systems, the presence of interactions often leads to new physical phenomena, unaccessible in the framework of single-particle physics [5]. In particular, many exotic behaviours occur in the regime of strong interactions, such as

fractional quantum-Hall effect [6], high-temperature superconductivity [7], topologically ordered phases [8] and many-body localizations [9]. Intimately connected with emergent phenomena is the inherent complexity of the description of such physical systems, arising from an increasing number of interacting elementary constituents. In this scenario, complexity leads to an exponential scaling of the phase space of the system with the number of particles. As a consequence, given complete knowledge of the underlying microscopic laws, the behaviour of a system becomes in practice increasingly harder, and ultimately impossible, to be exactly determined.

1.1 The sharp end of complexity

Before venturing into the realm of quantum mechanics, let us first demonstrate the complexity of many-body systems in classical physics. Imagine a gas of N classical particles with position $\boldsymbol{q}_j$ and momentum $\boldsymbol{p}_j$, and some Hamiltonian $H(\boldsymbol{q}, \boldsymbol{p})$. The time evolution of the particles is perfectly determined, in a precise mathematical framework, by the Hamilton equations of motion. Unfortunately, for a typical volume of gas containing $N \sim 10^{23}$ particles, solving this system of $6N$ equations is fundamentally not possible. This exploding complexity at the microscopic scale stands however in contrast with the behaviour of the system at the macroscopic scale, which is well determined by a small set of parameters, such as the pressure, the volume and the temperature, defining the *macrostate* of the system. In turn, there are many possible *microstates* (specific microscopic configurations) compatible with a given macrostate. The seemingly impossible task of extracting the macroscopic behaviour from this microscopic turmoil can be however accomplished with the probabilistic framework of statistical mechanics.

For simplicity, and in analogy with the rest of this thesis, let us now pin down the electrons in space and consider only the interactions between their magnetic moments σ_j^z. Rather than position and momentum, the state of the system is now described by a vector of magnetic orientations $\boldsymbol{\sigma}^z$. If we allow energy fluctuations, this effective spin system is characterized by the canonical Boltzmann distribution $p(\boldsymbol{\sigma}^z) \propto e^{-\beta H(\boldsymbol{\sigma}^z)}$, where $H(\boldsymbol{\sigma}^z)$ is

now a magnetic Hamiltonian, and $\beta = 1/T$ is the inverse temperature. If, for instance, we are interested in the average energy U, we can calculate its expectation value as

$$U = \langle H \rangle = Z^{-1} \sum_{\boldsymbol{\sigma}^z} H(\boldsymbol{\sigma}^z)\, e^{-\beta H(\boldsymbol{\sigma}^z)}\,, \tag{1.1}$$

where $Z = \sum_{\boldsymbol{\sigma}^z} e^{-\beta H(\boldsymbol{\sigma}^z)}$ is the *partition function*, and the sum runs over the full configuration space [1]. If the system is non-interacting, the summation can be easily carried out, leading to an exact solution of the model. In some other cases, such as in the high-temperature limit or for weak interactions, accurate (but approximate) solutions can be gained through perturbative expansions. However, in the more general scenario outside the reach of analytical tools, an approximate solution must be obtained using a numerical technique. Unfortunately, the calculation of the partition function (as well as of any observable) requires an integration over the full configuration space. Therefore, to calculate the energy, we need to perform a sum over all the possible states (e.g. 2^N for spin-$\frac{1}{2}$), inevitably leading to an exponential scaling of computational resources.

Despite the system being described by a classical theory, and notwithstanding the fact that we fully understand the underlying physical laws, this intrinsic complexity prevents us from exploring the physics of a system composed of a large number of particles. Remarkably, the proposal of a set of algorithms allowed classical computers to overcome this exponential scaling, and to reliably extract macroscopic properties. A notable example is Monte Carlo (MC). The general idea behind MC algorithms, is to simulate the evolution of a system by generating a sequence of microstates [2]. The microstates are selected via a proposal step, followed by acceptance/rejection, which can vary with different MC implementations. One example is importance sampling, which we will cover with in more details in Chapter 2. The result is a collection of microstates (called a *Markov chain*), each occurring with a probability approximately equal to the Boltzmann distribution $p(\boldsymbol{\sigma}^z)$. The Markov chain is then used to approximate the average value of some observable, thus avoid-

[1] Equation 1.1 holds for any physical observable.

[2] Note that this evolution is performed in so-called *Markov time*, not to be confused with real time evolution. In fact, MC algorithms can only simulate physical systems at equilibrium.

ing the full enumeration of exponentially many states. In general, accurate averages can be obtained in polynomial time [3] $T \sim O(\text{poly}(N_{\text{mc}}))$, with a statistical error $\varepsilon \sim 1/\sqrt{N_{\text{mc}}}$, where N_{mc} is the number of samples in the Markov chain.

The scenario changes when we instead consider a system in a regime where quantum effects are non-negligible, or possibly predominant. As the system transitions into a quantum mechanical regime, the appearance of quantum coherences leads to an exponential scaling of the Hilbert space, where the "state" governing the physics of the system is defined. This exponential scaling in complexity for quantum states of matter is often referred to as the *quantum many-body problem*.

1.1.1 The quantum many-body problem

In analogy to the classical spins, we now wish to extract the macroscopic physics of a quantum many-body system, which we can imagine as N quantum spins σ_j^z, interacting with a Hamiltonian $\hat{H}$. As the specific problem, we consider the stationary Schrödinger equation

$$\hat{H}|\psi\rangle = E|\psi\rangle \,, \tag{1.2}$$

and aim to obtain the ground state $|\psi_0\rangle$ and ground state energy

$$E_0 = \langle\psi_0|\hat{H}|\psi_0\rangle \,. \tag{1.3}$$

As a concrete example, we consider the *Heisenberg model*, an important model of magnetism that will serve as a reference for the remainder of the Chapter. For a one-dimensional ($1d$) chain of spin-$\frac{1}{2}$ particles with anti-ferromagnetic interactions, the Hamiltonian is

$$\hat{H} = \sum_{j=1}^{N-1} \hat{\boldsymbol{S}}_j \cdot \hat{\boldsymbol{S}}_{j+1} \,. \tag{1.4}$$

[3] This scaling applies as long as the sampling is ergodic. The ergodicity is a central issue in MC, and prevents simpler techniques to simulate systems with a rough free energy landscape, such as spin glasses.

where $\hat{S}_j^\alpha = \frac{1}{2}\hat{\sigma}_j^\alpha$ and $\hat{\sigma}_j^\alpha$ are the Pauli matrices ($\alpha = x, y, z$). The Hamiltonian can be easily re-written in the more convenient form

$$\hat{H} = \sum_{j=1}^{N-1} \left[\hat{S}_j^z \hat{S}_j^z + \frac{1}{2}\left(\hat{S}_j^+ \hat{S}_{j+1}^- + \hat{S}_j^- \hat{S}_{j+1}^+ \right) \right] \tag{1.5}$$

in terms of the raising $\hat{S}_j^+$ and lowering $\hat{S}_j^-$ spin operators [4]. Let us first consider just two interacting spins with Hamiltonian $\hat{H} = \hat{\boldsymbol{S}}_1 \cdot \hat{\boldsymbol{S}}_2$. The energy spectrum and eigenstates can be obtained by performing an exact diagonalization (ED) of the Hamiltonian [10]. This requires the choice of a basis for the Hilbert space and the storage of the Hamiltonian matrix in the computer The obvious choice for the basis is $\{|S_1^z, S_2^z\rangle = |\uparrow\uparrow\rangle, |\uparrow\downarrow\rangle, |\downarrow\uparrow\rangle, |\downarrow\downarrow\rangle\}$. The Hamiltonian in this basis is given by the 4×4 matrix:

$$\langle \boldsymbol{S}^z | \hat{H} | \boldsymbol{S}^{z\prime} \rangle = \begin{pmatrix} 1/4 & 0 & 0 & 0 \\ 0 & -1/4 & 1/2 & 0 \\ 0 & 1/2 & -1/4 & 0 \\ 0 & 0 & 0 & 1/4 \end{pmatrix} . \tag{1.6}$$

The full spectrum, which provides the complete knowledge of the physics of the system, can be found using standard routines of linear algebra. This produces the desired ground state eigenvector, finding the singlet state as lowest energy state

$$|\psi_0\rangle = \frac{1}{\sqrt{2}}\left(|\uparrow\downarrow\rangle - |\downarrow\uparrow\rangle \right) \tag{1.7}$$

and lowest energy eigenvalue $E_0 = -3/4$.

[4] These operators act as $\hat{S}_j^+|0\rangle = |1\rangle$, $\hat{S}_j^-|1\rangle = |0\rangle$ and $\hat{S}_j^-|0\rangle = \hat{S}_j^+|1\rangle = 0$.

For the full system of N spins, the basis is now given by $\{|\boldsymbol{S}^z\rangle = |S_1^z, S_2^z, \ldots, S_N^z\rangle\}$, and the Hamiltonian matrix can be constructed efficiently as follows:

$$H(\boldsymbol{S}^z, \boldsymbol{S}^{z\prime}) = \begin{cases} \sum_{j=1}^{N-1} S_j^z S_{j+1}^z, & \text{if } \boldsymbol{S}^z = \boldsymbol{S}^{z\prime} \\ +1/2 & \text{if } \boldsymbol{S}^z \text{ and } \boldsymbol{S}^{z\prime} \text{ differ for 2 nearby spin flips} \\ 0 & \text{otherwise} \end{cases} \tag{1.8}$$

However, the dimension of the matrix is $2^N \times 2^N$, and every time a new spin is added to the chain, the Hamiltonian matrix of the system doubles in size. Once again we face an exponential scaling, now of the Hilbert space of the system, which restricts us to very small system sizes. But how small in practice? The math can illuminate us. The CPU-time for full diagonalization of a $n \times n$ matrix is $T \sim n^3$ (in our case $n = 2^N$). In order to convert the cpu-time to computational time, we need to know how many operation per seconds the machine can perform. For instance, for a typical laptop this number ranges around 100 GFLOPs (i.e. 10^{11} floating point operations per second). This implies that it would take a few micro-seconds to obtain the full spectrum of $N = 8$ spins, around 45 minutes for $N = 16$ and 150 years for $N = 24$.

We were expecting an exponential scaling of classical resources, but regardless, this numbers seem rather disappointing. How do we expect to obtain any physical insights on realistic samples of matter, if simulations with classical computers would require more time than the age of the universe? Unless we own a quantum computer (more on this later), in order to explore larger systems with classical simulations, we need to find clever ways to fight this "curse of dimensionality". For example, the symmetries of the Hamiltonian can be used for reducing the computational time, by diagonalizing a block of the Hamiltonian corresponding to a specific conserved quantum number. For the Heisenberg model, the Hamiltonian is rotationally invariant, and thus it conserves the square of the total spin angular momentum, $\hat{\boldsymbol{S}}_T^2 = (\sum_j \hat{\boldsymbol{S}}_j)^2$ and the total z-component $S_T^z = \sum_j \hat{S}_j^z$. As such, we can build blocks corresponding to a specific value of S_T^z, and target the sector $S_T^z = 0$ to find the ground state. For the case of two spins, for example, the block corresponding to

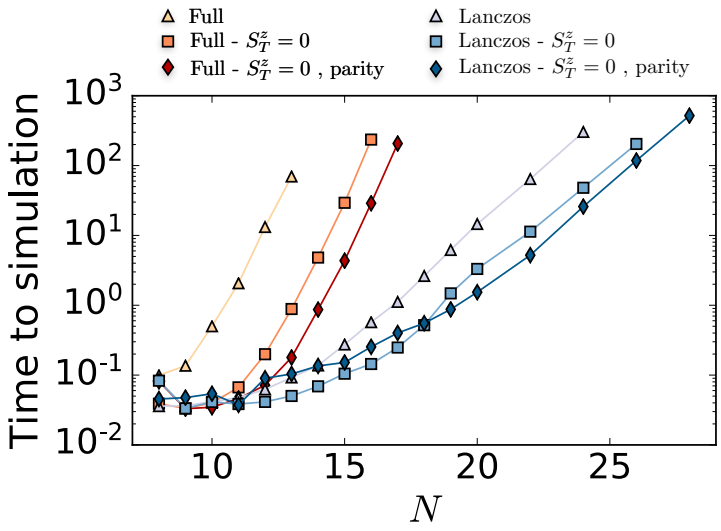

Figure 1.1: **Exact diagonalization**. We show the time (in seconds) required to reach the solution of the ED simulation for the $1d$ Heisenberg model. We carried out full diagonalization, where the whole spectrum is obtained, as well as the Lanczos algorithm targeting only the ground state. For both cases, we plot the results from considering the full matrix Hamiltonian, the $S^z_T = 0$ sector and the parity/reflection symmetry. Simulation were performed using the library *QuSpin* [11].

total magnetization $S^z_T = 0$ is

$$\langle \boldsymbol{S}^z | \hat{H} | \boldsymbol{S}^{z\prime} \rangle_{S^z_T=0} = \begin{pmatrix} -1/4 & 1/2 \\ 1/2 & -1/4 \end{pmatrix} . \tag{1.9}$$

This clearly helps, but the scaling $O(2^N/N)$ of the block size is still exponential. Additional symmetries can be implemented to further reduce the exponential overhead, such as reflection symmetry for this example. Finally, if we are interested in the ground state properties only, we don't need to solve the problem for the full energy spectrum. Using iterative diagonalization techniques (such as the Lanczos method [12]), one can obtain more efficient scaling by targeting the lowest end of the spectrum, obtaining the ground state and possibly a few low-lying excited states. In Fig. 1.1 we show the results of a set of ED simulations for the $1d$ Heisenberg model. In any instance we observe the exponential

scaling of the time to simulation. The slope of the scaling is clearly different depending on whether we are performing full diagonalization or extracting just the lowest energy state with the Lanczos algorithm.

Contrary to the classical scenario, where the state of the system was described by a vector of N orientations, quantum states suffer the very same exponential scaling of the Hilbert space. For the case above, the ground state $|\psi_0\rangle$ for N spins contains a total of 2^N complex coefficients. Consequently, if we simply store all coefficients, we need to account also for the resource scaling required for storage. We now give an example to put such exponential scaling in perspective (Fig. 1.2). We assume the quantum state (containing only real coefficients) is stored onto a set of hard drives as a vector of double-precision floating point numbers (8 bytes). A 10TB hard drive ($\sim$350\$) can store the quantum state for $N \simeq 40$ spins. By filling up a standard ship container with HardDrives (i.e. $\sim$88000), we can reach $N \simeq 56$. It would take the OOCL *Hong Kong* (currently the largest container ship with a capacity of 21413 containers) to get to $N \simeq 70$. Finally, for a dramatic finish, in order to store the quantum state for $N \simeq 90$ spins we must line up around 1 million ships (roughly the earth-moon distance), full of containers, full of hard drives. So, in practice, even in the fortunate case where a powerful numerical technique allows us to obtain an arbitrary large quantum state, its storage on a regular (classical) memory would become physically impossible for more than a small number of particles.

The cutting edge of ED algorithms now ranges up to and over $N = 50$ (for spin-$\frac{1}{2}$ degrees of freedom [13, 14]). These simulations are realized by fully exploiting all the symmetries in a highly structured Hamiltonian, as well as by cleverly parallelizing linear algebra operations over many computational cores. But this is usually not enough. Large super-computers are needed to explore these regimes, with a considerable cost in energy and money. Nevertheless, the outcome is worth the effort, since ED provides the exact results, the indisputable ground truth. This is in practice always required for benchmarking any other (approximate) technique. Over and above, some physical insights can be often obtained nonetheless from a moderate number of particles [15, 16]. Unfortunately however, collective phenomena in condensed matter emerge as the number of particles grows large, and become crisp in the thermodynamic limit ($N \to \infty$). It is also in this limit that

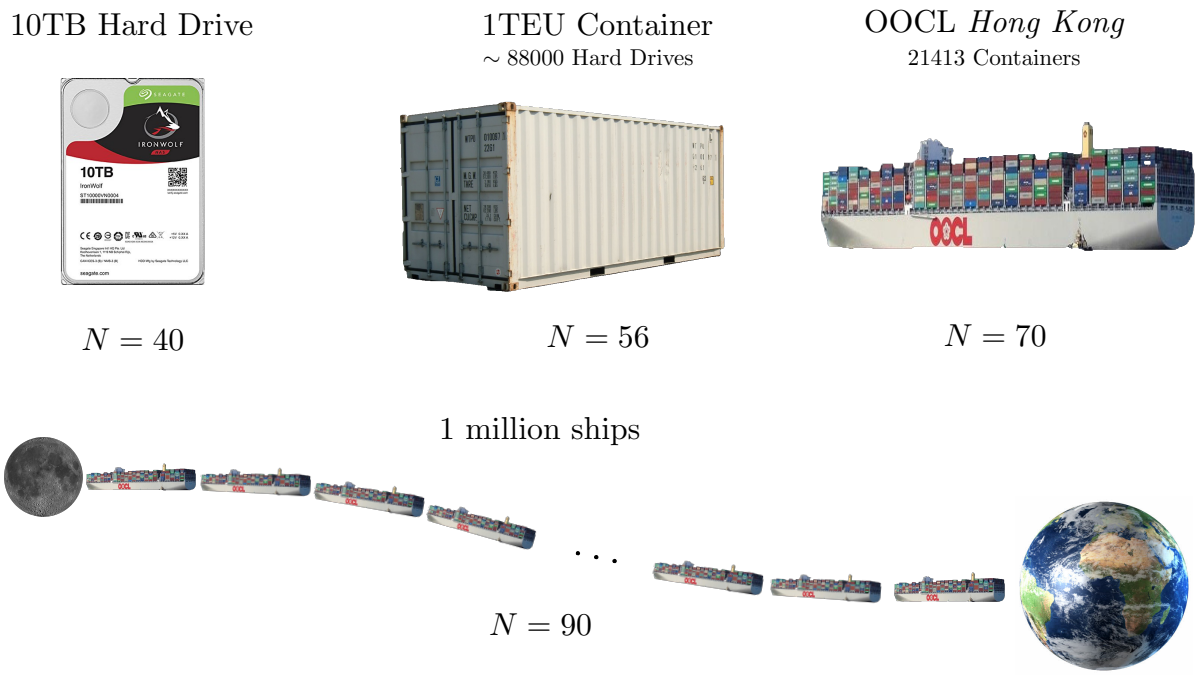

Figure 1.2: **The quantum many-body problem**. Practical picture of the exponential scaling of the Hilbert space. The various images correspond to the amount of classical resources (in units of a 10TB hard drives) required to fully store a quantum state $|\psi\rangle$ for N spins. A TEU is a volumetric measure of containers, with 1 TEU = 33.2m^3.

phase transitions and universal properties, a central subject of study in condensed matter, are well defined. So, on one hand, the nature of emergent phenomena requires large-scale simulations to obtain the correct physical properties. On the other hand, exact simulations of such large-scale materials face an infinitely complex task and will never be feasible, no matter the technological power we will ever possess.

Given this fundamental conundrum, it seems that the only way forward is to abandon exact solutions, and resort to approximate techniques instead. Given the broadness of the subject, here we will only cover two successful strategies to efficiently simulate large condensed matter systems on classical computers (tensor network states and quantum Monte Carlo). Before discussing these algorithms, we present first a general notation and definitions for quantum many-body states.

1.1.2 Quantum states

We assume a set of N discrete (generic) quantum degrees of freedom, each with a local Hilbert space $\mathcal{H}$. The many-body quantum state is defined in the Hilbert space $\boldsymbol{\mathcal{H}} = \mathcal{H}^{\otimes N}$. Given a set of quantum numbers $\{x_j\}$, an orthonormal reference basis of the Hilbert space is given by $\{|\boldsymbol{x}\rangle\}$, with

$$|\boldsymbol{x}\rangle = |x_1, x_2, \ldots, x_N\rangle \, . \tag{1.10}$$

These could be electronic orbitals, magnetic spins projections or the logical states of qubits, for example. If we assume that the physical system is perfectly isolated (i.e. no interaction with the environment), the quantum state, called *pure*, is described by a quantum wavefunction $|\psi\rangle \in \boldsymbol{\mathcal{H}}$. The representation of the quantum state in the reference basis is

$$|\psi\rangle = \sum_{\boldsymbol{x}} \psi(\boldsymbol{x}) |\boldsymbol{x}\rangle \, . \tag{1.11}$$

where $\psi(\boldsymbol{x}) = \langle \boldsymbol{x}|\psi\rangle$ is a high-dimensional vector of complex coefficients, and normalization imposes $\sum_{\boldsymbol{x}} |\psi(\boldsymbol{x})|^2 = 1$.

The quantum mechanical description differs when the purity of the state cannot be assumed. This is often true in realistic implementations of controlled quantum systems in the laboratory. In general, either because of thermal fluctuations or quantum correlations between the system and the environment, the quantum wavefunction of the material (or hardware) is not perfectly determined. Rather, the system is in a statistical mixture of pure states $|\psi_i\rangle$, each with a corresponding probability p_i. In this case, the representation of the quantum state, called *mixed*, is formulated in terms of the density operator

$$\hat{\rho} = \sum_i p_i |\psi_i\rangle\langle\psi_i| \tag{1.12}$$

with normalized states $|\langle\psi_i|\psi_i\rangle|^2 = 1$ and $\sum_i p_i = 1$. Within the reference basis $\{|\boldsymbol{x}\rangle\}$, the

density matrix for the system is

$$\hat{\rho} = \sum_{\boldsymbol{x},\boldsymbol{x}'} \rho(\boldsymbol{x},\boldsymbol{x}')|\boldsymbol{x}\rangle\langle\boldsymbol{x}'| \,, \tag{1.13}$$

with matrix elements:

$$\rho(\boldsymbol{x},\boldsymbol{x}') = \langle\boldsymbol{x}|\,\hat{\rho}\,|\boldsymbol{x}'\rangle = \sum_i p_i \psi_i(\boldsymbol{x})\psi_i^*(\boldsymbol{x}') \quad , \quad \psi_i(\boldsymbol{x}) = \langle\boldsymbol{x}|\psi_i\rangle \,. \tag{1.14}$$

The diagonal elements of the density matrix, $\rho(\boldsymbol{x},\boldsymbol{x}) = \sum_i p_i|\psi_i(\boldsymbol{x})|^2$, are called *populations*. They correspond to the set of probabilities for the system to be found in the state $|\boldsymbol{x}\rangle$ (upon a projective measurement in that basis). The off-diagonal elements $\rho(\boldsymbol{x},\boldsymbol{x}')$ are called *coherences*, as they encode the relative phase between the various basis states. [5]

The density operator formalism provides the most general framework to write quantum states. This representation however, requires more strict conditions than the pure state representation. As for wavefunctions, the density operator must be normalized, $\mathrm{Tr}(\hat{\rho}) = 1$. This follows directly from the normalization of the pure states $|\psi_i\rangle$ and the normalization of the probability distribution p_i. In addition, a physical density matrix must obey two other conditions. First, it must be a self-adjoint operator $\hat{\rho} = \hat{\rho}^\dagger$, and second, it must be positive semi-definite, which means that for any state $|\xi\rangle$ in the Hilbert space $\boldsymbol{\mathcal{H}}$, it holds $\langle\xi|\hat{\rho}|\xi\rangle \geq 0$. Provided these three condition are met, the operator $\hat{\rho}$ is a physical density matrix and can represent both a pure state and a mixed state. If the state is pure, the statistical mixture contains only one state $|\psi_i\rangle$ with unit probability, and thus, the density matrix has only one non-zero eigenvalue $\tau_i = 1$, reducing to $\rho(\boldsymbol{x},\boldsymbol{x}') = \psi_i(\boldsymbol{x})\psi_i^*(\boldsymbol{x}')$ in the reference basis. Because of this, the trace squared of the density matrix is $\mathrm{Tr}(\hat{\rho}^2) = 1$. On the other hand, if the state is not pure, it follows that $\mathrm{Tr}(\boldsymbol{\rho}^2) = \sum_i \tau_i^2 < 1$ (since normalization requires $\sum_i \tau_i = 1$). This is why the trace of the density matrix squared is often referred to as the *purity* of the quantum state $\hat{\rho}$ and, as we will see, it is strictly related to the entanglement between disjoint sub-regions of the system.

[5] By writing $\psi_i(\boldsymbol{x}) = |\psi_i(\boldsymbol{x})|e^{i\phi_i(\boldsymbol{x})}$, we see that the coherence $\rho(\boldsymbol{x},\boldsymbol{x}')$ depends on the set of phase differences $\phi_i(\boldsymbol{x}) - \phi_i(\boldsymbol{x}')$ of the states $|\psi_i\rangle$, between the two basis states $|\boldsymbol{x}\rangle$ and $|\boldsymbol{x}'\rangle$.

1.2 Tensor networks

The first technique we review is tensor networks (TN), one of the most powerful technique to reduce the exponential overhead of quantum states. In this framework, a (pure) quantum state is compressed using a networks of tensors, such that the total number of parameters in the network is much less than the size of the Hilbert space. The idea behind this strategy is that local Hamiltonians are characterized by only a small number of parameters $O(N)$, and as such, its ground state must reside in a small region of the Hilbert space, constrained by the Hamiltonian. Let us consider again a generic state of N quantum spins $\psi(\boldsymbol{\sigma}^z)$, which contains 2^N coefficients. This is equivalent to a rank-N tensor $\psi^{\sigma_1^z\,\sigma_2^z\cdots\sigma_N^z}$, where each of the N two-dimensional ($2d$) indices (called *physical*) represents the quantum number σ_j^z. The TN approach consists of replacing the big tensor $\psi^{\sigma_1^z\,\sigma_2^z\cdots\sigma_N^z}$ with a set of small tensors, with the aim of obtaining a representation with a polynomial number of parameters. Irrespective to the specific type of TN at hand, an efficient construction of the compressed quantum state relies then on the ability to target and parametrize this "corner" of the Hilbert space, typically consisting of ground states of gapped local Hamiltonians. As we are about to see, the fundamental ingredient characterizing this sub-space is entanglement.

1.2.1 The entanglement area law

Entanglement is among the most counter-intuitive properties of quantum mechanics. The entanglement between two quantum systems refers to the quantum correlations that cannot be captured by any local classical theory. Aside from being a fundamental resource for quantum technologies, entanglement also provides a powerful language to describe quantum many-body systems, leading to a novel perspective in condensed matter physics [17]. Consider again a pure state $|\psi\rangle$ and a bipartition of the system into two complementary sub-regions A and B. Then, the state can be always written using a Schmidt decomposition

$$|\psi\rangle = \sum_k \sqrt{p_k}|\psi_A^k\rangle \otimes |\psi_B^k\rangle \,, \tag{1.15}$$

where p_k are called *Schmidt coefficients*. If only one coefficient is different from zero, than that state becomes separable $|\psi\rangle = |\psi_A\rangle \otimes |\psi_B\rangle$. There are no quantum correlation in $|\psi\rangle$, and the quantum state of region A, $\hat{\rho}_A = \mathrm{Tr}_B(|\psi\rangle\langle\psi|) = |\psi_A\rangle\langle\psi_A|$ is fully determined by the pure state $|\psi_A\rangle$. In turn, if the decomposition presents more that one term in the sum, than the two sub-regions are entangled. This means that the state of sub-region A, if examined separately, is not perfectly determined, i.e. it is described by a "mixed" density operator ($\mathrm{Tr}(\hat{\rho}_A^2) < 1$). Therefore, two sub-systems are said to be entangled if the composite quantum state cannot be written in the separable form $|\psi\rangle = |\psi_A\rangle \otimes |\psi_B\rangle$.

This definition leads to the natural question on how to quantify the amount of entanglement. One might be tempted to use the number of non-zero Schmidt value as a candidate for the measure. Consider for instance the following quantum state of two sub-systems A and B, each containing one degree of freedom

$$|\phi^+\rangle = \frac{1}{\sqrt{2}}\big(|00\rangle + |11\rangle\big)\,, \tag{1.16}$$

also called a *Bell state*. This state is already in the Schmidt form, with two Schmidt values $p_1 = p_2 = 1/2$. Compare this with the following state [18]

$$|\psi^\epsilon\rangle = \sqrt{(1-2\epsilon^2)}|00\rangle + \epsilon|11\rangle + \epsilon|22\rangle \tag{1.17}$$

in the limit of $\epsilon \ll 1$. Clearly, even though the number of Schmidt coefficient is larger, for small ϵ the quantum state is close to the product state $|00\rangle$, effectively containing a small amount of quantum correlations.

A more appropriate measurement of entanglement is determined by the specific distribution of the Schmidt values, rather than their total number. For a pure quantum state, a measurement of entanglement is provided by the family of Renyi entropies [19]

$$S_\alpha(\hat{\rho}_A) = \frac{1}{1-\alpha}\log(\mathrm{Tr}(\hat{\rho}_A^\alpha))\,, \tag{1.18}$$

where $\hat{\rho}_A$ is the reduced density matrix of sub-system A, $\alpha > 0$ is called the *Renyi index*,

and $S_\alpha(\hat{\rho}_A) \geq S_\beta(\hat{\rho}_A)$ for $\alpha < \beta$. For the specific case of $\alpha = 1$, one obtains

$$S_{\alpha=1}(\hat{\rho}_A) = -\mathrm{Tr}(\hat{\rho}_A^\alpha \log \hat{\rho}_A^\alpha)) = -\sum_k p_k \log p_k \,, \tag{1.19}$$

which is called *von Neumann* entanglement entropy. From this definition, it is easy to prove that the maximum entropy $S_{\max}(\hat{\rho}_A) = \log n$ (independent of α) is obtained when all Schmidt values are $p_k = \frac{1}{n}$ (here n is lesser of the total dimensions of the two Hilbert spaces of sub-systems A and B). In this case, the composite state $|\psi_{AB}\rangle$ is called *maximally-entangled*. The Bell state $|\phi^+\rangle$, for example, is *maximally-entangled* with $S_\alpha(\hat{\rho}_A) = \log 2$, and both sub-systems described by a maximally-mixed states $\hat{\rho}_{A/B} = \frac{1}{2}\hat{I}$. For the quantum state $|\psi^\epsilon\rangle$ however, for $\epsilon \ll 1$ the von Neumann entropy is $S_1(\hat{\rho}_A) \sim O(\epsilon^2 \log \frac{1}{\epsilon}) \ll \log 2$, as expected. Finally, we point out the second Renyi entropy $S_{\alpha=2}(\hat{\rho}_A) = -\log(\mathrm{Tr}(\hat{\rho}_A^2))$, which is the logarithm of the purity of the quantum state $\hat{\rho}_A$.

A fundamental question in the study of condensed matter system is the behaviour of the entanglement entropy for a sub-region of the system. In general, for a quantum state picked at random in the full Hilbert space, the entanglement entropy is proportional to the number of degrees of freedom in A [20]. This is called a *volume law* of entanglement. In contrast, quantum states that are ground states of local Hamiltonians show a very different scaling of entanglement, which is proportional only to the boundary ∂A of the sub-region [21]. An important difference in the entanglement arises from the existence of a gap in the energy spectrum. While in general both gapped and gapless ground states of local Hamiltonians obey a dominant *area law* scaling of entanglement [22], gapless (critical) states possess sub-leading universal corrections to the entanglement entropy [23]. The general intuition for the area law is that the short-ranged nature of the interactions leads to quantum correlations that are locally distributed. As such, when we define a sub-region A of the full system, only the degrees of freedom near the boundary ∂A contributes to the entropy, with very low entanglement between degrees of freedom far apart from each other.

The entanglement area law has been formally proven for $1d$ quantum systems by Hastings [24], based on the existence of the Lieb-Robinson bound [25], and confirmed by many numerical calculations [26, 27]. In this case, the entanglement entropy for ground states of

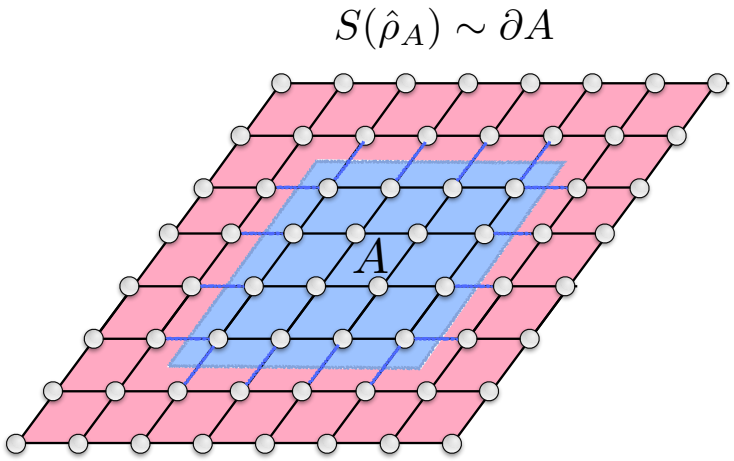

Figure 1.3: **Entanglement area law**. For ground states of gapped local Hamiltonian, the entanglement entropy $S_\alpha(\hat{\rho}_A)$ of a region A scales with the surface ∂A of the region.

local gapped Hamiltonians is constant and independent on the block size of the sub-region, $S(\hat{\rho}_A) \simeq O(1)$. Note that, while the Lieb-Robinson theorem is at the base of the exponential decay of correlations, the scaling of the entanglement and correlations are not directly related. In fact, there exists states with exponentially decaying correlations and large entanglement entropy [24]. Similarly, the area law can also be violated by gapped ground states of Hamiltonians with long range interactions, which approach a volume law scaling [28]. When the physical system is brought to a critical point (i.e. gapless ground state), the correlation length diverges ($\xi \to \infty$) and the system, which is now scale invariant, is described by a conformal field theory. Given that, powerful methods allows the calculation of the universal properties of the system, including entanglement entropy [29, 30]. For a $1d$ system containing L sites, one finds [31]

$$S_1(\hat{\rho}_A) = \frac{c}{3} \log\left(\frac{L}{a}\right) + A \tag{1.20}$$

where a is the lattice spacing (i.e. the ultraviolet regularization cut-off), c is the central charge of the field theory and A is a non-universal constant.

1.2.2 Matrix product states

Matrix product states (MPS) for $1d$ quantum systems are the most successful realization of a TN. Coupled with the density-matrix renormalization group (DMRG) framework [32], MPSs allows one to solve most of the physics of strongly-correlated systems in one dimension. The (exponentially large) tensor $\psi^{\sigma_1^z \sigma_2^z \cdots \sigma_N^z}$ is written here as a product of tensors along a $1d$ array. To each quantum degree of freedom (say $\sigma_j^z = \pm 1$) is associated a rank-3 tensor $M^{\sigma_j^z}_{\alpha_j,\alpha_{j+1}}$ with one physical index σ_j^z and two *bond* indices (α_j, α_{j+1}), whose dimension can vary. The MPS is then defined as

$$\psi(\boldsymbol{\sigma}^z) = \sum_{\alpha_1} \sum_{\alpha_2} \cdots \sum_{\alpha_{N-1}} [M_1]^{\sigma_1^z}_{\alpha_1} \, [M_2]^{\sigma_2^z}_{\alpha_1\alpha_2} \, [M_3]^{\sigma_3^z}_{\alpha_2\alpha_3} \, \ldots \, [M_N]^{\sigma_N^z}_{\alpha_{N-1}} \tag{1.21}$$

where the two tensors at the boundary have only one bond index. This is simply done in order to avoid taking the trace to obtain the wavefunction coefficient. To simplify the notation and avoid to keep all the sums and indices around, TNs also come with a powerful diagrammatic notation for both visualization and computation. Very simply, each tensor is represented by an object (e.g. a circle/square) with a number of legs (i.e. the indices). Connecting two tensors together corresponds to a contraction of the shared indices. Using this notation, a MPS is simply represented by a $1d$ chain of connected tensor, with N uncontracted physical indices (Fig. 1.4). To evaluate one element of the wavefunction $\psi(\boldsymbol{\sigma}^z)$ we just need to calculate the product of N matrices $M_j^{\sigma_j^z}$, each one selected according to the specific value of the state σ_j^z.

The dimension of each bond index α_j is a free parameter of this parametrization. The largest dimension $\chi = \max_{j\in(1,N)}(\dim(\alpha_j))$ is called the *bond dimension* of the MPS, and it is a natural convergence parameter of the representation. Note that for $\chi = 1$ the MPS is simply the product state

$$\psi(\boldsymbol{\sigma}^z) = M_1^{\sigma_1^z} \, M_2^{\sigma_2^z} \, M_3^{\sigma_3^z} \, \ldots \, M_N^{\sigma_N^z} \, , \tag{1.22}$$

which reduces to a mean-field theory description. However, as we increase the bond dimension, we obtain a MPS representation capable of sustaining an increasing amount of

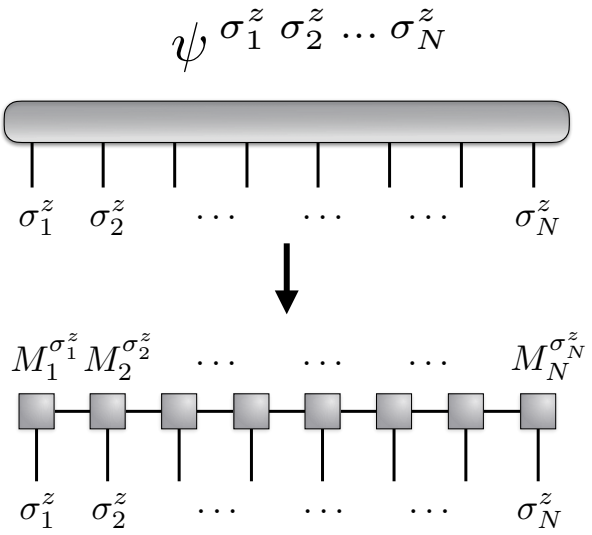

Figure 1.4: **Matrix product states**. Decomposition of a wavefunction into a MPS form, carried out via a series of singular values decomposition (not shown in the figure).

entanglement. This comes at a price of a large number of parameters in the MPS, and thus a slower computational efficiency. However, we know that ground states of gapped local Hamiltonians obey the area law of entanglement. For a $1d$ system, this implies that entanglement is independent on the block size L of the sub-system (at finite correlation length). The same entanglement scaling, related to the geometry of the TN, is captured by a MPS with only a polynomial number of parameters. More specifically, for a given bond dimension χ, the total number of parameter in the MPS is equal to $2N\chi^2$, which effectively reduces the scaling from exponential to polynomial (as long as the bond dimension remains sufficiently low). When the gap closes and the correlation length diverges, the entanglement entropy picks up a logarithmic correction in the size of the block. In such case, an MPS is still capable of obtaining a faithful representation, at a cost of a larger bond dimensions. In general, MPSs can populate the full Hilbert space, and represent exactly any pure quantum state, under the weak condition of an exponentially large bond dimension $\chi \to 2^N$. As such, entanglement in a quantum state is the main limiting factor for its representation using a MPS.

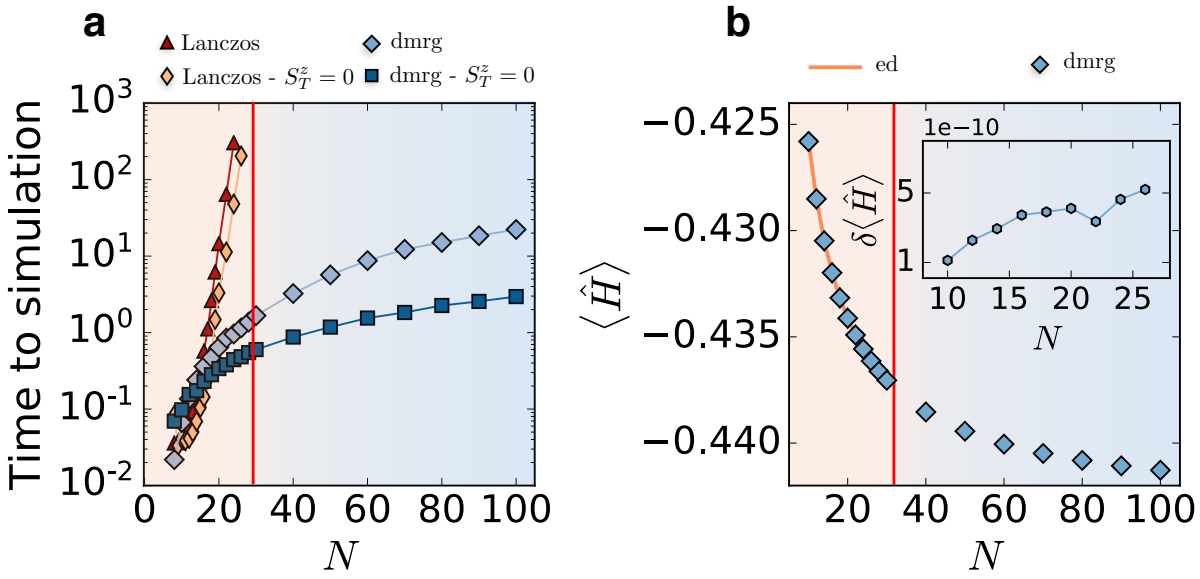

Figure 1.5: **DMRG simulation of the Heisenberg model**. We perform the simulation of the Heisenberg model in one dimension by optimizing a MPS with the DMRG algorithm. **a)** We compare the time to simulation for ED with Lanczos and MPSs for both the full Hamiltonian and the $S^z_T = 0$ sector. We can clearly see how the MPS in one dimension overcomes the complexity of the quantum many-body problem. **b)** Ground state energy per spin, as a function of system size. For N up to 28, we compare the DMRG results with ED. In the inset we show the relative error $\delta E_0 = |E_0^{\text{mps}} - E_0^{\text{ed}}|/|E_0^{\text{ed}}|$. The DMRG simulations were performed using the C++ library Itensor [33].

To show the power of MPSs in practice, we performed a DMRG simulation to find the ground state of the Heisenberg model, and compare with ED results. In Fig 1.5a, we show the scaling of the time to simulation against the system size. We show data-points from simulation of the full Hilbert space, as well as the sub-space corresponding to the conserved total spin $\hat{S}^z_T = 0$. In Fig 1.5b we show the ground state energy per spin as a function of N, and show in the inset the relative error with respect to the ED results. This clearly shows how MPSs are capable of taming the exponential scaling of quantum mechanics in one dimension, under certain reasonable conditions.

1.3 Quantum Monte Carlo

The framework of TNs has been most successful in one dimension. For higher-dimensional systems, simulations are generally restricted to moderate sizes, either due to the large bond dimension of an MPS representation, or the computational burden required to contract higher dimensional TNs, such as *projected entangled-pairs states* [34, 35]. Therefore, if we are interested in studying strongly-correlated systems in two or three spatial dimensions, we might consider a different approach, where one abandons the quantum state and considers directly expectation values of macroscopic equilibrium properties. In particular, *quantum Monte Carlo* (QMC) has been revealed to be capable of handling large-scale simulations, under certain assumptions. We will only overview the most traditional (and simplest) flavour of QMC, the discrete imaginary-time path integral, and discuss the sign problem, the major limitation of QMC algorithms.

1.3.1 Imaginary-time path integrals

As a system of interest, we consider a collection of quantum spins interacting with Hamiltonian $\hat{H}$ at inverse temperature β. The state of the system is now described by the thermal density matrix $\hat{\rho} = Z^{-1}\, e^{-\beta \hat{H}}$, where $Z = \mathrm{Tr}(e^{-\beta \hat{H}})$ is the quantum partition function. The exact expectation value of the energy, for example, is given by

$$U = \langle \hat{H} \rangle = Z^{-1} \mathrm{Tr}(\hat{\rho} \hat{H}) = Z^{-1} \sum_n E_n\, e^{-\beta E_n} |n\rangle\langle n| \tag{1.23}$$

where E_n are the energy eigenstates. In contrast to the case of zero-temperature, the computation of the expectation value now requires the full spectrum of the Hamiltonian, leading to a higher computation demand than for ground states. However, in the spirit of the success of MC for classical system at finite temperature, if we could sample quantum configurations $|\boldsymbol{\sigma}^z\rangle$ according to the partition function Z, we could compute expectation values as averages over Markov chains. Unfortunately, this is not possible due to the presence of a quantum Hamiltonian in the partition function. In other words, $e^{-\beta \hat{H}}$ has no

naive probabilistic interpretation. On the other hand, if there is a mapping from quantum states $|\boldsymbol{\sigma}^z\rangle$ to classical states $\mathbf{x}$, such that the partition function can be written exactly as

$$Z = \sum_{\mathbf{x}} W(\mathbf{x}) \,, \tag{1.24}$$

one could instead simulate the classical system using MC algorithms, under the requirement of being able to extract physical properties of the original system from the classical one. This is most simply done with a path integral expansion obtained from the time evolution in imaginary time, which effectively maps a d-dimensional quantum system into a $(d+1)$-dimensional classical system. As long as $W(\mathbf{x}) \geq 0$, the system can be described using statistical mechanics, with $W(\mathbf{x})$ being the (un-normalized) probability of the configuration $\mathbf{x}$ in the $(d+1)$ system.

To show how to access the distribution $W(\mathbf{x}) \geq 0$, we will look at a specific example, restricting to a one dimensional system for graphical simplification. For the Heisenberg model, due to the particular structure of the interaction, the Hamiltonian can be written as $\hat{H} = \hat{H}_A + \hat{H}_B$, with

$$\hat{H}_A = \sum_{j \in odd} \hat{\boldsymbol{S}}_j \cdot \hat{\boldsymbol{S}}_{j+1} \qquad\qquad \hat{H}_B = \sum_{j \in even} \hat{\boldsymbol{S}}_j \cdot \hat{\boldsymbol{S}}_{j+1} \,. \tag{1.25}$$

Each term in either Hamiltonian commutes with all others, but $[\hat{H}_A, \hat{H}_B] \neq 0$. Essentially, we are defining two sub-lattices A and B, containing odd and even site respectively, and interaction only takes place between spins from different sublattices. We can then re-write the exponential using the Suzuki-Trotter expansion, obtaining

$$e^{-\beta \hat{H}} = \prod_{j=1}^{m} e^{-\Delta\tau \hat{H}} \tag{1.26}$$

with $\Delta\tau = \beta/m$, and

$$e^{-\Delta\tau \hat{H}} = e^{-\Delta\tau(\hat{H}_A + \hat{H}_B)} = e^{-\Delta\tau \hat{H}_A} e^{-\Delta\tau \hat{H}_B} + O(\Delta\tau^2) \tag{1.27}$$

with error $O(\Delta\tau^2)$. The partition function becomes

$$\begin{aligned} Z = \mathrm{Tr}(e^{-\beta\hat{H}}) &= \mathrm{Tr}\bigg(\prod_{j=1}^{m} e^{-\Delta\tau\hat{H}_A}e^{-\Delta\tau\hat{H}_B}\bigg) + O(\Delta\tau^2) \\ &\simeq \sum_{\boldsymbol{\sigma}_1^z}\langle\boldsymbol{\sigma}_1^z|\prod_{j=1}^{m} e^{-\Delta\tau\hat{H}_A}e^{-\Delta\tau\hat{H}_B}|\boldsymbol{\sigma}_1^z\rangle\,, \end{aligned} \tag{1.28}$$

which can be re-written, using $2m$ resolutions of identities $\sum_{\boldsymbol{\sigma}_t}|\boldsymbol{\sigma}_t^z\rangle\langle\boldsymbol{\sigma}_t^z|$, as

$$Z = \sum_{\boldsymbol{\sigma}_1^z}\cdots\sum_{\boldsymbol{\sigma}_{2m}^z}\langle\boldsymbol{\sigma}_1^z|e^{-\Delta\tau\hat{H}_A}|\boldsymbol{\sigma}_2^z\rangle\langle\boldsymbol{\sigma}_2^z|e^{-\Delta\tau\hat{H}_B}|\boldsymbol{\sigma}_2^z\rangle\langle\boldsymbol{\sigma}_3^z|\ldots\langle\boldsymbol{\sigma}_{2m}^z|e^{-\Delta\tau\hat{H}_B}|\boldsymbol{\sigma}_1^z\rangle \tag{1.29}$$

This can be viewed as a series of $2m$ time slice from $\tau = 0$ to $\tau = \beta$, where $\tau = -it$ is imaginary time. Given a full configuration at each time slice, we can define the classical configuration as

$$\mathbf{C} \equiv (\boldsymbol{\sigma}_1^z, \ldots \boldsymbol{\sigma}_{2m}^z)\,, \tag{1.30}$$

and re-write the partition function as

$$Z = \sum_{\mathbf{C}} W(\mathbf{C})\,. \tag{1.31}$$

Referring to Fig. 1.6, the $1d$ quantum system is represented as a $2d$ system, where the $(d+1)$ configurations give rise to a set of *world-lines* with periodic boundary conditions imposed by the trace in the partition function.

Now that we have establish a mapping to a new classical system that can be simulated with MC, we need to evaluate the weights $W(\mathbf{C})$. This consists of computing the matrix elements $\langle\boldsymbol{\sigma}_t^z|e^{-\Delta\tau\hat{H}_A}|\boldsymbol{\sigma}_t^z\rangle$ for each propagator at every time-slice (similar for sub-lattice B). Since each sub-lattice Hamiltonian is the sum of commuting terms,

$$\langle\boldsymbol{\sigma}_t^z|e^{-\Delta\tau\hat{H}_A}|\boldsymbol{\sigma}_t^z\rangle = \prod_{j\in\mathrm{odd}}\langle\boldsymbol{\sigma}_t^z|e^{-\Delta\tau\hat{\boldsymbol{S}}_j\cdot\hat{\boldsymbol{S}}_{j+1}}|\boldsymbol{\sigma}_t^z\rangle\,, \tag{1.32}$$

the calculation simply reduces to a two-body problem, which we already cover earlier in

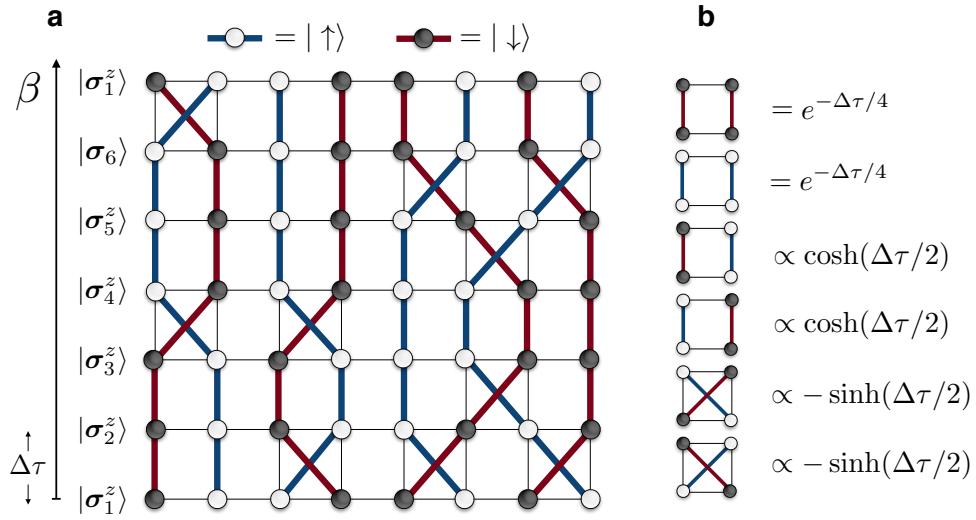

Figure 1.6: **Imaginary-time path integrals**. We show a set of world-lines in imaginary time, from the path integral expansion of the partition function. The world-lines correspond to either spin up (blue) or spin down (red), with periodic boundary conditions in imaginary time. **a)** We show an example for the $1d$ Heisenberg model with $N = 8$ spins and $m = 3$ Trotter steps, for a total of $2m = 6$ time slices. **b)** We report the plaquette amplitude (i.e. the matrix element of the imaginary-time propagator) for each of the six allowed transitions.

this Chapter. For a $N = 2$ system, of all the 16 possible matrix elements, only six are different from zero [36], and are given in Fig. 1.6. At a given time slice τ_k, the two-body propagators couple two spins $\boldsymbol{\sigma}_{\tau_k}^z$ with two spins $\boldsymbol{\sigma}_{\tau_k+\Delta\tau}^z$ at the next time slice. Because the Hamiltonian conserves the total spin angular momentum, only matrix elements that conserve the number of spins in a given plaquette between two sequential time slices are permitted. The value for the weight $W(\mathbf{C})$ for the full configuration is given by multiplying together all the plaquette contributions in the path integral expansion.

We can finally look at how to compute expectation values. For the energy, for example, we can obtain the expectation value as

$$U = \langle \hat{H} \rangle = \frac{\sum_{\mathbf{C}} W(\mathbf{C}) E(\mathbf{C})}{\sum_{\mathbf{C}} W(\mathbf{C})} \,, \tag{1.33}$$

where the new energy function, after being average over the time slices, is given by [36]

$$E(\mathbf{C}) = -\frac{1}{m} \frac{\partial}{\partial \Delta\tau} \log W(\mathbf{C}) \,. \tag{1.34}$$

As usual with MC, the expectation value is approximated by a Markov chain as

$$\langle \hat{H} \rangle = \frac{1}{n} \sum_{k=1}^{n} E(\mathbf{C}^{(k)}) \tag{1.35}$$

where the configurations $\mathbf{C}^{(k)}$ are sampled according to the distribution $W(\mathbf{C}^{(k)})$. Contrary to classical MC, a different breed of sampling algorithms needs to be used in the QMC framework. For instance, local single spin-flip updates (the simplest updates in MC), are not allowed for the Heisenberg model as they would break world-lines, and thus the conservation of total spin angular momentum $\hat{S}^z_T$. The simplest update for this specific case consists of locally creating or moving pairs of kinks in the world-lines. However, this type of sampling is generally non-ergodic, and more sophisticated cluster updates are required, such as the the worm update or directed loops [37, 38]. For ground state calculations, a sufficiently large β must also be used in the simulation, leading to a larger number of time slices, and thus an increase in computational demand. While the above path integral is instructive in understanding the core of QMC, better algorithms are generally used to perform large-scale simulations of both finite and zero temperature properties. For instance, the continuum limit can be taken explicitly, avoiding the systematic error in the Trotter decomposition. Moreover, there are several other techniques, such as the *stochastic series expansion* [39] *determinantal* QMC [40] and *variational Monte Carlo* (VMC) [41]. Any further discussion goes however beyond the scope of this Section.

1.3.2 The sign problem

The QMC framework is based on the fundamental assumption that the weights obtained for the configurations in imaginary-time are positive-definite, $W(\mathbf{C}) \geq 0$. In turn, the presence of a non-trivial *sign structure* (i.e. a pattern of different signs in the weights) disables the probabilistic interpretation of the path integral. We proceed now to give an overview of this *sign problem*, which represents the major limitation of QMC methods.

The sign problem manifests differently in different scenarios, so we will start by considering the path integral expansion. For the $1d$ Heisenberg model, the matrix elements of the propagation operator for spin-exchanges are negative:

$$\langle \uparrow\downarrow | e^{-\Delta\tau \hat{H}_{A/B}} | \uparrow\downarrow \rangle = \langle \downarrow\uparrow | e^{-\Delta\tau \hat{H}_{A/B}} | \uparrow\downarrow \rangle = -e^{\Delta\tau/4} \sinh(\Delta\tau/2) \, . \tag{1.36}$$

In principle, this minus sign can contribute to an overall negative weight $W(\mathbf{x}) < 0$. In practice this never happens, and the reason is the particular connectivity of the Hamiltonian. In fact, the interaction terms couple sites that belong to two different sub-lattices. In this case, the lattice is bipartite, and thus the total number of spin exchanges for a world-lines configuration can only be even, leading always to a positive weight $W(\mathbf{x}) > 0$. Instead, for non-bipartite lattices the world-lines can assume configurations with an odd number of spin exchanges, leading in turn to negative weights. This is the case, for example, for the Heisenberg model in the triangular lattice [42, 43], or the J1-J2 model on the square lattice [44]. The former can be extended to (quasi) one dimension by considering a spin ladder with Hamiltonian

$$\hat{H} = J_1 \sum_{j=1}^{N-1} \hat{\boldsymbol{S}}_j \cdot \hat{\boldsymbol{S}}_{j+1} + J_2 \sum_{j=1}^{N-2} \hat{\boldsymbol{S}}_j \cdot \hat{\boldsymbol{S}}_{j+2} \, , \tag{1.37}$$

and competing interactions J2 > 0. In the path integral, the presence of spin exchanges between next-to-nearest neighbours leads to the possibility of an odd number of negative amplitudes, and thus to an overall negative weight (Fig. 1.7a). This holds for any model with frustration, which cannot be in general simulated using QMC.

Aside from frustrated quantum spin models, there is another important class of system that suffers the sign problem. Consider now a lattice (e.g. in one dimension) with L sites and N non-interacting particles. The kinetic Hamiltonian

$$\hat{H} = \sum_{j=1}^{L-1} \hat{c}_j^\dagger \hat{c}_{j+1} + \text{ h.c.} \tag{1.38}$$

contains the creation $\hat{c}_j^\dagger$ and annihilation $\hat{c}_j$ operators of a particle at site j. The obvious choice for the Hilbert space basis is the occupation number representation $|\boldsymbol{n}\rangle = |n_1, \ldots, n_L\rangle$, with $n_j = \langle \hat{c}_j^\dagger \hat{c}_j \rangle$. Since the particles are not interacting, the ground state properties can be found by a Fourier transform into momentum space, where the Hamiltonian becomes diagonal (for any dimension). For bosonic particles, the ground state is obtained by placing all the N bosons in the lowest energy state at momentum $k = 0$:

$$|\psi_0^{\mathrm{B}}\rangle = \prod_{i=1}^{N} \hat{c}_{k=0}^\dagger |0\rangle \, . \tag{1.39}$$

Clearly, all the wavefunction coefficients appear with a positive sign,

$$\psi_0^{\mathrm{B}}(\boldsymbol{n}) = \langle n_1, n_2, \ldots, n_L | \psi_0^{\mathrm{B}} \rangle > 0 \, , \tag{1.40}$$

and thus there is no sign problem in this case. This extend to the case of interacting bosons (Bose-Hubbard model [45]). On the other hand, if the particles obey Fermi statistics, the ground state is given by a Slater determinant of single particle states, which can be negative due to the anti-symmetry of fermionic states. The presence of a sign structure in the real-space wavefunction generates in turn a sign problem in the corresponding path integral representation. In this case, the negative contribution to the total weight $W(\mathbf{C})$ appears when two fermions are exchanged an odd number of times, resulting from the fermionic anti-commutation rules. We show in Fig. 1.7b a typical world-line configuration for $N = 2$ fermions on a $1d$ lattice with $L = 6$ sites. Between the time slice $\tau = 4$ and $\tau = 5$ the two fermions are exchanged with a global π phase shift, generating a minus sign in the weight. We can see that the presence of a sign structure prevents QMC from

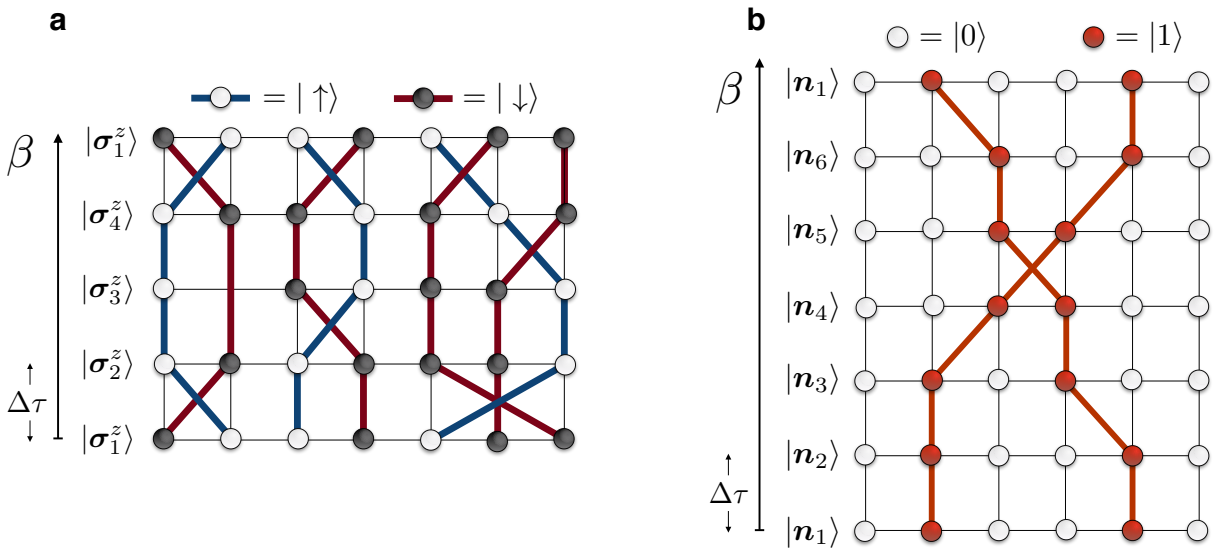

Figure 1.7: **Sign problem in imaginary-time path integrals**. We show world-lines configurations that lead to an overall negative sign for two different models. **a)** For the case of a frustrated quantum magnet, such as the J1-J2 model, due to the hopping term between next-to-nearest neighbours, there are now an odd number of spin exchanges, with a resulting negative weight. In this particular case, the sign problem is generated by the lattice being non-bipartite. **b)** An example of path integral configuration for two non-interacting fermions on a $1d$ lattice. The overall negative sign is generated by the anti-commutation rules when the two fermions are exchanged.

simulating fermionic systems, even without any interaction [6]. While the properties of non-interacting (or weakly-interacting) fermions are very well understood from analytical calculations (since the corresponding field theory is Gaussian), strongly-interacting fermions in higher-dimensions remains untouched territory for both field-theoretic techniques as well as numerical methods.

In the search of a solution to the sign problem, a first attempt consists of sampling the world-lines according to the absolute value of the weights $|W(\mathbf{C})|$. While this generates samples with the correct probability, it prevents to accurately estimate any physical

[6] The fermion sign problem can be in practice circumvented by QMC only for $1d$ systems.

observables. Within this assumption, the expectation value of the energy becomes

$$\begin{aligned}\langle \hat{H} \rangle &= \frac{\sum_{\mathbf{C}} |W(\mathbf{C})| E(\mathbf{C})}{\sum_{\mathbf{C}} |W(\mathbf{C})|} = \frac{\sum_{\mathbf{C}} |W(\mathbf{C})| \mathrm{Sign}\big[W(\mathbf{C})\big] E(\mathbf{C})}{\sum_{\mathbf{C}} |W(\mathbf{C})| \mathrm{Sign}\big[W(\mathbf{C})\big]} \\ &= \frac{\langle \mathrm{Sign} \cdot E \rangle_{|W|}}{\langle \mathrm{Sign} \rangle_{|W|}} \,.\end{aligned} \tag{1.41}$$

It can be shown that the average sign scales as [46]

$$\langle \mathrm{Sign} \rangle_{|W|} \sim e^{-\beta N \Delta F} \tag{1.42}$$

where ΔF is the free energy difference between the original system and the one obtained by considering the absolute value. For the specific case of an anti-ferromagnetic spin system, this corresponds to the difference in free energy from its ferromagnetic counterpart. As the system size grows, the average of the sign becomes exponentially small, leading to very large statistical uncertainty on the average energy $\langle \hat{H} \rangle$. After all, why should we hope to extract the physics of anti-ferromagnetism by simulating a spin model with ferromagnetic interactions?

In general, it is argued that a generic solution to the sign problem in QMC belongs to the complexity class of NP-hard problems [7] [46]. This result does not however imply that specific solutions cannot be discovered. One notable example is the calculation of correlation functions for a resonating-valence-bond wavefunctions on a triangular and kagome lattice, where the sign problem was solve by using a Pfiaffian representation of the quantum state [47]. In fact, an appropriate transformation on the system could be in general capable of removing the sign problem from the model Hamiltonian (or from a specific quantum state). In this spirit, let us go beyond the path integral representation in QMC, and back to the quantum states. As usual, we focus on the ground state $|\psi_0\rangle$ of some model Hamiltonian $\hat{H}$. In general, for any physical model, there is a preferential or natural basis for the Hilbert space $\{|\mathbf{x}\rangle\}$. We can then say that the model has a sign problem if there is a sign structure (or phase structure, for complex wavefunctions) in the

[7] Problems that cannot be solved in polynomial time by a classical computer.

coefficients [8] $\psi_0(\mathbf{x})$. The presence or lack of a non-trivial sign structure can be traced back to a specific property of the Hamiltonian. In particular, the wavefunction is positive ($\psi(\mathbf{x}) \geq 0$), only if all off-diagonal elements of the Hamiltonian are negative, $\langle \mathbf{x}|\hat{H}|\mathbf{x}'\rangle \leq 0$, $\forall \mathbf{x} \neq \mathbf{x}'$ [48, 49]. Intuitively, it would seem that a positive quantum state can be captured better (i.e. it is less complex) than a quantum state with a non-trivial pattern of signs or complex phases. In general in fact, a positive state can be described with a reduced number of parameters, and more importantly, it has an interpretation as a classical probabilistic model [9]. However, since the presence or not of a sign structure depends on the off-diagonal elements of the Hamiltonian, such definition of the sign problem would be basis-dependent [50], and as such not particularly meaningful. In fact, there could be a basis $\{|\mathbf{y}\rangle\}$ where the off-diagonal elements are all negative, and thus the ground state wavefunction is sign problem-free. In a more precise way, there might exist a canonical transformation $\hat{\mathcal{U}}$ such that $\langle \mathbf{y}|\hat{\mathcal{U}}\hat{H}\hat{\mathcal{U}}^\dagger|\mathbf{y}'\rangle \leq 0$ and $\psi_0(\mathbf{y}) = \hat{U}\psi_0(\mathbf{x}) \geq 0\ \forall|\mathbf{y}\rangle$. The obvious solution is the basis $\{|\boldsymbol{\phi}\rangle\}$ of the eigenstates of the Hamiltonian. Then, clearly we have $\langle \boldsymbol{\phi}|\hat{H}|\boldsymbol{\phi}'\rangle = 0\ \forall \boldsymbol{\phi} \neq \boldsymbol{\phi}'$, but finding this basis is equivalent to solving the problem, and thus it is exponentially hard. Given all of the above, we can imagine a classification of complexity of quantum states based on positivity of the state, and if not, on how difficult it is to find the appropriate canonical transformation that removes the signs, as well as how feasible it is to implement such transformation.

We now give a simple example, and consider again the Heisenberg model for only two spins. The ground state is the single state

$$|\psi_0\rangle = \frac{1}{\sqrt{2}}\Big(|\uparrow\downarrow\rangle - |\downarrow\uparrow\rangle\Big)\,, \tag{1.43}$$

with a non-trivial sign structure. The sign however, can be obtained with the Marshall-Peierls sign rule due to the bipartite nature of the lattice. This prescription also provides us with a way to remove the sign structure. For two spins, this simply consists on applying the spin rotation around the z axis [10] $\hat{U} = e^{i\pi\hat{\sigma}^z_2/2}$. This transformation leaves invariant the

[8] With sign structure we refer to the wavefunction coefficients $\psi_0(\mathbf{x})$ being both positive and negative.
[9] This statement will become clear by the end of Chapter 3.
[10] The transformation can be equivalently applied to the other spin.

longitudinal interaction term $\hat{S}_1^z \hat{S}_2^z$ and flips the signs of the spin-exchange terms, leading to the Hamiltonian

$$\hat{H}' = \hat{U}\hat{H}'\hat{U}^\dagger = \hat{S}_1^z \hat{S}_2^z - \frac{1}{2}\Big(\hat{S}_1^+ \hat{S}_2^- + \hat{S}_1^- \hat{S}_2^+\Big) \tag{1.44}$$

with negative off-diagonal matrix elements. In fact, if we now diagonalize the new Hamiltonian $\hat{H}'$, we find the ground state

$$|\psi_0'\rangle = \frac{1}{\sqrt{2}}\Big(|\uparrow\downarrow\rangle + |\downarrow\uparrow\rangle\Big) = \hat{U}|\psi_0\rangle \tag{1.45}$$

without any sign structure [11]. This result can be easily extended to a system with N spins. Now, the rotations around the z-axis are applied to all the spins of one of the two sub-lattices, with resulting canonical transformation $\hat{U} = \bigotimes_{j\in\text{even}} e^{i\pi\hat{\sigma}_j^z/2}$ [12]. The full unitary transformation is pictured in Fig 1.8a, using a TN diagram. The ground state in the reference basis $\{|\boldsymbol{\sigma}^z\rangle\}$, expressed as a N-rank tensor, is transformed according to $\hat{U}$, consisting of $N/2$ local rotations (here rank-2 tensors). Note that for this case, the *depth* of this TN diagram is equal to unity and independent of N.

The Marshall sign rule holds as long as the lattice is bipartite. If, for example, we re-introduce the next-to-nearest neighbours interactions, we observe the breakdown of the transformation, as the competing interaction J2 > 0 grows large. This is shown in Fig 1.8c, where we plot the average sign of the ground state wavefunction,

$$\langle\text{Sign}\rangle = \sum_{\boldsymbol{\sigma}^b} |\psi_0(\boldsymbol{\sigma}^b)|^2 \text{Sign}[\psi_0(\boldsymbol{\sigma}^b)] \tag{1.46}$$

calculated using ED after the canonical transformation of the Marshall sign rule is applied. While for small J2 the average sign remains unity, as J2 becomes larger the average sign decreases towards zero with a stronger decay for larger system sizes, showing the break-

[11] The ground state energy $E_0 = -3/4$ is clearly conserved by the transformation, as all the other physical properties (observables).

[12] Note that this unitary transformation also changes the signs of the spin-exchange matrix elements in the path integral from negative to positive. For the specific case, the transformation is then irrelevant, as only an even number of spin exchanges can occur in the path integral.

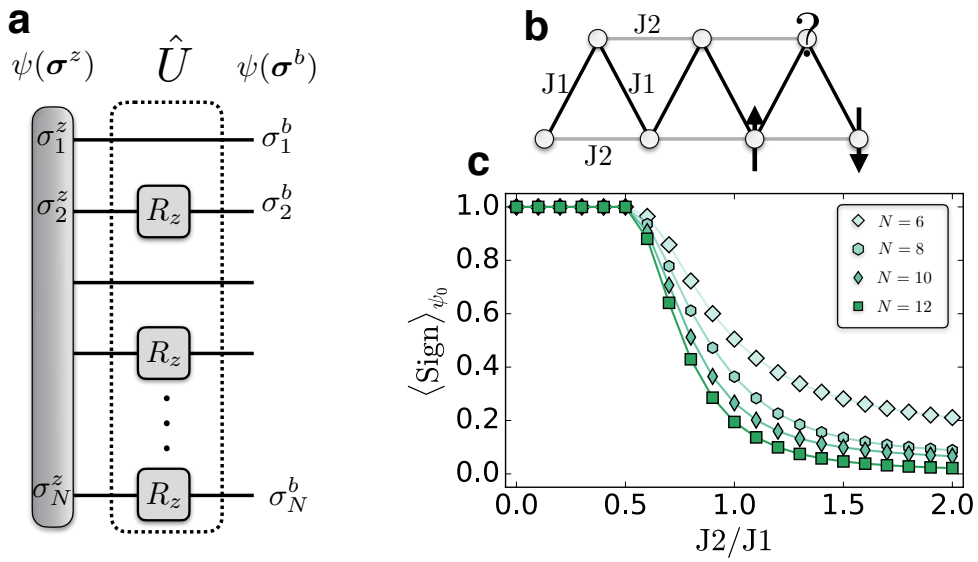

Figure 1.8: **Breakdown of the Marshall sign rule.** **a**) Canonical transformation associated with the Marshall sign rule. A rotation is applied to each spin of one of the two sub-lattices, with a resulting positive wavefunction $\psi(\boldsymbol{\sigma}^b) > 0$. **b**) J1-J2 model, where competing interactions lead to frustration. **c**) Average sign of the ground state wavefunction of the J1-J2 model, after the Marshall signe rule transformation (in **a**) has been applied to the Hamiltonian (before carrying out its ED).

down of the bipartite properties of the state. As such, QMC can be implemented on the transformed Hamiltonian $\hat{H}' = \hat{U}\hat{H}\hat{U}^\dagger$ as long as the average sign remains close to unity. If instead we want to explore the regime of large J2, we should find a new canonical transformation $\hat{U}_{\mathrm{J2}}$, which keeps the sign of the ground state wavefunction arbitrarily close to one for any value of J2. Moreover, to be useful, the depth of the circuit implementing such canonical transformation, once broken down in terms of k-local unitaries (i.e. acting locally on at most k spins), should not scale exponentially with N. This is in general the case for finding the basis of the energy eigenstates. While the best case scenario is a constant depth circuit (e.g. the Marshall sign rule), a polynomial scaling can likely be accommodated. However, proving that such a short-depth transformation exists, and actually finding the unitary circuit, remains an open challenge in quantum many-body physics and Hamiltonian complexity theory.

1.4 Quantum simulation

We have been navigating through the most successful techniques to simulate quantum mechanics on classical hardware. While on one hand, efficient algorithms and large computational power allowed the solution of many classes of quantum many-body problems, on the other, there are still many model Hamiltonians out of the reach of classical simulation, leaving open questions in condensed matter physics. It is in fact unlikely that classical hardware will ever fully overcome the complexity of the simulation of quantum mechanics, the sign problem being one example. Yet, nature continuously solves the quantum many-body problem in any real-world sample of quantum matter. This suggests a different path to solving the problem. As proposed by Feynman in his seminal paper [51], the large complexity of many-body quantum states can be captured efficiently by using quantum mechanics itself. In other words, instead of a simulating quantum mechanics with a classical computer, one should instead operate with a machine where the elementary degrees of freedom are described by quantum mechanics, i.e. a *quantum computer*.

Since naturally any quantum hardware functions in real time, we will consider for the problem at hand the time-dependent Schrödinger equation, whose solution (for a time-independent Hamiltonian) is given by

$$\psi(\boldsymbol{r}, t) = \hat{U}(t)\psi(\boldsymbol{r}, 0) = e^{-\frac{i}{\hbar}\hat{H}t}\psi(\boldsymbol{r}, 0)\,. \tag{1.47}$$

Here, the set of quantum degrees of freedom $\{|\boldsymbol{r}\rangle\}$ is again assumed to be binary. As we are interested in local Hamiltonians, we will also assume $\hat{H} = \sum_k \hat{H}_k$, with $\hat{H}_k$ acting locally on a small number of degrees of freedom. The goal of a *quantum simulation* is to implement a different and controllable quantum system with quantum numbers $\{|\mathbf{x}\rangle\}$ to simulate or emulate the original system undergoing dynamics with the physical Hamiltonian $\hat{H}$. In

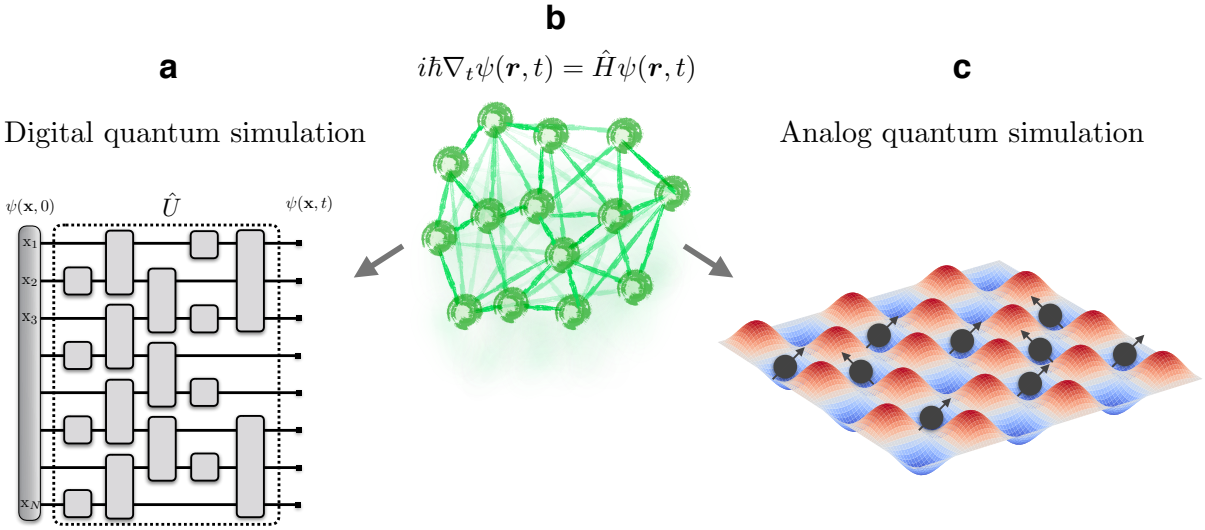

Figure 1.9: **Quantum simulation.** The properties of a generic physical quantum system governed by the Schroedinger equation (**b**) can be simulated using quantum mechanics in two different ways. Digital quantum simulation (**a**) maps the specific problem into a quantum circuit acting on a set of quantum bits. Instead, an analog quantum simulation (**c**), the system is simulated using another physical system whose interactions can be engineered to emulate the physics of the original system of interest.

other words, given the mapping $|\boldsymbol{r}\rangle \to |\mathbf{x}\rangle$ and assuming we can prepare the initial state $\phi_0(\mathbf{x}) = \psi(\boldsymbol{r}, 0)$ of the simulator, we wish to implement the propagator $\hat{\mathcal{U}}_t$ such that

$$\hat{\mathcal{U}}_t\phi_0(\mathbf{x}) = \phi_t(\mathbf{x}) = \psi(\boldsymbol{r}, t)\,. \tag{1.48}$$

The solution of quantum simulation, i.e. the state $\phi_t(\mathbf{x})$, can be obtained with two different approaches. In an analog quantum simulation, the dynamics is emulated with a different physical system, where the interactions are engineered to mimic the original system. One example are cold atoms in optical lattices. In a digital quantum simulation, the physical degrees of freedom are mapped into quantum bits of information and processed by a quantum hardware. In this latter case, the possibility to re-program the hardware to perform arbitrary computations gives the machine the name of *universal quantum simulator*.

1.4.1 Universal quantum simulators

The feasibility of universal quantum simulator, first introduced by Feynman, was later proven by Seth Lloyd [52] in the context of the simulation of local Hamiltonians. Instead of classical bits, such machine is made up of *qubits*, quantum degrees of freedom living in the $2d$ Hilbert space spanned by the *computational basis* $\{|0\rangle, |1\rangle\}$. Depending on the system, one needs a different mapping between the physical Hilbert space $\{|\boldsymbol{r}\rangle\}$ and the computational basis states $|\mathbf{x}\rangle \in \{|0\rangle, |1\rangle\}^{\otimes N}$. For example, for spin-$\frac{1}{2}$ with basis states $|\boldsymbol{\sigma}^z\rangle$, the mapping is simply given by $\sigma_j^z = 1 - 2\mathrm{x}_j$. For higher-dimensional local Hilbert spaces, either a larger-dimensional computational basis or a larger number of qubits is required to map the problem into the quantum computer. Then, given the specific mapping, the first step is state preparation, that is the capability of preparing the qubits in the state

$$|\phi_0\rangle = \sum_{\mathbf{x}} \phi_0(\mathbf{x})|\mathbf{x}\rangle \,, \tag{1.49}$$

with $\phi_0(\mathbf{x}) = \psi(\boldsymbol{r}, 0)$. We assume state preparation to be feasible, and proceed to discretize the real-time evolution operator, assuming a local Hamiltonian. Because in general the full dynamics cannot be broken down into individual local propagations

$$\hat{U}(t) \neq \prod_k \hat{U}_k(t) = e^{-i\hat{H}_k t} \,, \tag{1.50}$$

the time-evolution operator is then approximated via the Trotter formula

$$\hat{U}(t) = \left(e^{-i\hat{H}\Delta t}\right)^{t/\Delta t} = \left(e^{-i\sum_k \hat{H}_k \Delta t}\right)^{t/\Delta t} \,, \tag{1.51}$$

where to propagator for time Δt is written down as

$$e^{-i\sum_k \hat{H}_k \Delta t} \simeq \prod_k e^{-i\hat{H}_k \Delta t} \tag{1.52}$$

with systematic error $O(\Delta t^2)$. Therefore, the full dynamics can be simulated with a quantum hardware using a sequence of $m = t/\Delta t$ time slices, with each slice consisting of a time evolution through a set of local propagators $e^{-i\hat{H}_k \Delta t}$. Now we need to see how such operation can be implemented into the hardware.

A powerful property of a universal quantum simulator, is that any unitary quantum operation on the qubits (such as quantum dynamics) can be compiled into a set of elementary quantum operations, called a universal set of *quantum gates*. The minimum requirements are arbitrary single qubit rotations

$$\mathrm{R}_\alpha(\theta) = e^{-i\hat{\sigma}^\alpha \theta/2}\,, \tag{1.53}$$

where $\alpha = x, y, z$, and two-qubits entangling gates, such as the controlled-NOT gate

$$\mathrm{C}_X = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}\,. \tag{1.54}$$

which flips the state of a target qubit, depending on the state of a control qubit. Similarly to the universality of the NAND operation for classical computing, given a universal set of quantum gates, any arbitrary quantum computation can be realized. In the context of quantum simulation, this implies that such a machine can simulate the time evolution of an arbitrary physical system with local Hamiltonian. Finally, the ultimate step in the quantum simulation is the read-out. We require to be capable of performing measurements with high fidelity, which can then be used to extract physical properties of interest. We will discuss in more details the measurement step in Chapter 4.

Adiabatic state preparation

So far we have been discussing the quantum simulation of the time-dependent Schrödinger equation, obtaining the dynamics generated by an Hamiltonian. If we are in turn interested in static properties, such as the ground state, we need to find an alternative way to perform the simulation. In contrast with QMC and projectors methods, where the ground state is obtained from an expansion in imaginary time, a quantum simulation is restricted to real time only. The static properties can be however obtained, within some accuracy, using an adiabatic evolution. If we prepare the system into the ground state of $\hat{H}_0$, and slowly change the Hamiltonian in time, the resulting dynamics will force the quantum state to remain close to the ground state of the instantaneous Hamiltonian [53]. As such, if we want to adiabatically prepare the ground state of the Heisenberg model, for example, we could proceed as follows. We initialize at $t = 0$ the qubits in the hardware to the anti-ferromagnetic state

$$|\psi(\boldsymbol{\sigma}^z, t = 0)\rangle = |\uparrow\downarrow\uparrow \ldots \downarrow\rangle \,, \tag{1.55}$$

which is a ground state of the Hamiltonian $\hat{H}_0 = \sum_j \hat{S}^z_j \hat{S}^z_{j+1}$. If want to reach the ground state of the Heisenberg model after time $t = t^*$, the Hamiltonian is changed in time as

$$\hat{H}(t) = \Big(1 - \frac{t}{t*}\Big)\hat{H}_0 + \frac{t}{t^*}\hat{H} = \sum_j \Big[\hat{S}^z_j \hat{S}^z_{j+1} + \frac{t}{2t^*}\Big(\hat{S}^+_j \hat{S}^-_{j+1} + \text{ c.c.}\Big)\Big] \,. \tag{1.56}$$

As long as the evolution is carried out sufficiently slow, the quantum state $|\psi(\boldsymbol{\sigma}^z, t)\rangle$ will remain an eigenstate of the Hamiltonian $\hat{H}(t)$, obtaining for (large) t^* the desired ground state of the Heisenberg Hamiltonian $\hat{H}$. In practice, the time t^* to adiabatically prepare the ground state needs to be larger that the time-scale set by the energy gap $\Delta_0 = E_1(t) - E_0(t)$ between the ground and first excited state [53].

1.4.2 Digital quantum simulation with superconducting qubits

To give a more concrete example, we now show how to design and perform a quantum simulation of a spin Hamiltonian on a real quantum hardware. For practical reason, we consider the $1d$ *transverse-field Ising model* (TFIM), where spin-$\frac{1}{2}$ interact with Hamiltonian

$$\hat{H} = -\sum_{j=1}^{N-1} \hat{\sigma}_j^z \hat{\sigma}_{j+1}^z - h \sum_{j=1}^{N} \hat{\sigma}_j^x \tag{1.57}$$

and h is the strength of the magnetic field. We will discuss this model further in Chapter 4. For the sake of this experiment, let us just consider the dynamics generated by the propagator $\hat{U}(t) = e^{-i\hat{H}t}$ for some initial state $\psi(\boldsymbol{\sigma}^z, 0)$. As initial state, we choose the fully polarized state

$$|\psi(\boldsymbol{\sigma}^z, 0)\rangle = |\uparrow\uparrow\uparrow\uparrow\ldots\uparrow\rangle \tag{1.58}$$

and drive the time evolution with the Hamiltonian with magnetic field $h = 1$. This is called a *quantum quench*, i.e. the time evolution caused by an abrupt change in the Hamiltonians parameters. In this case, the initial state is a ground state of the TFIM with $h = 0$ magnetic field. At time $t = 0$ we switch on the transverse field $h = 1$ and let the system evolve in time. Since $\psi(\boldsymbol{\sigma}^z, 0)$ is effectively a high energy state compared to the ground state $\psi_0(\boldsymbol{\sigma}^z)$ of $\hat{H}$ (with $h = 1$), the dynamics induces the creation of quasi-particles excitation along the chain, responsible for the increase in entanglement entropy [54] and thus a non-trivial dynamics. Given a fixed Trotter step Δt, the time evolution operator is approximate as

$$\hat{U}(\Delta t) = e^{-i\hat{H}\Delta t} \simeq \prod_j \hat{U}_j^z(\Delta t) \prod_j \hat{U}_j^x(\Delta t)\,, \tag{1.59}$$

where $\hat{U}_j^z(\Delta t) = e^{i\Delta t \hat{\sigma}_j^z \hat{\sigma}_{j+1}^z}$ and $\hat{U}_j^x(\Delta t) = e^{i\Delta t \hat{\sigma}_j^x}$ (Fig. 1.10a).

We wish now to use a digital quantum simulators to implement the quantum dynamics $\hat{U}(t)$. First of all, we need to compile the unitary operations into a set of elementary universal quantum gates. Let us start with the interaction $\hat{\sigma}_j^x$ between the j-th spin and the magnetic field. We can easily see that the propagator $\hat{U}_j^x(\Delta t) = e^{i\Delta t \hat{\sigma}_j^x}$ corresponds to a

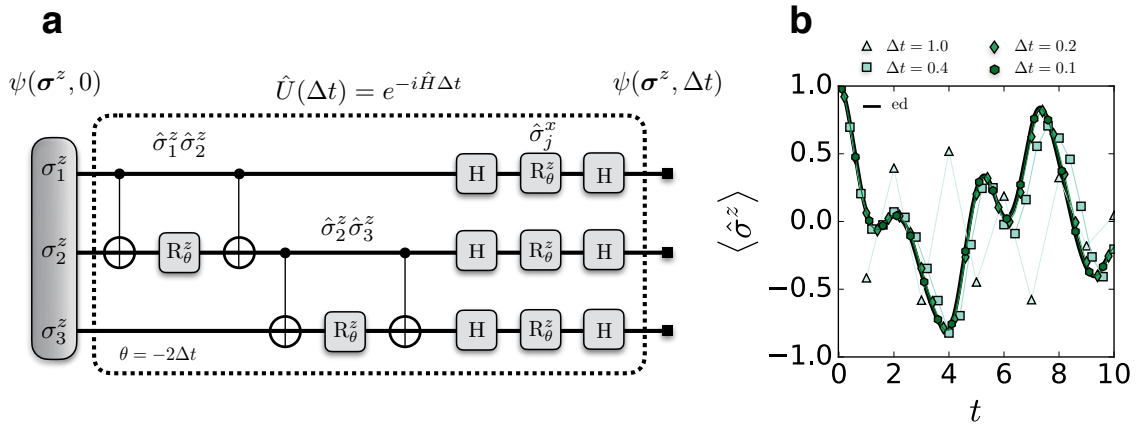

Figure 1.10: **Digital quantum simulation. a)** Quantum circuit implementing one Trotter step of time evolution for the TFIM with $N = 3$ spins. The Ising interactions are compiled as controlled-NOT and single qubit z rotations, while the interaction with the magnetic field (a x rotation) is compiled as a z rotation between two Hadamard gates. The latter can be parallelized easily, its depth being constant with respect to the number of spins. **b)** Real time evolution with an initial fully polarized state, generated by the TFIM Hamiltonian at the critical point. The black line is obtained with an ED simulation of the Schrödinger equation, while the markers are the result of a classical simulation of the quantum circuit, with an Trotter step increasing from $\Delta t = 0.1$ to $\Delta t = 1.0$.

unitary single qubit rotation $\mathrm{R}_x(\theta)$ around the x axis of an angle $\theta = -2\Delta t$. For practical implementation, this rotation is broken up further as

$$\hat{U}_j^x(\Delta t) = \mathrm{R}_x(-2\Delta t) = \mathrm{H}\,\mathrm{R}_x(-2\Delta t)\,\mathrm{H} \tag{1.60}$$

where we omitted a global phase factor $e^{2i\Delta t}$ and we introduced the Hadamard gate

$$\mathrm{H} = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}. \tag{1.61}$$

The other propagator $\hat{U}_j^z(\Delta t) = e^{i\Delta t\hat{\sigma}_j^z\hat{\sigma}_{j+1}^z}$ acts on a pair of neighbouring qubits, and can be realized using a combination of two controlled-NOT gates and a rotation around the z axis of an angle $\theta = -2\Delta t$. As such, one Trotter step requires a total of $6N - 3$ gates.

Superconducting quantum hardware

At last, we proceed to show an actual quantum simulation run on a superconducting quantum hardware. As of now, superconducting hardware has revealed to be the most powerful platform for quantum computation and simulation. One reason is the large flexibility in the qubits manufacturing, which can be configured in various way through the technique of electron-beam lithography. While in principle this technology is highly scalable, in practice the small coherence times of the hardware limits (for now) its capabilities to a relatively small number of qubits. The different types of superconducting qubits, such as *flux* and *charge* qubits, mostly rely on the same principles. First, there needs to be a well defined $2d$ sub-space encoding the computational basis. The logical $|0\rangle$ state is provided by the ground state of a Cooper pair formed by two electrons, and the logical $|1\rangle$ is simply the first excited state. Further, in order to avoid transitions out of the computational basis, the energy spectrum of the qubit must be made anharmonic, usually done with a Josephson junction [55]. Once the local Hilbert space is defined, operations on the qubits are performed using pulses of microwave radiation, as well as external magnetic fields. A review of quantum simulation with superconducting quantum hardware is given in [56].

We implement the quantum simulation on the IBM Q 5 Tenerife (ibmxq4 chip), shown in Fig. 1.11a, and built with 5 *transmon* qubits [57]. This type of quantum hardware has been already applied for the variational optimization of the ground state of small molecules and quantum magnets [58, 59, 60]. As a demonstration, we implemented a quantum simulation for the TFIM using only two out of the five qubits available. After initializing the hardware to the state $|00\rangle$, we apply the quantum circuit in Fig 1.10a with different Trotter step Δt and up to a total time $t_f = 3.0$. At the end of the circuit, we performed a projective measurement on the computational basis, which outputs the number of counts for the different configurations of the computational basis. We show the results in Fig. 1.11b, where we plot the average longitudinal magnetization as a function of time. We compare the results from the real quantum simulation, with the ED solution (black line) and the full circuit (classical) simulation carried out with $\Delta t = 0.1$ (green markers). We can see that up to $t \simeq 0.5 - 0.7$ all simulations performed with the ibmqx4

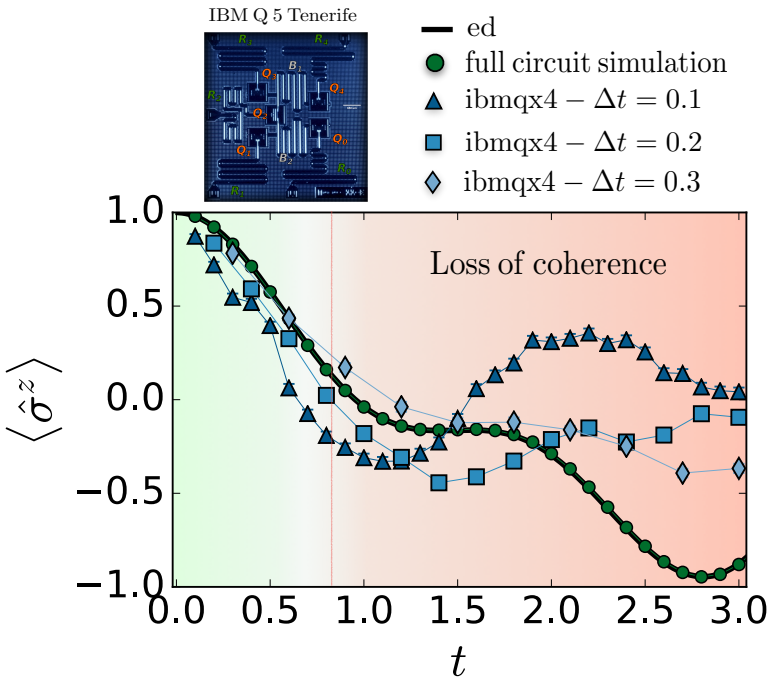

Figure 1.11: **Quantum simulation with superconducting hardware**. Quantum simulation of the time evolution of the TFIM for a quantum quench from $h = 0$ to $h = 1$. We compare the results from a simulation of the results from the ibmqx4 hardware, with results from ED (black) and classical circuit simulation (green).

are fairly close to the correct value, while we observe substantial deviations as the time increases. More interestingly, the simulation with smallest step $\Delta t = 0.1$ appears to deviate the most from the correct dynamics. This is expected since by decreasing the size of the Trotter step, and given a fixed time $t = 3.0$, the number of gates increases. For instance, for $\Delta t = 0.1$ the dynamics is compiled with 270 gates, while $\Delta t = 0.3$ only requires 90. At the time of the simulation, for the two qubits used in the simulation, the single qubit gate errors were 0.12% and 0.163%, the readout errors were 6.6% and 5.2%, while the controlled-NOT gate had an error of 3.9%. As such, the accuracy of the simulation is not restricted by the size of the Trotter step (with error $\sim O(\Delta t^2)$), but rather from the errors in the hardware. In particular, as the time evolution goes, errors can build up and eventually lead to a complete loss of coherence.

Decoherence

The current quantum technology paradigm has been name the *noisy intermediate-scale quantum hardware* era [61]. Within the past 2-3 years we have witness an important advancement in qubits manufacturing and their coherent control. However, in order to perform quantum simulation of even a moderate system size, a large number of quantum gates are required. The same holds for mostly any practical application of any quantum algorithm. During the quantum computation time, the hardware should be in principle completely isolated with the outside world. In fact, any interaction with the surrounding environment can degrade the quantum coherence between the qubits, and irreversibly destroy the quantum computation/simulation. Quoting John Preskill:

"Classically there would be nothing wrong with looking at every time step, what the state of the hardware is. That wouldn't prevent the computer from getting the right answer. Quantumly, if we keep looking at the computer, that will destroy these delicate superpositions. It's a secret computation..." [62].

So in practice, the hardware should be completely isolated, but at the same time it needs to be accurately controlled from the outside. Therefore, as for classical computing, one should develop a protocol to recover from the errors before they build up and completely corrupt the quantum state of the hardware. We will discuss more about *quantum error correction* in Chapter 5. Regardless, the current generation of quantum hardware is not yet capable of being error-prone, and thus the name *noisy*. This means that arbitrary computation can be performed up until to some time, after which coherence is lost. This depends on many factors, such as the quality of the qubits (e.g. in terms of the interactions with the environment), the fidelities of the quantum gates, as well as the total number of gates in the full circuit. However, even with a moderate number of qubits, and noisy hardware, many interesting application are being explored. At the same time, it is of utmost importance to reduce the noise sources and improve the quality of qubits (rather than the quantity), to allow longer and longer quantum computation.

1.5 Machine learning

We now leave for a moment the microscopic scale of atoms, molecules and qubits, and move to the human scale, where once again emergence and complexity are intimately related with each other. We are not really interested in emergent phenomena at this scale per se, but rather in the complexity of problems found in the world of industry, information and technology. These can be most diverse, such as the forecasts of financial trends, the prediction of natural disaster, risk prevention or resource optimization. So, where once we had particles, now we have variables, where we had interactions, we have constraints. Regardless, complexity emerges in a similar fashion because of the constraints between a large number of variables. In the following, we narrow down the problem to a specific case, the classification of an image. As we are about to see, this problem faces a curse of dimensionality which resembles very much the many-body problem in physics.

1.5.1 Complexity reloaded

A close friend, who is a photographer, finds herself in trouble when facing the task of sorting all the pictures she took over the years. More than a million digital pictures need to be divided in groups according to the type of landscape (mountains, deserts, cities, etc). So she started labelling them one by one, but soon realized that this would take more than a month's worth of full-time work. "*Isn't there some automatic way of doing so? It does not seem such a difficult task for a computer.*". So she asks for help in writing a computer algorithm to automate the task. Let us look at the problem more closely. We have a very large dataset of images, to be divided between in a small number of categories, or *labels*. Imagine the label "mountains". There are many images that belong to this category, possibly very different but strongly conveying the same concept. All these images shares the same relevant, but unknown, features. For example, we could try to distinguish between images by the presence of lack of particular shapes. To this end, we could hand-craft different spatial filters corresponding to various geometrical shapes and convolving them with a given image. A filter with a pointy-like feature, for instance, could

detect the peak of a mountain. Unfortunately, at the same time it could very well detect a pine treen, or a skyscraper. Even assuming that we can define all the relevant features of a category, the very complex mutual relationship between the labels can easily lead to ambiguities, requiring in turn more and more hard-coded "rules" to increase the accuracy. More importantly, this approach is strictly specialized, with no generalization capabilities to additional new labels. As we increase the number of categories, the problem becomes increasingly complex and intractable for any practical application.

Although this might seem an unlikely situation, it was in fact very common in the facial recognition programs ran by the various investigation agencies in the 70s. Similar to our story, all the biometrics of the face (the features) were measured, classified with the corresponding person's identification (the label), and stored in a big database. Given a new picture, the identity retrieval consisted of measuring the same set of biometrics and compared them with the ones in the database, until the closest match was found. Let us now be more precise. Every image is formed by a set of pixels, each containing real-valued intensities for the red, green and blue colour. For simplicity, let assume black and white pictures. For the total of N pixels, we can write down the "state" of the image as $\boldsymbol{x} = (x_1, \dots, x_N)$. Let us also assume, though unrealistic, binary pixels $x_j = 0, 1$. Then, the state $\boldsymbol{x}$ lives in a space with dimension N, with the total number of possible configurations being 2^N. Each state $\boldsymbol{x}$ conveys a specific "concept", encoded as a label $y(\boldsymbol{x})$ living in a much smaller space $y(\boldsymbol{x}) \in \{y_1, \dots, y_K\}$, with dimension K.

Analogously to statistical mechanics, we can imagine the labels as macrostates and the various images as microstates. For a given macrostate, there exists many different compatible microstates. The link between macro and microstates is the set of features $\mathcal{F}_k = (f_1^{(k)}, f_2^{(k)}, \dots)$, which characterizes the macrostate y_k. For example, for natural images of giraffes ($y_k =$ giraffe), we could have $\mathcal{F}_{\text{giraffe}} = \{\text{long neck}, \text{skinny legs}, \text{etc}\}$. The fundamental problem, in contrast with statistical mechanics, is that we do not know exactly what the relevant features are, and how to encode them into an algorithm. For a classical spin system at thermal equilibrium, the features (i.e. macroscopic properties) are defined by the Hamiltonian $H(\boldsymbol{\sigma}^z)$ and the inverse temperature β. In this case, we know exactly

the probability distribution of the various microstates

$$p(\boldsymbol{\sigma}^z) = Z^{-1}\, e^{-\beta H(\boldsymbol{\sigma}^z)}\,. \tag{1.62}$$

However, there is no such a thing as a simple elementary giraffe "Hamiltonian" governing the behaviours of the images (microstates)

$$p(\boldsymbol{x}) \neq e^{-\mathcal{H}_{\text{giraffe}}}(\boldsymbol{x}) = \ ?? \tag{1.63}$$

but instead, the probability distribution characteristic of the giraffe depends directly on the features $\mathcal{F}_{\text{giraffe}}$ in a way that is not known.

We have seen that the brute force approach of hard-coding the known features into the algorithm leads to severe complexity issues. Yet, each human performs this task continuously and very efficiently. How do we achieve that? Since we are born, we are exposed to a constant flow of images. We received them through our eyes and process them through various regions of our brain, where information is extracted, analyzed and stored. It is in fact the exposure to data that enables us to build the internal representation of the features. The data is provided by the external environment, the world around us. In addition, a learning mechanism is required, which improves our prediction rate over time. With enough data and training, we are capable of becoming very accurate image discriminators. We can then follow a similar approach in building a computer algorithm for image classification. Rather than hand-crafting rules, we can seek an automated way to extract the features directly from the data available. In fact, the data is all we had in the first place, i.e. the large dataset of images. We are then left with the task to designing the learning algorithm to capture the unknown hidden features.

This approach is the core principle of *Machine Learning* (ML), a paradigm whereby computer algorithms are designed to learn from – and make predictions on – data. Importantly, the success of ML algorithms relies on their ability to infer the features and patterns without explicit guidance from a human programmer. Such automatic encoding proceeds by first "training" the algorithm on a large data set and then asking the trained machine

to perform some task. If the ultimate goal is to perform classification, like in our example, the algorithm is trained on a set of labelled data, using *supervised learning*. Supervised here refers to the fact that for each image $\boldsymbol{x}$ we possess the correct label $y(\boldsymbol{x})$ (i.e. the teaching signal). The algorithms is then trained to capture to mutual relationship between images and labels. If however labels are not available, the algorithm can be trained using *unsupervised learning*, where now the full probability distribution of the images $p(\boldsymbol{x})$ is the learning target. We will now show the fundamental properties of supervised learning, and devote Chapter 2 to discuss in detail unsupervised learning.

Supervised learning

In supervised learning, the training dataset $\mathcal{D}$ is made up of images and their correct labels, $\mathcal{D} = \{(\boldsymbol{x}^{(i)}, y^{(i)})\}$ [13]. The task of correctly build the classifier translates into discovering the correct conditional relationship between images $\boldsymbol{x}$ and labels y. If we have access to the conditional probability distribution $p(y_k \mid \boldsymbol{x}; \mathcal{F}(y_k))$, we can predict the label of an image $\boldsymbol{x}$ as:

$$\ell(\boldsymbol{x}) = \underset{k\in[1,K]}{\operatorname{argmax}} \Big\{ p(y_k \mid \boldsymbol{x}; \mathcal{F}(y_k)) \Big\} \,. \tag{1.64}$$

The problem is that this distribution implicitly depends on the set of features $\mathcal{F}(y_k)$, which are unknown. Instead, we parametrize the distribution using a set of parameters $\boldsymbol{\lambda} = (\lambda_1, \lambda_2, \dots)$

$$p(y_k \mid \boldsymbol{x}; \mathcal{F}(y_k)) \longrightarrow p_{\boldsymbol{\lambda}}(y_k \mid \boldsymbol{x}) \,, \tag{1.65}$$

where each λ_j is a real number. The first natural question regards the parametric form of the distribution. For now, let us imagine this as a black box containing some possibly complicated circuit (parametrized by $\boldsymbol{\lambda}$). Given an input image $\boldsymbol{x}$, the circuit outputs the probability $p_{\boldsymbol{\lambda}}(y_k \mid \boldsymbol{x})$ that the image $\boldsymbol{x}$ belong to label y_k. The second question regards the value of the parameters $\boldsymbol{\lambda}$, and how to discover the optimal set $\boldsymbol{\lambda}^*$ that generates the correct conditional distribution. To this end, we require a "measure" of the performance of the classifier, which explicitly depends on $\boldsymbol{\lambda}$, and can be written as an average over the

[13] In supervised learning, some degree of pre-processing is required, such as manually labelling the images.

dataset:

$$\mathcal{C}_{\boldsymbol{\lambda}} = \|\mathcal{D}\|^{-1} \sum_{i=1}^{\|\mathcal{D}\|} C_{\boldsymbol{\lambda}}(\boldsymbol{x}^{(i)}, y^{(i)}) \,. \tag{1.66}$$

The metric $\mathcal{C}_{\boldsymbol{\lambda}}$ is called the *cost function*. For the current example, a typical choice is the cross-entropy between the data and the model distribution [63]:

$$\mathcal{C}_{\boldsymbol{\lambda}} = -\|\mathcal{D}\|^{-1} \sum_{i=1}^{\|\mathcal{D}\|} \sum_{k=1}^{K} \delta(y^{(i)} - y_k) \log p_{\boldsymbol{\lambda}}(y_k \mid \boldsymbol{x}^{(i)}) \,, \tag{1.67}$$

which has a minimum in the parameter space $\boldsymbol{\lambda}$ when

$$\ell(\boldsymbol{x}^{(i)}) = \underset{k\in[1,K]}{\operatorname{argmax}} \left\{ p_{\boldsymbol{\lambda}}(y_k \mid \boldsymbol{x}^{(i)}) \right\} = y^{(i)} \quad \forall (\boldsymbol{x}^{(i)}, y^{(i)}) \in \mathcal{D} \,. \tag{1.68}$$

Starting from some random values of $\boldsymbol{\lambda}$, if we make a small perturbation $\boldsymbol{\lambda} \to \boldsymbol{\lambda} + \delta\boldsymbol{\lambda}$, and assuming the cost function is smooth in $\boldsymbol{\lambda}$, then the change caused by $\delta\boldsymbol{\lambda}$ is $\delta\mathcal{C}_{\boldsymbol{\lambda}} = \nabla_{\boldsymbol{\lambda}}\mathcal{C}_{\boldsymbol{\lambda}}\delta\boldsymbol{\lambda}$. This suggests a straightforward way to implement the learning mechanism, that is making a small change in the parameters $\boldsymbol{\lambda}$ in such a way that the cost function decreases. The simplest version of the learning rule is given by gradient descent

$$\boldsymbol{\lambda} \longrightarrow \boldsymbol{\lambda} - \eta \nabla_{\boldsymbol{\lambda}} \mathcal{C}_{\boldsymbol{\lambda}} \tag{1.69}$$

where η is the step of the update, also called the *learning rate*. By repeating this process many times, we should in principle find the optimal values $\boldsymbol{\lambda}^*$ minimizing the cost function, corresponding to a successful image classifier. In practice however, there are many factor which can prevent this to happen, such as the choice of the cost function, the gradient update technique, and more importantly the parametric form of the probability distribution.

There is much freedom in choosing the specific form of the model, i.e. how the parameters $\boldsymbol{\lambda}$ relates to probability distribution $p_{\boldsymbol{\lambda}}$. Currently, many ML applications are performed with *artificial neural networks*, which essentially fit the data to a graph structure composed of many nodes and edges (Fig. 1.12). In the context of our example of

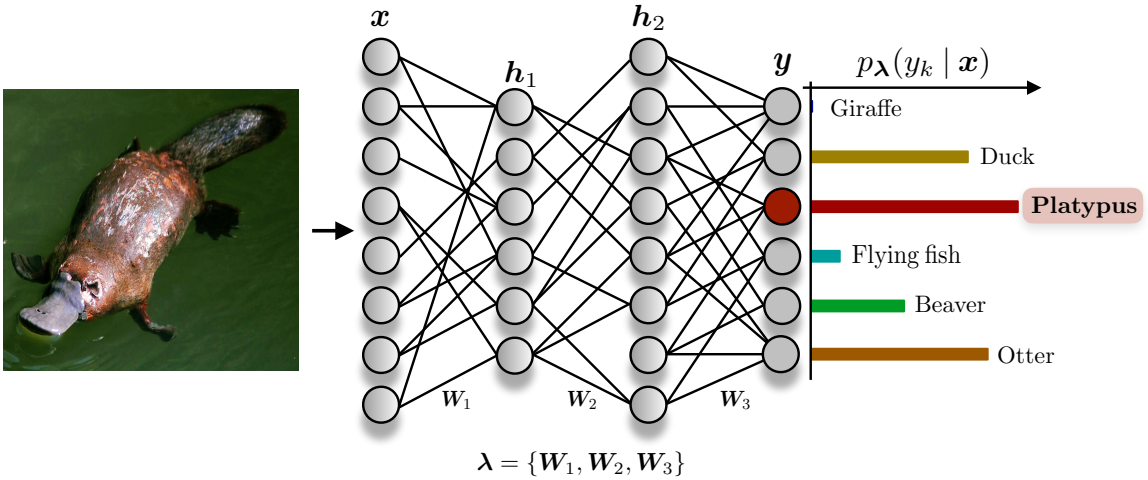

Figure 1.12: **Feedforward neural networks.** An image of a platypus is provided as input to a feedforward neural network. The pixel intensities are processed through the network and the output shows the probabilities for different animals (labels).

image recognition, the image is received by a layer $\boldsymbol{x}$ of input neurons, where the "state" of each neuron x_j is set of the intensity of the corresponding pixel in the image. The image is the processed through a set of layers $\boldsymbol{h}^{(j)}$ of interconnected neurons. Each neuron in one layer $\boldsymbol{h}^{(j)}$ receives a signal from the previous layer $\boldsymbol{h}^{(j-1)}$, it processes it [14], and send it to the neurons in the next layer $\boldsymbol{h}^{(j+1)}$. The signal is forwarded from the input layer to the output layer, naming the model *feedforward neural network*. The output layer contains K neurons, corresponding to the K possible labels in the dataset. We can then select the most likely label $\ell(\boldsymbol{x})$ according to the state of the neurons in the output layer. For artificial neural networks, the parameters $\boldsymbol{\lambda}$ are the strengths of the connections between the neurons in adjacent layers.

[14]The type of computation that each neuron applies to its input signal it somehow arbitrary and can vary between different network realizations.

1.6 Conclusions

The fundamental interactions between many elementary components, whether for particles in a strongly-correlated material, or entangled qubits in a quantum hardware, results into an intractable complexity of the physics at the microscopic scale. For condensed matter systems, such complexity often leads to the emergence of new collective behaviours. Under some circumstances, classical algorithms running on conventional computers are capable of extracting physical properties without the exponential overhead of quantum mechanics. If that is not the case, solutions to the many-body problem should be found using simulations running on quantum hardware.

Similarly to quantum many-body systems, an exploding complexity is observed in many problems in the world of information and technology. At such different scale, ML, and in particular artificial neural networks, provide a powerful framework to discover approximate representations from raw data. Despite being researched for many decades, the performances required for solving highly complex problems in real-world applications has been achieved only relatively recently, using networks made up of several layers, called *deep learning* [64]. The power of the deep neural networks stems from their ability to identify the low dimensional manifold where the relevant features are encoded, effectively compressing the high-dimensional data [65]. This led recently to solutions to long-standing problems, such as image recognition [66], speech recognition [67] or natural language understanding [68].

There is a natural connection between the "curse of dimensionality" encountered in ML and data science, and the quantum many-body problem. In both cases, the problem at hand suffers an exponential scaling in complexity with the number of variables. Furthermore, solutions to the problem can be in general identified with a low-dimensional manifold within the exponentially large configurational space. In the same way that neural networks capture this low-dimensional features using training algorithms and raw data, MPSs exploit their geometry and entanglement properties to capture the low-dimensional "corner" of the Hilbert space, where ground states of gapped local Hamiltonian are found.

After some early works on ML applications to physical problems, such as searching for exotic particles in high-energy physics [69] or solving dynamical mean-field theory in strongly-correlated materials [70], an abrupt turning point took place in early 2016, with seminal works on supervised learning of phases of matters [71, 72] and the introduction of a stochastic neural network called a *restricted Boltzmann machine* (RBM) to study the physics of many-particle systems [73, 74]. The synergy between physics and ML is currently being vigorously investigated in a widespread effort, covering many areas and disciplines. These include the use of ML to improve the performances of MC algorithms [75, 76], the implementation of neural networks as variational wavefunctions for VMC simulations [74, 77, 78, 79, 80] and the application of ML to quantum error correction (QEC) [81, 82, 83, 84, 85]. A more fundamental investigation concerns the connection between deep learning and neural networks with TNs [86, 87, 88, 89], the renormalization group [90, 91, 92], as well as the holographic principle [93]. Finally, we also mention the inverse approach, where techniques and algorithms successful in many-body physics, such as TN, have been exported to the world of ML to solve problems in data science [94, 95, 96, 97].

Much of the attention in ML from the physics community was sparked by the representational power of neural networks as general functions approximator. In this Thesis, we instead focus on the data, and the capability of extracting relevant features from it. We ask whether neural networks are capable of learning physical properties of interest from data generated by quantum systems. This can be either synthetic data generated with computer simulation or, more interestingly, experimental data measured from controlled quantum matter in laboratories. In the following chapters, we will present a general representation of quantum states with neural networks, a set of algorithms to reconstruct unknown quantum state from measurement data, and a framework based on ML for the quantum error correction of topological qubits.

Outline of the Thesis

In Chapter 2, we introduce generative modelling, a ML paradigm based on unsupervised learning that allows a neural network to learn and sample an unknown probability distribution. We begin with an historical overview on the first attempts to build an artificial intelligence, with a particular focus on the early work of John Hopfield, who first established a connection between cognitive science and statistical mechanics. We then introduce the RBM, discussing its properties in detail, as well as the training algorithm. At last, we demonstrate the power of generative modelling by deploying a RBM to learn the thermal distribution of a classical spin system.

In Chapter 3, we introduce a neural-network representation of quantum states based on the connection between generative models and the probabilistic nature of the measurement process in quantum mechanics. We show how to parametrize quantum wavefunctions and density operators describing pure and mixed states respectively. Then, we discuss how to calculate expectation values of physical observables, and present an extension of the replica trick, used to compute entanglement entropy in QCM, to our neural-network framework.

In Chapter 4, we implement the proposed neural-network representation to perform the reconstruction of unknown quantum states from measurement data. We present a set of algorithms that carry out the reconstruction using unsupervised learning, and demonstrate the capability of our technique by performing numerical simulation on both synthetic and real experimental data.

In Chapter 5, we go beyond the characterization of the current generation of quantum hardware, and introduce a framework for quantum error correction based on a RBM. The resulting *neural decoder* can be applied to a wide variety of quantum error correcting codes, with very little specialization. We show numerical calculation for $2d$ toric code.

In Chapter 6, we summarize the results of this Thesis, and conclude by discussing their implications for the current experiments on controlled quantum matter.

Chapter 2

Generative modelling

Modern artificial intelligence relies on the representational power and computational capabilities of networks of interconnected elementary processing units (neurons). This concept came to light during the 1950s, where a collective effort, widespread over many different disciplines, took on the daunting mystery of the nature of human intelligence and the mechanism behind cognition and reasoning. The first proposal of artificial neurons, followed by learning theories, gave rise to the beginning of modern cognitive science, and the birth of neural networks. Much interest in the subject was sparked also by the possibilities offered by the first computing machines. In fact, for the first time, the various proposals for cognition and learning mechanisms could be simulated and tested, gaining invaluable insight in the matter. Furthermore, this new technology raised the question whether such computing devices, equipped with powerful algorithms and enough computational power, could show intelligent behaviour.

2.1 Emergent intelligence

One belief that became shared among many cognitive scientists, was that the human mind acts like a sort of complex processor. Information is received, for instance from sensory inputs, is processed and stored internally. However, the community found itself divided

on the nature of the internal representation of *knowledge*, with two competing paradigms: symbolic manipulation and connectionism. The symbolic approach is based on the idea that knowledge has an internal symbolic representation. It is organized in structures with well-defined mutual relationships, and it is processed through symbolic manipulation. Symbolic structures have the very desirable property of being easy to interpret, while still capable of building sophisticated representation of complex concepts. However, an important shortcoming is that the learning process becomes very challenging. As a result, the symbolic manipulation necessary to process information must follow a pre-determined set of rules, resulting in a poor flexibility in handling exceptions and generalization. One notable example of a symbol system is the CYC project (from EnCYClopedia [98]), founded by Douglas Lenat in 1984. With the goal of coding into human-readable form all the pieces of knowledge that composed human common sense, CYC is the largest symbolic artificial intelligence project ever pursued. It features around 1.5 millions ontological terms and 25 millions common sense rules, provided as statements and instruction by human programmers (for a total of 1000 person-years worth of work! [98]).

The other paradigm, called *connectionism*, relies on a distributed representation of information into a large collection of elementary units, interacting with each other through network connections. Contrary to the symbolic approach, each unit represents a feature of some higher-level object, rather than entire object. The symbolic manipulation of the higher-level objects is then approximated by the lower-level computation performed by a neural network. Intelligent behaviour (or cognition) in connectionist models, arises as a collective property characterized by the interactions of neurons. Here, there are no pre-determined rules. Instead, the network learns a set of internal representations which allows it to process information as if *it knew* the underlying rules. The learning process needs a teaching signal, which must be provided by an external environment (connected to the inputs of the network). Furthermore, the network requires a learning procedure able to encode the relevant input space of the environment into a set of internal representations (i.e. via interactions). Depending on the properties of the network, the nature of the input patterns and the cognitive task at hand, the model requires a learning rule, that is the process driven by the environment according to which the connections are modified.

We summarize the main ingredients for these connectionist models, often called *parallel distributed processing systems* [99]:

- *Environment*: a time-varying Markovian stochastic function over the configuration space of the input patterns. It provides a teaching signal, parametrized as input patterns that occur according to some probability distribution.
- *Processing units*: a set of elementary units that receive information from the environment, process it internally and (possibly) send it out of the cognitive system.
- *Connectivity*: a particular structure of the network connections.
- *Interactions*: A set of strengths associated with the network connections.
- *Elementary computations*: a simple calculation that each unit carries out on the input signal it receives. The specific calculation can vary between different units.
- *Learning rule*: the process that modifies the network connections.

2.1.1 Artificial neural networks

The connectionist school has its root in the mid-1940's, with the work of McCullogh and Pitts [100]. Inspired by recent advances in neurobiology, they proposed a model of computation based on a network of connected neurons. A single neuron is schematically drawn in Fig. 2.1a. The activity of the neuron is assumed to be a "all-or-none" process, where its state can assume one of two values 0 or 1. The neuron is connected to a set of input units, which send either excitatory x_j, or inhibitory $\bar{x}_j$ signals. The elementary computation that the neuron perform on the input is given by

$$\text{output} = \begin{cases} 1, & \text{if } \sum_j x_j \geq \theta \text{ and } \bar{x}_i = 0 \ \forall i \\ 0, & \text{otherwise} \end{cases} \tag{2.1}$$

which gives this neuron the name of *threshold logic unit*. If the neuron receives an inhibitory signal, the results of the computation is 0. Otherwise, it computes the total excitation

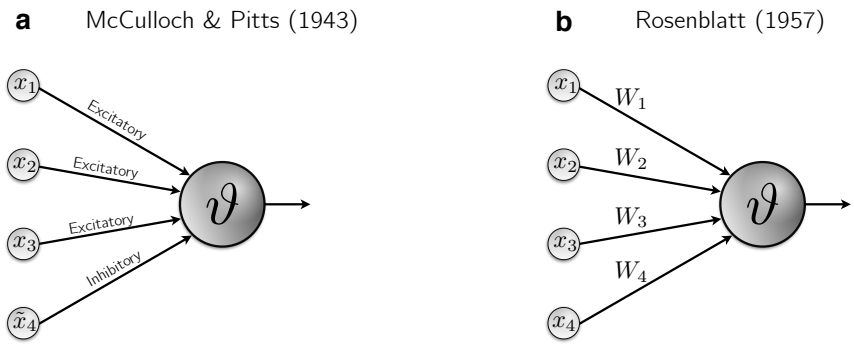

Figure 2.1: **Early models of artificial neurons**. **a)** The first model of artificial neuron, introduced by McCulloch and Pitts in 1943. **b)** The Rosenblatt perceptron neuron.

and calculates the results by comparing it with its threshold θ. By carefully designing the structure of the input signals and choosing a proper threshold θ, the neuron can implement simple logic operations, such as AND, OR and NOT. In turn, more involved operations can be carried out by inter-connecting a collection of single neurons. Despite the potential to realize complex computational circuits, this neuron-like system is limited by the absence of a learning procedure, since in order to implement a given logic circuit, the network design has to be solved explicitly by hand-crafting the connections.

The lack of learning abilities was tackled a few years later, with the seminal work on "cell-assemblies" of the neurophysiologist Donald Hebb. He proposed the strengths of the connections not be fixed. In turn, learning is realized by changing their values according to the "teaching" input signal. More precisely, the connections between two neurons are strengthened when they are simultaneously activated by the external stimuli. This picture seemed oversimplified but, as it turns out, it captures the fundamental essence of the learning mechanism. Shortly afterwards, the very early computing machines were invented:

It had three hundred tubes and a lot of motors. It needed some automatic electric clutches, which we machined ourselves. The memory of the machine was stored in the positions of its control knobs, it used the clutches to adjust its own knobs. We used a surplus gyropilot from a B24 bomber to move the clutches. M. Minsky, New Yorker (1981) [101]

Even though very rudimental, these early machines allowed the first simulations of cognitive models and learning processes (such as the Hebbian learning). The resulting increase in attention in neuron-like processing systems led to the true beginning of connectionism with the invention of the perceptron by Frank Rosenblatt in 1957 [102]. The Rosenblatt neuron resembles much of the McCulloch and Pitts neuron. The input signal is expressed as a binary vector $(x_1, \ldots, x_N)$ and the neuron has a threshold θ. To allow learning, each connection in the perceptron is weighted by a real number W_j. Once again, the neuron computes the signal it receives from input connections and outputs the following result:

$$\text{output} = \begin{cases} 1, & \text{if } \sum_j W_j x_j + b \geq 0 \\ 0, & \text{otherwise} \end{cases} \tag{2.2}$$

where from now on we adopt the bias definition $b = -\theta$. The perceptron introduces expressivity [1] in the model by using a set of interactions $\boldsymbol{W}$ to weight the different inputs. This also allows the perceptron to be trained from examples of the input space. For the case of discriminating two classes of input patterns, provided a solution exists, the perceptron learning theorem provides a systematic way to update the interactions upon convergence. Rather than changing the weights according to the correlations between units (as for the Hebbian learning), the perceptron learning rule minimizes the misclassification error of the training inputs. It is worth mentioning that Rosenblatt also developed a perceptron-based framework for spontaneous learning, such as the clustering of input vectors.

The increasing popularity of neural networks and the initial positive results in applications of the perceptron unreasonably raised the expectations of this neural model, which ended up having an unsettling effect on the cognitive science community. Rosenblatt above all was extremely enthusiastic about the perceptron:

[1] With expressivity we refer to the capability of the processing system to capture a larger class of functions and patterns.

"[*The perceptron is*] *the embryo of an electronic computer that* [*the navy*] *expects will be able to walk, talk, see, write, reproduce itself and be conscious of its existence.*" [103]

Certainly the perceptron offered a new perspective for learning distributed representations, but it was in fact very limited in the computational tasks it could learn. The main criticism to the perceptron was put forward in the famous book *Perceptrons* in 1969 - by Minsky and Papert - where they proved that a single perceptron could only learn linearly separable functions. As such, the perceptron can learn (from training examples) the AND operation, but not the XOR [2]. The effect of the book of Minksy and Papert (whom both believed in the symbolic approach) was disastrous for connectionism. The period of time following the publication of the book until the early 80s will later be called the *first AI winter*, with funding cuts and a general decrease in popularity of artificial neural networks. Minsky will later admit that more investigation should have been done before abandoning the model:

"*I now believe the book was an overkill. So after being irritated with Rosenblatt for overclaiming and diverting all those people along a false path, I started to realize that for what you get out of it - the kind of recognition it can do - it is such a simple machine that it would be astonishing if nature did not make use of it somewhere."* [99]

Notwithstanding the winter, further research in neural networks led to a real resurgence of connectionism in the early 80s. One notable example is the back-propagation algorithm [104], which allowed the learning of multi-layer perceptrons. More interestingly, in 1982 John Hopfield proposed a model for associative memory based on the idea that:

"*[...] the bridge between simple circuits and complex computational properties of higher nervous system may be spontaneous emergence of new computational capabilities from the collective behaviour of large numbers of simple processing elements*". J Hopfield 1982 [105]

[2]Note that this argument does not apply to networks with multiple layers of perceptron, but for this case the learning procedure and a convergence criteria were not known.

2.1.2 The Hopfield associative memory

Consider the problem of retrieving a memory (previously stored) from some partial information. Instead of an error correction "software", the idea is to find a physical system that can spontaneously perform this task, in the sense that the memory retrieval autonomously occurs at the hardware level. To store memories as N-bit objects, Hopfield introduced a network of N binary neurons, described by a state $\boldsymbol{x} = (x_1, \ldots, x_N)$, with a symmetric connectivity (possibly fully-connected). For a memory $\boldsymbol{X}$ to be robust, the physical system, given a distorted/corrupted version $\widetilde{\boldsymbol{X}} = \boldsymbol{X} + \boldsymbol{\delta}$, should spontaneously carry out the computation to retrieve $\boldsymbol{X}$. In other words, given an initial point $\widetilde{\boldsymbol{X}}$ in the phase space, the equations of motion should lead the system to the stable point $\boldsymbol{X}$.

In the Hopfield model, the interactions are encoded into a weight matrix $\boldsymbol{W}$, according to the connectivity. The energy function is

$$E(\boldsymbol{x}) = -\sum_{ij} W_{ij} x_i x_j \,, \tag{2.3}$$

where W_{ij} represents the strength of the interaction between neuron i and neuron j (we consider $b_i = 0$ without loss of generality). The dynamics is given by an update rule identical to the one for the threshold logic unit used in the perceptron:

$$\text{output} = \begin{cases} 1, & \text{if } \sum_j W_{ij} x_j > 0 \\ 0, & \text{otherwise} \end{cases} \tag{2.4}$$

Note however that the connections in the Hopfield model are undirected and that each neuron is updated asynchronously. Following the energy definition, we see that the term in Eq. 2.4 is just the energy difference between the two states $(x_1, \ldots, x_i = 0, x_{i+1}, \ldots)$ and $(x_1, \ldots, x_i = 1, x_{i+1}, \ldots)$:

$$\Delta E_i = E_{x_i=0} - E_{x_i=1} = \sum_j W_{ij} x_j \,. \tag{2.5}$$

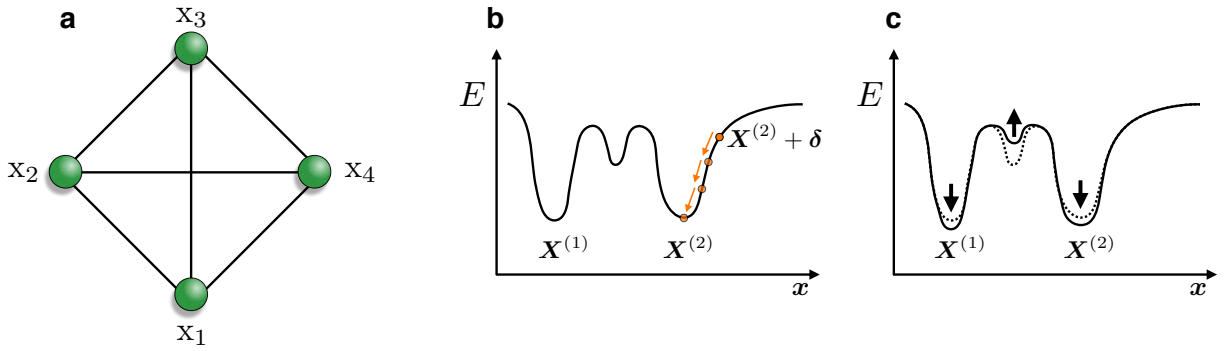

Figure 2.2: **Hopfield model**. **a)** The Hopfield network with $N = 4$ fully connected neurons. **b)** The energy landscape where two memories $\boldsymbol{X}^{(1)}$ and $\boldsymbol{X}^{(2)}$ are encoded. If the system is initialized in a corrupted memory $\widetilde{\boldsymbol{X}} = \boldsymbol{X}^{(2)} + \boldsymbol{\delta}$, the dynamics leads the network to settle to its closest energy minima, which is the correct memory $\boldsymbol{X}^{(2)}$. **c)** During the learning, the energy is decreased for the locations of the real memories. During the unlearning, the energy is increased for spurious minima.

Consequently, the i-th neuron is updated to $x_i = 1$ if $E_{x_i=0} > E_{x_i=1}$ (and vice versa), decreasing the total energy $\Delta E_i \leq 0$. The dynamics of the network corresponds to a trajectory in the phase space following the energy gradient until the closest local energy minimum is found. At these stable points, the system settles down and stops evolving in time. This physical process suggests to encode a set of memories $\{\boldsymbol{X}^{(k)}\}_{k=1}^{n}$ into stable points of the energy landscape of the model. Provided that there is little overlap between the different memories $\boldsymbol{X}^{(k)}$, the energy minima at those points are realized by using the following interaction strengths [105]

$$W_{ij} = \sum_{k=1}^{n} (2X_i^{(k)} - 1)(2X_j^{(k)} - 1) \,, \tag{2.6}$$

which closely resembles the Hebbian learning, where a connection is strengthened if the two neurons are active together. The form of Eq. 2.6 suggests a way to encode/learn a new memory $\boldsymbol{X}^{(n+1)}$, i.e. by increasing the network interactions as

$$\Delta W_{ij} \propto (2X_i^{(n+1)} - 1)(2X_j^{(n+1)} - 1) \,, \tag{2.7}$$

which creates a new energy minimum at the position $\boldsymbol{X}^{(n+1)}$. One problem of this procedure is that in general there will be spurious minima in the landscape, which can trap the system and lead to the retrieval of a false memory. A following work, in analogy with the REM sleep phase, showed that both limitations can be addressed by an *unlearning* phase [106], where the system is allowed to reach equilibrium $\boldsymbol{x}'$ from a random state, and the connections are weakened by

$$\Delta W_{ij} = -\varepsilon(2x'_i - 1)(2x'_j - 1) \, . \tag{2.8}$$

We will soon discover that this learning and unlearning mechanism naturally emerges in the learning process of a RBM.

2.2 Learning internal representations

The work of Hopfield established a connection between the problem of relaxation search and the optimization of a "potential" energy associated with a neural network. Given a set of network parameters (i.e. connection strengths) that maps solutions of a specific problem into local minima of the energy potential, the Hopfield network is able to find a solution through a relaxation process, consisting in asynchronous updates of its units. This process is guaranteed to find a local minima, as long as the initial state is fairly close to the solution (i.e. the deviation $\boldsymbol{\delta}$ is sufficiently small). This assumption, reasonable in the context of content-addressable memories, clearly breaks down for more generic problems, where the solution might be encoded into the lowest energy state of the system, for example. More often, the problem might feature a large set of weak constraints, so that any state with a sufficiently low energy might be a good candidate for the solution of the problem. Considering this more general scenario, Hinton and Sejnowki proposed, shortly after the work of Hopfield, a different type of neural network, called the Boltzmann machine.

2.2.1 The Boltzmann machine

The Boltzmann machine (BM) has a network structure similar to the Hopfield model. A set N binary neurons x_j are coupled together with a weight matrix $\boldsymbol{W}$ (given some connectivity). The total energy is given by

$$E(\boldsymbol{x}) = -\sum_{ij} W_{ij} x_i x_j - \sum_j b_j x_j \, . \tag{2.9}$$

In the context of a relaxation search, the interactions/weights and biases are pre-determined by the particular optimization problem at hand. Its solution might correspond to the global minimum of $E(\boldsymbol{x})$. It is clear that the dynamics generated by the Hopfield update becomes inadequate here. Since we are interested in the global minimum, the system must be able to escape local energy minima, which can be achieved by occasionally allowing jumps to higher energy states. A natural way to create energy jumps is to place the system at thermal equilibrium at inverse temperature $\beta = 1/T$. The thermal fluctuations modify the elementary computation of each neuron from deterministic to stochastic. In the Boltzmann machine, a neuron i updates to the state $x_i = 1$ with probability

$$p_i = \frac{1}{1 + e^{-\beta \Delta E_i}} \tag{2.10}$$

and to the state $x_i = 0$ with probability $1 - p_i$. The energy gap is $\Delta E_i = \sum_j W_{ij} x_j + b_i$. At zero temperature ($\beta \to \infty$) we recover the Hopfield model with a deterministic update rule, minimizing energy. At any finite temperature, the dynamics of the stochastic neurons minimize free energy $F = U - TS$, where $U = \langle E \rangle$ is the internal energy and S is the entropy. This means that higher energy states can be reached with a probability that increases with the temperature. If we let the system update with the above rule, the network will reach the following equilibrium Boltzmann distribution

$$p(\boldsymbol{x}) \propto \; e^{-\beta E(\boldsymbol{x})} \tag{2.11}$$

One effective strategy to exploit the thermal fluctuations to find the global minimum of

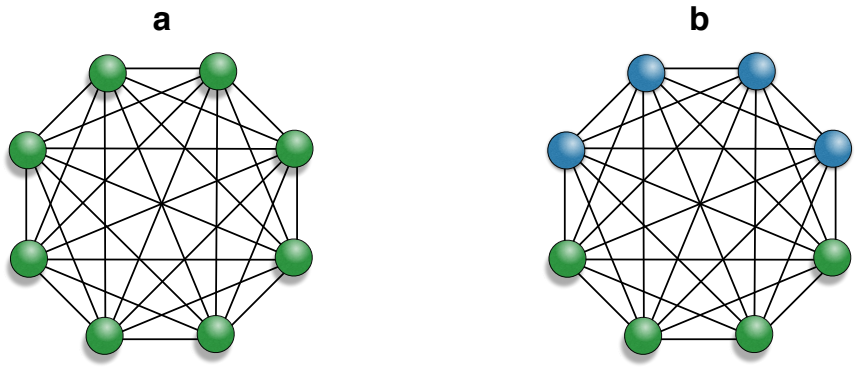

Figure 2.3: **Boltzmann machine**. **a)** Fully-visible Boltzmann machine. **b)** Boltzmann machine with hidden neurons (blue).

$E(\boldsymbol{x})$ was invented around the same time of the Hopfield network. By starting at high temperature and gradually decreasing it with a controlled schedule, this technique, called *simulated annealing* [107], can succeed to discover, at sufficiently low temperature, a final state corresponding to a"good" solution of the optimization problem.

Boltzmann learning

Beyond relaxation searches, another mechanism that benefits from the stochastic nature of the neurons is learning. The optimization, now in parameters space (rather than phase space), takes place according to a distribution of patterns in the input space, and it is solved by modifying the interactions strengths. The flavour of the process is somehow similar to earlier learning mechanisms used in neural networks, but with one important difference: we now want the neural network to build an internal representation of the various constraints underlying the input space. Given enough examples of the input space (governed by some probability distribution), we require the network to learn the constraints and to act as a *generative model*. This means that the BM, upon successful learning, should be able to reproduce the various input patterns, each occurring with the correct probability (enforced by the unknown constraints).

We proceed to qualitatively describe the learning in BMs. Let us consider a N-dimensional input space and a set of input patterns $\boldsymbol{X}^{(k)}$ distributed according to an unknown probability distribution. Let us also assume a BM defined on the same domain, i.e. we associate a stochastic neuron x_j with each bit of the input signal. The goal now is to find a learning procedure which systematically modifies the BM connections $\boldsymbol{W}$ so that the machine is more likely to generate configurations $\boldsymbol{x} = \boldsymbol{X}^{(k)}$ appearing in the input space (i.e. the teaching environment). In other words, we want the BM to learn the unknown probability distribution. The cost function for the corresponding optimization is given by the (log) likelihood of the BM averaged on the data $\langle \log p(\boldsymbol{x}) \rangle_{\text{data}}$. Then, contrary to the case of multi-layer perceptrons, the BM provides a natural way to relate the change in the "output" of the network, with the change in parameters. For a small δW_{ij}, the resulting change in the log-probability is:

$$\delta \log p(\boldsymbol{x}) = \beta \big[x_i x_j - p_{ij}^- \big] \delta W_{ij} \ , \tag{2.12}$$

where p_{ij}^- is the probability that the i-th and j-th neurons are both found in the state 1, when the network runs freely at the equilibrium distribution. This equation explicitly tells us how to modify each weight to increase the likelihood of the model. The update rule for learning is given by

$$\Delta W_{ij} \propto \beta \big[p_{ij}^+ - p_{ij}^- \big] \ . \tag{2.13}$$

where p_{ij}^+ is the probability that both neurons are firing when the state is "clamped" by the environment (this rule will be formally derived in the next section for the restricted version of the BM). We see that the change in the interaction strength W_{ij} is determined by two competing terms. During the *positive phase*, driven by the environment, the weights are increased by the probability p_{ij}^+. This corresponds to traditional Hebbian learning driven by the teaching signal. In the other phase, called the *negative phase*, the interactions are decreased by the probability p_{ij}^-, which is related to the correlation between the two neurons at thermal equilibrium. Note the strong similarity with the learning and unlearning phases in the Hopfield memory. Here, during the positive phase, the energy landscape is modified to decrease the energy of the configuration in phase space corresponding to the input data.

In the negative phase, a reverse learning raises the energy when driven by the equilibrium distribution of the machine.

When the phase space of the BM coincides with the input space, the optimization problem becomes convex, with no local minima in the parameters landscape. While this is certainly a very desirable property, it comes at a heavy price: only second-order correlations between input variables x_i can be learned with a pair-wise interaction W_{ij}. This can be easily seen by considering the XOR problem, where the input space is encoded by a 3-bit string, where two inputs (x_1, x_2) represent the argument of the XOR, and the third bit x_3 represents the result $\mathrm{XOR}(x_1, x_2)$. As for the environment, the teaching distribution shows the four possible configurations: $(0, 0, 1)$, $(1, 0, 0)$, $(0, 1, 0)$ and $(1, 1, 0)$ (each with equal probability). However, since the parity is a higher-order correlation, a BM fails to learn the difference between this set of inputs, and a set where all possible configurations appear with equal probability. The only way to capture higher-order correlations with pair-wise interactions, is to enlarge the physical system with auxiliary neurons which do not appear in the input space. The state of the system can be now written as $\boldsymbol{x} = (\boldsymbol{v}, \boldsymbol{h})$, where the *visible* neurons $\boldsymbol{v}$ are the original neurons (one for each bit of information in the input state), and the new *hidden* neurons are used to capture the correlations among the visible. Each hidden neuron can be thought as a detector for a particular feature/constraint of the inputs (for the XOR example, one single hidden neuron is enough to obtain a solution).

The addition of hidden units increases the expressivity of the network, which is now able to capture more complex constraints in the input space. There is however still one fundamental issue, that in practice prevents learning: the negative phase requires estimating the correlation functions between the neurons at thermal equilibrium. This contribution depends directly on the interaction strengths $\boldsymbol{W}$ and continuously changes during learning. As such, to obtain the required statistics for the parameter update $\Delta\boldsymbol{W}$, the BM must reach thermal equilibrium at each learning iteration. The long equilibration time for large systems and the possibly glassy landscape due to random-like connections results into a computational bottleneck, preventing BM to be learn in a reasonable time.

2.2.2 The restricted Boltzmann machine

Around the same years, a different model called Harmonium (yet very similar to the BM) was proposed by Smolensky. It is a stochastic neural network with two sets of units, visible and hidden (called *representational features* and *knowledge atoms* in the original paper [108]). However, the connectivity in the Harmonium is restricted and only interactions between different types of units are allowed (i.e. no visible-visible or hidden-hidden connections). This neural network is now widely known as *Restricted Boltzmann machine* (RBM). In the RBM, the elementary units are divided into a visible layer $\boldsymbol{v} = (v_1, \dots, v_N)$ and a hidden layer $\boldsymbol{h} = (h_1, \dots, h_{n_h})$, fully connected with a symmetric interaction $\boldsymbol{W}$. The biases are defined as two vectors $\boldsymbol{b}$ and $\boldsymbol{c}$ for the visible and hidden units respectively. (Fig. 2.4). The energy function becomes

$$E_{\boldsymbol{\lambda}}(\boldsymbol{v}, \boldsymbol{h}) = -\sum_{j=1}^{N}\sum_{i=1}^{n_h} W_{ij} h_i v_j - \sum_{j=1}^{N} b_j v_j - \sum_{i=1}^{n_h} c_i h_i \,, \tag{2.14}$$

where we defined $\boldsymbol{\lambda} = (\boldsymbol{W}, \boldsymbol{b}, \boldsymbol{c})$ as the set of network parameters. The probability distribution of the network is given by the Boltzmann distribution

$$p_{\boldsymbol{\lambda}}(\boldsymbol{v}, \boldsymbol{h}) = Z_{\boldsymbol{\lambda}}^{-1} e^{-\beta E_{\boldsymbol{\lambda}}(\boldsymbol{v}, \boldsymbol{h})} \tag{2.15}$$

with partition function

$$Z_{\boldsymbol{\lambda}} = \sum_{\boldsymbol{v}, \boldsymbol{h}} e^{-\beta E_{\boldsymbol{\lambda}}(\boldsymbol{v}, \boldsymbol{h})} \tag{2.16}$$

From now on we will always consider the RBM at the finite temperature $\beta = 1$.

We are interested in using the RBM for generative modelling. Thus, the "output" of the network is given in terms of the probability distribution that the RBM associates with the input space (i.e. the visible layer). This visible distribution is obtained simply by tracing out the hidden degrees of freedom:

$$p_{\boldsymbol{\lambda}}(\boldsymbol{v}) = \mathrm{Tr}_{\boldsymbol{h}}\Big[p_{\boldsymbol{\lambda}}(\boldsymbol{v}, \boldsymbol{h})\Big] = Z_{\boldsymbol{\lambda}}^{-1} \sum_{\boldsymbol{h}} e^{-E_{\boldsymbol{\lambda}}(\boldsymbol{v}, \boldsymbol{h})} \,. \tag{2.17}$$

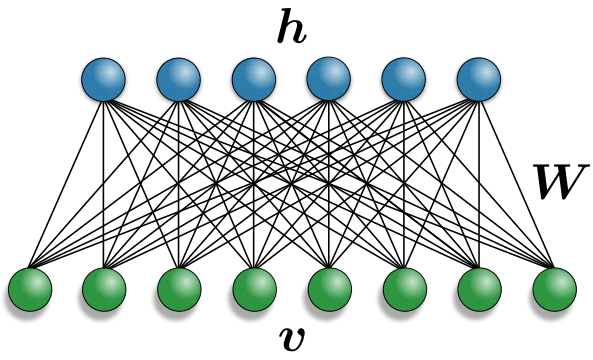

Figure 2.4: **Restricted Boltzmann machine**. The visible nodes (green) are connected to the hidden nodes (blue) with a symmetric matrix of weights $\boldsymbol{W}$. Each visible and hidden unit is also couple to an external field $\boldsymbol{b}$ and $\boldsymbol{c}$ (not drawn in the figure).

Due to the bipartite structure of the network, the probability density factorizes and the summation can be evaluated exactly:

$$\begin{aligned} p_{\boldsymbol{\lambda}}(\boldsymbol{v}) &= Z_{\boldsymbol{\lambda}}^{-1} e^{\sum_j b_j v_j} \sum_{\boldsymbol{h}} e^{\sum_{ij} W_{ij} v_j h_i + \sum_i c_i h_i} \\ &= Z_{\boldsymbol{\lambda}}^{-1} e^{\sum_j b_j v_j} \prod_{i=1}^{n_h} \sum_{h_i=0,1} e^{(\sum_j W_{ij} v_j + c_i) h_i} \\ &= Z_{\boldsymbol{\lambda}}^{-1} e^{\sum_j b_j v_j} \prod_{i=1}^{n_h} \left(1 + e^{\sum_j W_{ij} v_j + c_i}\right) \\ &= Z_{\boldsymbol{\lambda}}^{-1} e^{-\mathcal{E}_{\boldsymbol{\lambda}}(\boldsymbol{v})} \, . \end{aligned} \tag{2.18}$$

Here, we have defined the new energy function

$$\mathcal{E}_{\boldsymbol{\lambda}}(\boldsymbol{v}) = -\sum_{j=1}^{N} b_j v_j - \sum_{i=1}^{n_h} \log\left(1 + e^{\sum_j W_{ij} v_j + c_i}\right) \, , \tag{2.19}$$

often called *free energy* in the machine learning community. However, we will refer to $\mathcal{E}_{\boldsymbol{\lambda}}(\boldsymbol{v})$ as *effective visible energy* (or effective energy), not to be confused with the free energy of the neural network $F = -\log Z_{\boldsymbol{\lambda}}$.

An important property of the RBM, compared to a regular BM, is the conditional independence of the units within the same layer. This means that the state of a hidden unit h_i, only depends on the current state of the visible layer $\boldsymbol{v}$ (and vice versa). The two conditional distributions $p_{\boldsymbol{\lambda}}(\boldsymbol{v} \mid \boldsymbol{h})$ and $p_{\boldsymbol{\lambda}}(\boldsymbol{h} \mid \boldsymbol{v})$, factorize over each single unit:

$$p_{\boldsymbol{\lambda}}(\boldsymbol{v} \mid \boldsymbol{h}) = \prod_{j=1}^{N} p_{\boldsymbol{\lambda}}(v_j \mid \boldsymbol{h}) \qquad , \qquad p_{\boldsymbol{\lambda}}(\boldsymbol{h} \mid \boldsymbol{v}) = \prod_{i=1}^{n_h} p_{\boldsymbol{\lambda}}(h_i \mid \boldsymbol{v}) \, . \tag{2.20}$$

Each independent probability can be obtained using Bayes rule. For instance, the probability for a visible unit j to be active, given a hidden layer in the state $\boldsymbol{h}$ is:

$$p_{\boldsymbol{\lambda}}(v_j = 1 \mid \boldsymbol{h}) = \frac{p_{\boldsymbol{\lambda}}(v_j = 1, \boldsymbol{h})}{p_{\boldsymbol{\lambda}}(\boldsymbol{h})} = \frac{\sum_{\boldsymbol{v}_{/j}} p_{\boldsymbol{\lambda}}(v_1, \ldots, v_j = 1, \ldots, \boldsymbol{h})}{\sum_{\boldsymbol{v}} p_{\boldsymbol{\lambda}}(\boldsymbol{v}, \boldsymbol{h})} \, , \tag{2.21}$$

with $\boldsymbol{v}_{/j} = (v_1, \ldots, v_{j-1}, v_{j+1}, \ldots, v_N)$. Since the partition function cancels out in the ratio of the two distributions, we can easily estimate the conditional distribution

$$\begin{aligned} p_{\boldsymbol{\lambda}}(v_j = 1 \mid \boldsymbol{h}) &= \frac{\cancel{e^{\sum_i e_i h_i}} e^{\, b_j + \sum_i W_{ij} h_i} \prod_{j' \neq j} \sum_{v_{j'}} e^{\, (\sum_i W_{ij'} h_i + b_{j'}) v_{j'}}}{\cancel{e^{\sum_i e_i h_i}} \prod_j \sum_{v_j} e^{\, (\sum_i W_{ij} h_i + b_j) v_j}} \\ &= \frac{e^{\, b_j + \sum_i W_{ij} h_i} \prod_{j' \neq j} \left(1 + e^{\, \sum_i W_{ij'} h_i + b_{j'}}\right)}{\prod_j \left(1 + e^{\, \sum_i W_{ij} h_i + b_j}\right)} \\ &= \frac{e^{\, b_j + \sum_i W_{ij} h_i}}{1 + e^{\, b_j + \sum_i W_{ij} h_i}} \\ &= \frac{1}{1 + e^{-\Delta_{v_j}}} \end{aligned} \tag{2.22}$$

where we defined $\Delta_{v_j} = \sum_i W_{ij} h_i + b_j$. Analogously, the conditional probability of activating hidden units i given the state $\boldsymbol{v}$ is:

$$p_{\boldsymbol{\lambda}}(h_i = 1 \mid \boldsymbol{v}) = \frac{1}{1 + e^{-\Delta_{h_i}}} \, , \tag{2.23}$$

with $\Delta_{h_i} = \sum_j W_{ij} v_j + c_i$.

Unsupervised learning

Generative modelling consists of learning the constraints underlying a distribution defined over the input space, so that the neural network can generate by itself input patterns according to the correct (unknown) probability distribution, which we call $q(\boldsymbol{v})$. The learning occurs by changing the internal parameters and discovering an optimal set $\boldsymbol{\lambda}^*$, such that the RBM distribution closely mimics the target distribution, $p_{\boldsymbol{\lambda}^*} \sim q$. If such set of parameters is found, then the RBM has an internal representation of the target state. We now proceed to derive the learning rule and show the emergence of the positive and negative Hebbian learning phases, previously discussed for the BM.

The learning mechanism is formulated, as usual, with an optimization problem through of a cost function $\mathcal{C}_{\boldsymbol{\lambda}}$. For instance, for a perceptron learning of pattern recognition, the cost function is given by the miss-classification rate (as it is for many of supervised learning tasks). For the case of generative modelling, the ultimate goal is to reduce the "distance" between the input distribution $q(\boldsymbol{v})$ and the RBM distribution $p_{\boldsymbol{\lambda}}(\boldsymbol{v})$. We adopt the standard choice of Kullbach-Leibler (KL) divergence (or relative entropy), defined as:

$$\mathcal{C}^q_{\boldsymbol{\lambda}} \equiv \mathbb{KL}(q \,||\, p_{\boldsymbol{\lambda}}) = \sum_{\boldsymbol{v}} q(\boldsymbol{v}) \log \frac{q(\boldsymbol{v})}{p_{\boldsymbol{\lambda}}(\boldsymbol{v})} = \langle \mathcal{L}_{\boldsymbol{\lambda}} \rangle_q - \mathbb{H}_q \tag{2.24}$$

where

$$\mathbb{H}_q = -\sum_{\boldsymbol{v}} q(\boldsymbol{v}) \log q(\boldsymbol{v}) \tag{2.25}$$

is the entropy of the distribution $q(\boldsymbol{v})$ and we defined the average negative log-likelihood

$$\langle \mathcal{L}_{\boldsymbol{\lambda}} \rangle_q = -\sum_{\boldsymbol{v}} q(\boldsymbol{v}) \log p_{\boldsymbol{\lambda}}(\boldsymbol{v}) \,. \tag{2.26}$$

Note that the KL divergence is not a proper distance measure, since it is non-symmetric and does not satisfy the triangle inequality. Nevertheless, since $\mathbb{KL}(q \,||\, p_{\boldsymbol{\lambda}}) > 0 \;\forall q, p_{\boldsymbol{\lambda}}$ and $\mathbb{KL}(q \,||\, p_{\boldsymbol{\lambda}}) = 0$ if and only if $p_{\boldsymbol{\lambda}} = q$, we can safely use the KL divergence to quantify how close the two distributions are.

We proceed now to calculate the gradient of the cost function with respect to all network parameters $\boldsymbol{\lambda}$. The unknown distribution $q(\boldsymbol{v})$ is in practice encoded into a training dataset $\mathcal{D} = \{\boldsymbol{v}_1, \boldsymbol{v}_2, \dots\}$ containing $\|\mathcal{D}\|$ independent configurations $\boldsymbol{v}_k$, identically distributed according to $q(\boldsymbol{v})$. The unknown distribution $q(\boldsymbol{v})$ is approximated by the data distribution

$$q(\boldsymbol{v}') \simeq \|\mathcal{D}\|^{-1} \sum_{\boldsymbol{v}_k \in \mathcal{D}} \delta(\boldsymbol{v}' - \boldsymbol{v}_k) \,, \tag{2.27}$$

which results into the approximate divergence

$$\mathcal{C}^q_{\boldsymbol{\lambda}} \simeq \mathcal{C}^{\mathcal{D}}_{\boldsymbol{\lambda}} = \langle \mathcal{L}_{\boldsymbol{\lambda}} \rangle_{\mathcal{D}} - \mathbb{H}_{\mathcal{D}} \,. \tag{2.28}$$

Since the entropy of the data $\mathbb{H}_{\mathcal{D}}$ is constant, the only relevant term for the optimization is the negative log-likelihood:

$$\begin{aligned} \langle \mathcal{L}_{\boldsymbol{\lambda}} \rangle_{\mathcal{D}} &= -\|\mathcal{D}\|^{-1} \sum_{\boldsymbol{v}'} \sum_{\boldsymbol{v}_k \in \mathcal{D}} \delta(\boldsymbol{v}' - \boldsymbol{v}_k) \log p_{\boldsymbol{\lambda}}(\boldsymbol{v}') \\ &= -\|\mathcal{D}\|^{-1} \sum_{\boldsymbol{v}_k \in \mathcal{D}} \log p_{\boldsymbol{\lambda}}(\boldsymbol{v}_k) \\ &= \|\mathcal{D}\|^{-1} \sum_{\boldsymbol{v}_k \in \mathcal{D}} \mathcal{E}_{\boldsymbol{\lambda}}(\boldsymbol{v}_k) + \log Z_{\boldsymbol{\lambda}} \,. \end{aligned} \tag{2.29}$$

By taking the gradient of the negative log-likelihood, we obtain

$$\begin{aligned} \nabla_{\boldsymbol{\lambda}} \langle \mathcal{L}_{\boldsymbol{\lambda}} \rangle_{\mathcal{D}} &= \|\mathcal{D}\|^{-1} \sum_{\boldsymbol{v}_k \in \mathcal{D}} \nabla_{\boldsymbol{\lambda}} \mathcal{E}_{\boldsymbol{\lambda}}(\boldsymbol{v}_k) + \nabla_{\boldsymbol{\lambda}} \log Z_{\boldsymbol{\lambda}} \\ &= \|\mathcal{D}\|^{-1} \sum_{\boldsymbol{v}_k \in \mathcal{D}} \nabla_{\boldsymbol{\lambda}} \mathcal{E}_{\boldsymbol{\lambda}}(\boldsymbol{v}_k) + Z_{\boldsymbol{\lambda}}^{-1} \sum_{\boldsymbol{v}} \nabla_{\boldsymbol{\lambda}} p_{\boldsymbol{\lambda}}(\boldsymbol{v}) \\ &= \|\mathcal{D}\|^{-1} \sum_{\boldsymbol{v}_k \in \mathcal{D}} \nabla_{\boldsymbol{\lambda}} \mathcal{E}_{\boldsymbol{\lambda}}(\boldsymbol{v}_k) - \sum_{\boldsymbol{v}} p_{\boldsymbol{\lambda}}(\boldsymbol{v}) \, \nabla_{\boldsymbol{\lambda}} \mathcal{E}_{\boldsymbol{\lambda}}(\boldsymbol{v}) \\ &= \big\langle \nabla_{\boldsymbol{\lambda}} \mathcal{E}_{\boldsymbol{\lambda}}(\boldsymbol{v}) \big\rangle_{\mathcal{D}} - \big\langle \nabla_{\boldsymbol{\lambda}} \mathcal{E}_{\boldsymbol{\lambda}}(\boldsymbol{v}) \big\rangle_{p_{\boldsymbol{\lambda}}} \end{aligned} \tag{2.30}$$

The gradients of the effective visible energy have the simple form:

$$\begin{aligned}
\frac{\partial}{\partial W_{ij}}\mathcal{E}_{\boldsymbol{\lambda}}(\boldsymbol{v}) &= -\frac{e^{\sum_j W_{ij}v_j+c_i}}{1+e^{\sum_j W_{ij}v_j+c_i}}\, v_j = -\frac{1}{1+e^{-\Delta_{h_i}}}\, v_j \\
\frac{\partial}{\partial b_j}\mathcal{E}_{\boldsymbol{\lambda}}(\boldsymbol{v}) &= -v_j \\
\frac{\partial}{\partial c_i}\mathcal{E}_{\boldsymbol{\lambda}}(\boldsymbol{v}) &= -\frac{1}{1+e^{-\Delta_{h_i}}}\,.
\end{aligned} \tag{2.31}$$

where we recall the definition:

$$\Delta_{h_i} = \sum_j W_{ij}v_j + c_i\,. \tag{2.32}$$

Note how the process of learning and unlearning emerges as a natural property the Boltzmann distribution. In the positive phase driven by the data $\langle\nabla_{\boldsymbol{\lambda}}\mathcal{E}_{\boldsymbol{\lambda}}(\boldsymbol{v})\rangle_{\mathcal{D}}$, the energy is lowered for input data configurations (thus increasing their probabilities). During the negative phase, the learning occurs in reverse, with the signal generated by the RBM equilibrium distribution

$$\big\langle\nabla_{\boldsymbol{\lambda}}\mathcal{E}_{\boldsymbol{\lambda}}(\boldsymbol{v})\big\rangle_{p_{\boldsymbol{\lambda}}} = \sum_{\boldsymbol{v}} p_{\boldsymbol{\lambda}}(\boldsymbol{v})\,\nabla_{\boldsymbol{\lambda}}\mathcal{E}_{\boldsymbol{\lambda}}(\boldsymbol{v}) \tag{2.33}$$

Since the evaluation of the negative phase requires a summation over an exponential number of states $\boldsymbol{v}$, we calculate this term using MC sampling of the RBM as

$$\big\langle\nabla_{\boldsymbol{\lambda}}\mathcal{E}_{\boldsymbol{\lambda}}(\boldsymbol{v})\big\rangle_{p_{\boldsymbol{\lambda}}} \simeq \frac{1}{M}\sum_{\ell=1}^{M}\nabla_{\boldsymbol{\lambda}}\mathcal{E}_{\boldsymbol{\lambda}}(\boldsymbol{v}_\ell)\,. \tag{2.34}$$

The effective energy gradient is averaged over M configurations $\boldsymbol{v}_\ell$ sampled from the RBM partition function $\boldsymbol{Z}_{\boldsymbol{\lambda}}$. Given the restricted conditional independence of the RBM graph, a sampling technique called *Gibbs sampling* allows a fast evaluation of the negative phase.

Block Gibbs sampling

The general philosophy behind the MC is to simulate the evolution of the system (in our case the visible layer) state by state, and consider the expectation value of some observables (such as the effective energy gradient) as a time average in *Markov time* (not to be confused with real time evolution). Contrary to full enumeration, where each state $\boldsymbol{v}$ is weighted in the sum, we build a sequence of $M \ll 2^N$ states in such a way that each state $\boldsymbol{v}$ appears with a probability $p_{\boldsymbol{\lambda}}(\boldsymbol{v})$ (also called *importance sampling*). The key to the algorithm is the type of transitions allowed from a state to another. In particular, we consider Markov chains, where a configuration $\boldsymbol{v}^{(k+1)}$ only depends on the previous one $\boldsymbol{v}^{(k)}$ through a non-deterministic process (e.g. memoryless). The protocol driving the Markov-time evolution is characterized by a transition operator $\boldsymbol{T}$ associating a probability $T(\boldsymbol{v} \to \boldsymbol{v}')$, with the transition $(\boldsymbol{v} \to \boldsymbol{v}')$. The transition operator satisfies

$$T(\boldsymbol{v} \to \boldsymbol{v}') \geq 0 \qquad , \qquad \sum_{\boldsymbol{v}'} T(\boldsymbol{v} \to \boldsymbol{v}') = 1 \, . \tag{2.35}$$

At each step in Markov time, a new state $\boldsymbol{v}'$ is selected using $T(\boldsymbol{v} \to \boldsymbol{v}')$. The condition for obtaining a stationary distribution is written as

$$p_{\boldsymbol{\lambda}}(\boldsymbol{v}') = \sum_{\boldsymbol{v}} T(\boldsymbol{v} \to \boldsymbol{v}') \, p_{\boldsymbol{\lambda}}(\boldsymbol{v}) \, , \tag{2.36}$$

and a solution is the so-called *detailed balance* condition:

$$T(\boldsymbol{v}' \to \boldsymbol{v}) \, p_{\boldsymbol{\lambda}}(\boldsymbol{v}') = T(\boldsymbol{v} \to \boldsymbol{v}') \, p_{\boldsymbol{\lambda}}(\boldsymbol{v}) \, . \tag{2.37}$$

In general, we can split the transition probability as

$$T(\boldsymbol{v} \to \boldsymbol{v}') = g(\boldsymbol{v} \to \boldsymbol{v}') A(\boldsymbol{v} \to \boldsymbol{v}') \, , \tag{2.38}$$

where $g(\boldsymbol{v} \to \boldsymbol{v}')$ is the probability of proposing the move $\boldsymbol{v} \to \boldsymbol{v}'$ (*selection probability*) and $A(\boldsymbol{v} \to \boldsymbol{v}')$ is the probability the move accepted (*acceptance probability*).

There exist many different algorithms based on these general principles. One example, commonly used to simulate equilibrium properties of spin systems, is the *Metropolis-Hastings* algorithm [109]. A new configuration is chosen by flipping the state of a spin, randomly chosen with equal probability. If the energy is lowered by the move ($\Delta E < 0$), the update is accepted; otherwise it is accepted with probability $p_{\boldsymbol{\lambda}}(\boldsymbol{v}')/p_{\boldsymbol{\lambda}}(\boldsymbol{v}) = e^{-\Delta E}$. As you can see, the likelihood of the move depends on the ratio of probabilities, where the partition function cancels out. A different strategy consists into updating each variable sequentially, conditioned on the values of all the other variables. This sampling technique is called *Gibbs sampling*, also known as *heat bath* or *Glauber dynamics*. For the case of the RBM, this corresponds to samping each visible unit v_j from the conditional distribution $p_{\boldsymbol{\lambda}}(v_j \mid \boldsymbol{v}_{/j}, \boldsymbol{h})$, and each hidden unit h_i from the conditional distribution $p_{\boldsymbol{\lambda}}(h_i \mid \boldsymbol{v}, \boldsymbol{h}_{/i})$. However, the RBM structure has the special property the each unit is conditionally independent from the others of the same layer. Thus, we can instead sample all the units in one layer simultaneously (*Block Gibbs sampling*). Given an initial state $\boldsymbol{v}$, the selection probability to sample a new state $\boldsymbol{v}'$ is given by the two RBM layer-wise conditional distributions

$$g(\boldsymbol{v} \to \boldsymbol{v}') = p_{\boldsymbol{\lambda}}(\boldsymbol{v}' \mid \boldsymbol{h})\, p_{\boldsymbol{\lambda}}(\boldsymbol{h} \mid \boldsymbol{v})\,, \tag{2.39}$$

corresponding to the transitions $\boldsymbol{v} \to \boldsymbol{h}$ and $\boldsymbol{h} \to \boldsymbol{v}'$. Given this expression, we can easily verify that $g(\boldsymbol{v} \to \boldsymbol{v}')$ satisfy detailed balance condition. Furthermore, once a new state has been chosen according to the selection probability, the move is always accepted with unit probability:

$$\begin{aligned} A(\boldsymbol{v} \to \boldsymbol{v}') &= \text{Min}\left\{1, \frac{p_{\boldsymbol{\lambda}}(\boldsymbol{v}')}{p_{\boldsymbol{\lambda}}(\boldsymbol{v})} \frac{g(\boldsymbol{v}' \to \boldsymbol{v})}{g(\boldsymbol{v} \to \boldsymbol{v}')}\right\} \\ &= \text{Min}\left\{1, \frac{p_{\boldsymbol{\lambda}}(\boldsymbol{v}')}{p_{\boldsymbol{\lambda}}(\boldsymbol{v})} \frac{p_{\boldsymbol{\lambda}}(\boldsymbol{v} \mid \boldsymbol{h})\, p_{\boldsymbol{\lambda}}(\boldsymbol{h} \mid \boldsymbol{v}')}{p_{\boldsymbol{\lambda}}(\boldsymbol{v}' \mid \boldsymbol{h})\, p_{\boldsymbol{\lambda}}(\boldsymbol{h} \mid \boldsymbol{v})}\right\} \\ &= \text{Min}\left\{1, \frac{p_{\boldsymbol{\lambda}}(\boldsymbol{v}')}{p_{\boldsymbol{\lambda}}(\boldsymbol{v})} \frac{p_{\boldsymbol{\lambda}}(\boldsymbol{v}, \boldsymbol{h})}{p_{\boldsymbol{\lambda}}(\boldsymbol{h})} \frac{p_{\boldsymbol{\lambda}}(\boldsymbol{v}', \boldsymbol{h})}{p_{\boldsymbol{\lambda}}(\boldsymbol{v}')} \frac{p_{\boldsymbol{\lambda}}(\boldsymbol{h})}{p_{\boldsymbol{\lambda}}(\boldsymbol{v}, \boldsymbol{h})} \frac{p_{\boldsymbol{\lambda}}(\boldsymbol{v})}{p_{\boldsymbol{\lambda}}(\boldsymbol{v}', \boldsymbol{h})}\right\} \\ &= 1 \end{aligned} \tag{2.40}$$

Note that such property does not necessarily imply that Block Gibbs sampling allows to

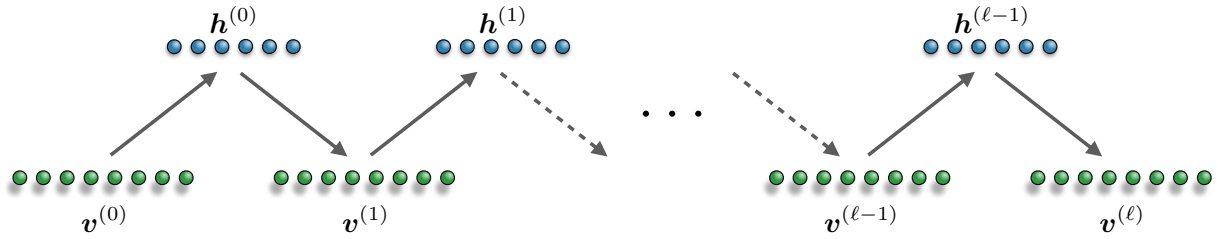

Figure 2.5: **Block Gibbs sampling**. Given some initial state $\boldsymbol{v}^{(0)}$, a new state $\boldsymbol{v}^{(1)}$ is sampled from the distribution $p_{\boldsymbol{\lambda}}(\boldsymbol{v})$ in two steps: first the hidden state is sampled from the conditional distribution $p_{\boldsymbol{\lambda}}(\boldsymbol{h}^{(0)} \mid \boldsymbol{v}^{(0)})$ and then the visible state is sampled from $p_{\boldsymbol{\lambda}}(\boldsymbol{v}^{(1)} \mid \boldsymbol{h}^{(0)})$. Because of the bipartite structure of the RBM graph, each variable in a layer can be samples simultaneously, leading to layer-wise block sampling.

rapidly converge to the equilibrium distribution, although this is indeed true in many cases.

The calculation of the statistics for the negative phase using block Gibbs sampling proceeds as follow. The Markov chain is initialized with a random state $\boldsymbol{v}^{(0)}$ of the visible layer (the sampling can start equivalently from the hidden layer). A new state $\boldsymbol{h}^{(0)}$ of the hidden layer is selected with the conditional distribution $p_{\boldsymbol{\lambda}}(\boldsymbol{h}^{(0)} \mid \boldsymbol{v}^{(0)})$, where the state of each unit i is simultaneously updated according to

$$p_{\boldsymbol{\lambda}}(h_i^{(0)} = 1 \mid \boldsymbol{v}^{(0)}) = \frac{1}{1 + e^{-\sum_j W_{ij} v_j^{(0)} - c_i}} \tag{2.41}$$

After $\boldsymbol{h}^{(0)}$ has been determined, the sampling proceeds by selecting a new state $\boldsymbol{v}^{(1)}$ from the conditional distribution

$$p_{\boldsymbol{\lambda}}(v_j^{(1)} = 1 \mid \boldsymbol{h}^{(0)}) = \frac{1}{1 + e^{-\sum_i W_{ij} h_i^{(0)} - b_j}} \,. \tag{2.42}$$

This process, which constitutes one step of block Gibbs sampling, is repeated for the duration of the Markov chain (Fig. 2.5). Note how the (Markov) dynamics following Gibbs sampling is equivalent to the neuronal update rule of the Boltzmann machine given in Eq. 2.10. However, the advantage of having restricted connection is that, given a fixed number of updates, a RBM can explore a bigger pat of the phase space than a regular BM.

2.3 Training the machine

We have seen that the learning signal for training a RBM has two components. While the positive phase contribution can be easily calculated from the data, the negative phase term needs to be approximated by a MC average

$$\left\langle \nabla_{\boldsymbol{\lambda}} \mathcal{E}_{\boldsymbol{\lambda}}(\boldsymbol{v}) \right\rangle_{p_{\boldsymbol{\lambda}}} \approx \frac{1}{M} \sum_{k=1}^{M} \nabla_{\boldsymbol{\lambda}} \mathcal{E}_{\boldsymbol{\lambda}}(\boldsymbol{v}_k) \,. \tag{2.43}$$

Provided the RBM has reached thermal equilibrium, the configuration $\boldsymbol{v}_k$ generated with block Gibbs sampling are distributed according to the RBM distribution $p_{\boldsymbol{\lambda}}(\boldsymbol{v})$. This means that, for any given learning iteration, the Markov chain should run for an appropriate amount of time, before the statistics can be collected. Since the equilibrium distribution constantly change with $\boldsymbol{\lambda}$ during learning, the chain need to equilibrate at each learning step, creating then a computational bottleneck in the training. This problem was solved by Hinton in 2002 [110], by replacing the KL divergence with an approximate distance measure called *Contrastive Divergence* (CD).

2.3.1 Contrastive divergence

Recall the definition of the cost function for learning a RBM,

$$\mathbb{KL}(q \,||\, p_{\boldsymbol{\lambda}}) = \sum_{\boldsymbol{v}} q(\boldsymbol{v}) \log q(\boldsymbol{v}) - \sum_{\boldsymbol{v}} q(\boldsymbol{v}) \log p_{\boldsymbol{\lambda}}(\boldsymbol{v}) \,, \tag{2.44}$$

which measures the (statistical) divergence between the data distribution $q(\boldsymbol{v})$ and the equilibrium RBM distribution $p_{\boldsymbol{\lambda}}(\boldsymbol{v})$. The possibly long equilibration time is not the only issue of training the network using the KL divergence. In fact, the variance of the gradient might grow large, being the difference between average gradients calculated on the data and the RBM distributions. To introduce the new cost function, we consider a collection of Gibbs sampling Markov chains. Instead of random initial configurations, each chain is initialized with a sample from the training dataset. So, at Markov time $k = 0$, the

distribution of the chains is given by the data distribution $q(\boldsymbol{v})$. After performing k steps of block Gibbs sampling, the chains are distributed with probability $p^{(k)}_{\boldsymbol{\lambda}}(\boldsymbol{v})$, which becomes the equilibrium distribution in the large k limit:

$$\lim_{k\to\infty} p^{(k)}_{\boldsymbol{\lambda}}(\boldsymbol{v}) = p_{\boldsymbol{\lambda}}(\boldsymbol{v})\,. \tag{2.45}$$

Instead of minimizing the KL divergence, Hinton considered the k-steps contrastive divergence (CD_k), defined as

$$\mathbb{CD}_k = \mathbb{KL}(p^{(0)} \,||\, p_{\boldsymbol{\lambda}}) - \mathbb{KL}(p^{(k)}_{\boldsymbol{\lambda}} \,||\, p_{\boldsymbol{\lambda}})\,. \tag{2.46}$$

Using this cost function, the contribution to the gradient from the equilibrium distribution can be approximately eliminated:

$$\nabla_{\boldsymbol{\lambda}}\mathbb{CD}_k \approx \left\langle \nabla_{\boldsymbol{\lambda}}\mathcal{E}_{\boldsymbol{\lambda}}(\boldsymbol{v})\right\rangle_q - \left\langle \nabla_{\boldsymbol{\lambda}}\mathcal{E}_{\boldsymbol{\lambda}}(\boldsymbol{v})\right\rangle_{p^{(k)}_{\boldsymbol{\lambda}}}\,. \tag{2.47}$$

Clearly, this approximation reduces substantially the computational burden, since we need to run the Markov chain for only k steps, before evaluating the negative phase contribution. Furthermore, the value of k needs not to be large in general. In fact, empirical evidence showed that even a chain as short as $k = 1$ could provide a good enough learning signal to achieve convergence in the training. Starting from the data samples, one Gibbs sampling step generates a new visible state $\boldsymbol{v}$. The distribution underlying the new states of the Markov chains, is now statistically closer to the distribution $p_{\boldsymbol{\lambda}}$. If follows that [110]:

$$\mathbb{KL}(p^{(0)} \,||\, p_{\boldsymbol{\lambda}}) > \mathbb{KL}(p^{(1)} \,||\, p_{\boldsymbol{\lambda}})\,, \qquad \text{unless} \;\; p^{(0)} = p^{(1)}_{\boldsymbol{\lambda}} \tag{2.48}$$

$$\text{If } \; p^{(0)} = p^{(1)}_{\boldsymbol{\lambda}}\,, \quad \text{then } \; p^{(0)} = p_{\boldsymbol{\lambda}}\,, \tag{2.49}$$

where in the last statement we are assuming that all the elements of the transition matrix $\boldsymbol{T}$ of the Markov chains are non-zero.

Let us now take a closer look at the gradient of $\mathbb{CD}_k$, and derive the learning rule. We already know the gradient of the first term, so let us consider the KL divergence between

the distribution at equilibrium and Markov time k:

$$\mathbb{KL}(p_{\boldsymbol{\lambda}}^{(k)} \,||\, p_{\boldsymbol{\lambda}}) = \sum_{\boldsymbol{v}} p_{\boldsymbol{\lambda}}^{(k)}(\boldsymbol{v}) \log p_{\boldsymbol{\lambda}}^{(k)}(\boldsymbol{v}) - \sum_{\boldsymbol{v}} p_{\boldsymbol{\lambda}}^{(k)}(\boldsymbol{v}) \log p_{\boldsymbol{\lambda}}(\boldsymbol{v}) \tag{2.50}$$

where now the entropy contribution $\mathbb{H}(p_{\boldsymbol{\lambda}}^{(k)}) = \sum_{\boldsymbol{v}} p_{\boldsymbol{\lambda}}^{(k)} \log p_{\boldsymbol{\lambda}}^{(k)}$ also depends on the parameters $\boldsymbol{\lambda}$. By taking the gradient $\nabla_{\boldsymbol{\lambda}}$ of this KL divergence we obtain

$$\begin{aligned}
\nabla_{\boldsymbol{\lambda}}\mathbb{KL}(p_{\boldsymbol{\lambda}}^{(k)} \,||\, p_{\boldsymbol{\lambda}}) &= \sum_{\boldsymbol{v}} \nabla_{\boldsymbol{\lambda}} p_{\boldsymbol{\lambda}}^{(k)}(\boldsymbol{v}) \log \frac{p_{\boldsymbol{\lambda}}^{(k)}(\boldsymbol{v})}{p_{\boldsymbol{\lambda}}(\boldsymbol{v})} - \sum_{\boldsymbol{v}} p_{\boldsymbol{\lambda}}(\boldsymbol{v}) \nabla_{\boldsymbol{\lambda}} \mathcal{E}_{\boldsymbol{\lambda}}(\boldsymbol{v}) + \cancel{\sum_{\boldsymbol{v}} p_{\boldsymbol{\lambda}}^{(k)} \nabla_{\boldsymbol{\lambda}} \mathcal{E}_{\boldsymbol{\lambda}}^{(k)}(\boldsymbol{v})} \\
&\quad - \cancel{\sum_{\boldsymbol{v}} p_{\boldsymbol{\lambda}}^{(k)} \nabla_{\boldsymbol{\lambda}} \mathcal{E}_{\boldsymbol{\lambda}}^{(k)}(\boldsymbol{v})} + \sum_{\boldsymbol{v}} p_{\boldsymbol{\lambda}}^{(k)}(\boldsymbol{v}) \nabla_{\boldsymbol{\lambda}} \mathcal{E}_{\boldsymbol{\lambda}}(\boldsymbol{v}) \\
&= \sum_{\boldsymbol{v}} \nabla_{\boldsymbol{\lambda}} p_{\boldsymbol{\lambda}}^{(k)}(\boldsymbol{v}) \log \frac{p_{\boldsymbol{\lambda}}^{(k)}(\boldsymbol{v})}{p_{\boldsymbol{\lambda}}(\boldsymbol{v})} - \left\langle \nabla_{\boldsymbol{\lambda}} \mathcal{E}_{\boldsymbol{\lambda}}(\boldsymbol{v}) \right\rangle_{p_{\boldsymbol{\lambda}}} + \left\langle \nabla_{\boldsymbol{\lambda}} \mathcal{E}_{\boldsymbol{\lambda}}(\boldsymbol{v}) \right\rangle_{p_{\boldsymbol{\lambda}}^{(k)}} .
\end{aligned} \tag{2.51}$$

Therefore, the gradient of $\mathbb{CD}_k$ is:

$$\begin{aligned}
\nabla_{\boldsymbol{\lambda}}\mathbb{CD}_k &= \nabla_{\boldsymbol{\lambda}}\mathbb{KL}(q \,||\, p_{\boldsymbol{\lambda}}^{(\infty)}) - \nabla_{\boldsymbol{\lambda}}\mathbb{KL}(p_{\boldsymbol{\lambda}}^{(k)} \,||\, p_{\boldsymbol{\lambda}}^{(\infty)}) \\
&= \left\langle \nabla_{\boldsymbol{\lambda}} \mathcal{E}_{\boldsymbol{\lambda}}(\boldsymbol{v}) \right\rangle_q - \cancel{\left\langle \nabla_{\boldsymbol{\lambda}} \mathcal{E}_{\boldsymbol{\lambda}}(\boldsymbol{v}) \right\rangle_{p_{\boldsymbol{\lambda}}}} \\
&\quad - \sum_{\boldsymbol{v}} \nabla_{\boldsymbol{\lambda}} p_{\boldsymbol{\lambda}}^{(k)}(\boldsymbol{v}) \log \frac{p_{\boldsymbol{\lambda}}^{(k)}(\boldsymbol{v})}{p_{\boldsymbol{\lambda}}(\boldsymbol{v})} + \cancel{\left\langle \nabla_{\boldsymbol{\lambda}} \mathcal{E}_{\boldsymbol{\lambda}}(\boldsymbol{v}) \right\rangle_{p_{\boldsymbol{\lambda}}}} - \left\langle \nabla_{\boldsymbol{\lambda}} \mathcal{E}_{\boldsymbol{\lambda}}(\boldsymbol{v}) \right\rangle_{p_{\boldsymbol{\lambda}}^{(k)}} \\
&= \left\langle \nabla_{\boldsymbol{\lambda}} \mathcal{E}_{\boldsymbol{\lambda}}(\boldsymbol{v}) \right\rangle_q - \left\langle \nabla_{\boldsymbol{\lambda}} \mathcal{E}_{\boldsymbol{\lambda}}(\boldsymbol{v}) \right\rangle_{p_{\boldsymbol{\lambda}}^{(k)}} - \sum_{\boldsymbol{v}} \nabla_{\boldsymbol{\lambda}} p_{\boldsymbol{\lambda}}^{(k)}(\boldsymbol{v}) \log \frac{p_{\boldsymbol{\lambda}}^{(k)}(\boldsymbol{v})}{p_{\boldsymbol{\lambda}}(\boldsymbol{v})} .
\end{aligned} \tag{2.52}$$

We can see that by considering the difference between the two KL divergences, the averages of the gradients of the effective energy with respect the equilibrium distribution cancel out. We are left with two averages: one is over the data distribution (and trivial to compute) and the other is over the RBM distribution after k steps of block Gibbs sampling (tractable

for moderate values of k). The approximation consists into neglecting the last term

$$\delta_{CD} = \sum_{\boldsymbol{v}} \nabla_{\boldsymbol{\lambda}} p_{\boldsymbol{\lambda}}^{(k)}(\boldsymbol{v}) \log \frac{p_{\boldsymbol{\lambda}}^{(k)}(\boldsymbol{v})}{p_{\boldsymbol{\lambda}}(\boldsymbol{v})} \tag{2.53}$$

whose contribution was found to be small in numerical experiments. A rigorous calculation of this approximation error was given later, in the form of two upper bounds, from Bengio and Delalleau [111] and Fischer and Igel [112]. The latter, considering an expansion of the log-probability into irreducible Markov chains, showed that the CD bias is bounded by:

$$\delta_{CD} \leq \frac{1}{2} \sum_{\boldsymbol{v}} |q(\boldsymbol{v}) - p_{\boldsymbol{\lambda}}(\boldsymbol{v})| (1 + e^{-(N+n_h)\Delta})^k \tag{2.54}$$

where Δ is the maximum energy gap in the RBM obtained by changing the value of one unit, i.e. $\Delta = \max\{\Delta_{\boldsymbol{v}}, \Delta_{\boldsymbol{h}}\}$, with parameters

$$\Delta_{\boldsymbol{v}} = \sup_{\ell=1,\dots,N} \Big\{ |E(\boldsymbol{v}, \boldsymbol{h}) - E(\boldsymbol{v}', \boldsymbol{h})|, \ \forall (\boldsymbol{v}, \boldsymbol{v}', \boldsymbol{h}) \text{ s.t. } v_j = v'_j \ \forall j \neq \ell \Big\} \tag{2.55}$$

$$\Delta_{\boldsymbol{h}} = \sup_{\ell=1,\dots,n_h} \Big\{ |E(\boldsymbol{v}, \boldsymbol{h}) - E(\boldsymbol{v}, \boldsymbol{h}')|, \ \forall (\boldsymbol{v}, \boldsymbol{h}, \boldsymbol{h}') \text{ s.t. } h_i = h'_i \ \forall i \neq \ell \Big\} \tag{2.56}$$

The first term $|q(\boldsymbol{v}) - p_{\boldsymbol{\lambda}}(\boldsymbol{v})|$ decreases during the learning as $p_{\boldsymbol{\lambda}}(\boldsymbol{v})$ becomes closer to the data distribution. The other term approach zero when k grows large, where we retrieve the original KL divergence

$$\lim_{k\to\infty} \mathbb{CD}_k = \mathbb{KL}(p^{(0)} \,||\, p_{\boldsymbol{\lambda}}) - \lim_{k\to\infty} \mathbb{KL}(p_{\boldsymbol{\lambda}}^{(k)} \,||\, p_{\boldsymbol{\lambda}}) \approx \mathbb{KL}(p^{(0)} \,||\, p_{\boldsymbol{\lambda}}) \,. \tag{2.57}$$

For a moderate to small value of k, the approximation error depends on the size of the RBM and the energy gap. As such, it is a good practice to keep the weights small during the training, for examples by using a weight penalty. However, numerical experiments showed that, even for a small k and large weights (leading to a large bias δ_{CD}), the learning signal provided by the CD is in general still a good approximation of the KL divergence, in the sense that the sign of the learning is mostly correct [113].

2.3.2 Stochastic optimization

For the RBM, the cost function $\mathcal{C}_{\boldsymbol{\lambda}}$ to minimize is given by a constant entropy term $\mathbb{H}(\hat{q})$ (which we omit), plus the average negative log-likelihood over the entire dataset $\mathcal{D}$:

$$\langle \mathcal{L}_{\boldsymbol{\lambda}} \rangle_{\mathcal{D}} = -\|\mathcal{D}\|^{-1} \sum_{\boldsymbol{\sigma}_k \in \mathcal{D}} \log p_{\boldsymbol{\lambda}}(\boldsymbol{v}_k) \tag{2.58}$$

The most simple and straightforward way to approach this problem is gradient descent, which consists into sequentially updating each parameter λ_j in the opposite direction of the gradient of the cost function with respect to such parameter. Starting from some initial guess $\boldsymbol{\lambda}^{(0)}$, the parameter at *optimization time* $(t+1)$ is given by

$$\lambda_j^{(t+1)} = \lambda_j^{(t)} - \eta \, G_{\boldsymbol{\lambda},j}^{(t)} \tag{2.59}$$

where the learning rate η controls the size of the update step. We also introduced the "full" gradient $G_{\boldsymbol{\lambda},j}$, evaluated over the entire dataset:

$$G_{\boldsymbol{\lambda},j} = \frac{\partial}{\partial \lambda_j} \langle \mathcal{L}_{\boldsymbol{\lambda}} \rangle_{\mathcal{D}} = -\|\mathcal{D}\|^{-1} \sum_{\boldsymbol{\sigma}_k \in \mathcal{D}} \frac{\partial}{\partial \lambda_j} \log p_{\boldsymbol{\lambda}}(\boldsymbol{v}_k) \tag{2.60}$$

This optimization procedure is guaranteed to converge to the global minimum, if the objective function $\mathcal{C}_{\boldsymbol{\lambda}}$ is convex. As this is false for RBMs, the trajectory of the system in parameter space following the gradient $G_{\boldsymbol{\lambda}}$ only guarantees to converge to a local minimum of $\mathcal{C}_{\boldsymbol{\lambda}}$. Once the system settles in one of these minima, it won't be able to escape it. Further, the evaluation of the full gradient $G_{\boldsymbol{\lambda}}$ is also computationally expensive, since it scales with the data-set size as $O(\|\mathcal{D}\| \dim(\boldsymbol{\lambda}))$.

Stochastic gradient descent

The vanilla gradient descent is plagued by a similar problem encountered before in the retrieval of memories in the Hopfield network. The system is unable to escape the local minima. Note however that here we have learning trajectories in parameter space ($\boldsymbol{\lambda}$),

while in the Hopfield network the dynamics to retrieve a memory was in the network phase space ($\boldsymbol{x}$). The phase space trajectories of the RBM are governed by the stochastic dynamics at temperature $T = 1$, which allows one to extract the statistic for the learning. At a different time-scale, the RBM flows in the parameter space, according to the gradient of the KL divergence. To aid this latter process and avoid getting trapped in local minima, we restrict the calculation of the gradient over a small subset of M of training samples (with $M \ll \|\mathcal{D}\|$). The update rule becomes

$$\lambda_j^{(t+1)} = \lambda_j^{(t)} - \eta \, g_{\boldsymbol{\lambda},j} \tag{2.61}$$

where the *batch gradient* is

$$g_{\boldsymbol{\lambda},j} = \frac{\partial}{\partial \lambda_j} \langle \mathcal{L}_{\boldsymbol{\lambda}} \rangle_{\mathcal{D}^{[M]}} = -\frac{1}{M} \sum_{k=1}^{M} \frac{\partial}{\partial \lambda_j} \log p_{\boldsymbol{\lambda}}(\boldsymbol{v}_k) \tag{2.62}$$

Note how this optimization strategy, called *batch stochastic gradient descent* (SGD), is not performing a true descent in the parameters landscape. In fact, since $g_{\boldsymbol{\lambda},j} \neq G_{\boldsymbol{\lambda},j}$, the cost function $\mathcal{C}_{\boldsymbol{\lambda}}$ can eventually increase within small optimization transients, due to the noise present in the stochastic gradient $g_{\boldsymbol{\lambda},j}$. We then tackled both problems of vanilla gradient: noise can help escape local minima and the reduced size of the batch leads to a faster convergence (more gradient updates per unit time).

One problem of SGD is given by ravines, i.e. regions where the gradient is more steep in one dimension with respect to others, thus resulting into oscillations in that dimension and slow progression in others. This is solve with the *momentum* technique [114], by adding a fraction γ of the previous gradient updates to the current move:

$$\lambda_j^{(t+1)} = \lambda_j^{(t)} - \eta \, g_{\boldsymbol{\lambda},j} - \gamma(\lambda_j^{(t)} - \lambda_j^{(t-1)}) \tag{2.63}$$

Further, there are several ways to make the SGD adaptive to each weights, so that the learning rate now becomes a vector, where η_j represent the step size of the update along the dimension λ_j. Some examples are Adam [115], AdaGrad [116] and AdaDelta [117].

The effect of SGD (compared to the vanilla gradient descent) can be understood in analogy with the noise present in Langevin dynamics of system of classical particles. The gradient $g_{\boldsymbol{\lambda},j}$ used for SGD updates is an estimator of the true gradient $G_{\boldsymbol{\lambda},j}$ (computed over a very large number of samples). According to the central limit theorem, $g_{\boldsymbol{\lambda}}$ is Gaussian distributed around $\bar{g}_{\boldsymbol{\lambda}} = G_{\boldsymbol{\lambda}}$ with variance $\mathrm{Var}(G_{\boldsymbol{\lambda}})/\sqrt{M}$. The update rule for SGD can be re-written as

$$\begin{aligned} \lambda_j^{(t+1)} &= \lambda_j^{(t)} - \eta\, G_{\boldsymbol{\lambda}} - \eta\, \mathcal{N}(0, \frac{\mathrm{Var}(G_{\boldsymbol{\lambda}})}{\sqrt{M}}) \\ &= \lambda_j^{(t)} - \eta\, G_{\boldsymbol{\lambda}} - \eta\, \frac{\mathrm{Var}(G_{\boldsymbol{\lambda}})}{\sqrt{M}} \mathcal{N}(0,1) \end{aligned} \tag{2.64}$$

where $\mathcal{N}(\mu, \sigma)$ is a Gaussian centered at μ and with variance σ. This expression is identical to the equation governing the discrete Langevin dynamics of a system of classical particle interacting with a potential V at temperature T. In this case, the positions $R_j^{(t+1)}$ of particle j after a discrete evolution of time Δ_t from position $R_j^{(t)}$ is:

$$R_j^{(t+1)} = R_j^{(t)} - \Delta_t\, \nabla_{\boldsymbol{R}} V(\boldsymbol{R}) + \sqrt{2T\Delta_t}\, \mathcal{N}(0,1) \tag{2.65}$$

By comparing the two equation, we see that the learning dynamics in SGD is equivalent of the Langevin dynamics of the network parameters. We can define an effective temperature for the SGD as

$$T_{eff} \propto \frac{\sqrt{\eta}}{M} \tag{2.66}$$

which is perfectly reasonable. As we decrease the batch size M, the gradient used in the move becomes a biased estimator, corresponding to a noisier trajectory, and thus to a higher effective temperature. The opposite argument goes for the learning rate η.

Natural gradient descent

In the numerical calculation performed, SGD (possibly with momentum) was able to converge to an acceptable solution of the optimization for most of the cases of study. However, for a few instances, first order optimization techniques failed to find a solution. For

these cases, we resorted to second-order methods, specifically the *natural gradient descent* (NGD) [118, 119]. We found this optimization method to be in general more effective, though at the cost of increased computational resources. In this case we update the parameters as

$$\lambda_j^{(t+1)} = \lambda_j^{(t)} - \eta \sum_i S_{ij}^{-1} \, g_{\boldsymbol{\lambda},j} \, , \tag{2.67}$$

where we have introduced the Fisher information matrix

$$S_{ij} = \|\mathcal{D}\|^{-1} \sum_{\boldsymbol{\sigma}_k \in \mathcal{D}} \frac{\partial \log p_{\boldsymbol{\lambda}}(\boldsymbol{v}_k)}{\partial \lambda_i} \frac{\partial \log p_{\boldsymbol{\lambda}}(\boldsymbol{v}_k)}{\partial \lambda_j} \, . \tag{2.68}$$

The update can be either computed on the entire dataset or a small batch of samples. The learning rate magnitude η is usually set to

$$\eta = \frac{\eta_0}{\sqrt{\sum_{ij} S_{ij} \; g_{\boldsymbol{\lambda},i} \; g_{\boldsymbol{\lambda},j}}} \tag{2.69}$$

and changed during learning, starting from an initial learning rate η_0 [120]. The matrix S_{ij} takes into account the fact that, since the parametric dependence of the RBM function is non-linear, a small change of some parameters may correspond to a very large change of the distribution. In this way one implicitly uses an adaptive learning rate for each parameter λ_j and speed-up the optimization compared to the simplest SGD. We notice that a very similar technique is successfully used in QMC for optimizing high-dimensional variational wavefunctions [121]. Similarly to our case, noisy gradients, which come from the MC statistical evaluation of energy derivatives with respect to the parameters, are present, while the matrix S is instead given by the covariance matrix of these forces. Finally, since the matrix S_{ij} is affected by statistical noise, we regularize it by adding a small diagonal offset, thus improving the stability of the optimization.

Hyper-parameters

In addition to the optimization problem in the parameter space $\boldsymbol{\lambda}$, the RBM training requires another level of optimization, for the so-called *hyper-parameters*. These includes all the "knobs" that control the RBM representation (such as the number of hidden units) and the training algorithm (such as the learning rate). These parameters do not appear explicitly in the cost function, and has to be carefully chosen in a different way. Although there exist strategies to automate this hyper-optimization (such as Bayesian optimization), we obtained the optimal hyper-parameters using a simple grid search (optimal here means that the values discovered allowed convergence in the training). The complete set of hyper-parameters are:

- *Number of hidden units* n_h: the amount of resources required to represent a given distribution. The scaling of performances with n_h naturally provides a convergence parameters for the simulations.
- *Order of Contrastive Divergence*: number of block Gibbs sampling of the visible layer performed before obtaining the statistics.
- *Learning rate* η: the step size for the gradient descent update. The learning rate can made adaptive to the parameters and decreases during the training.
- *Batch size* M: the number of training configurations used to evaluate the gradient of the cost function at each training step.
- *Number of Markov Chains* M_C: total number of parallel Markov chains used to compute the negative phase of contrastive divergence.
- *Regularizations*: the amount of weight decay regularization used for the gradients [122], as well as the regularization γ_{NG} used in NGD.
- *Initial conditions*: the distribution used to initialize the parameters. Unless otherwise stated, all initial parameters are drawn from a Gaussian distribution centered around zero and with variance equal to the hyper-parameter w_0.

Evaluation

The evaluation of the training for generative models is in general more involved than discriminative models. If the goal of the training is the classification of input patterns, the model can be simply evaluated by calculating the miss-classification rate on a validation dataset, containing input samples not included in the training dataset. For generative models, a RBM model with parameters $\boldsymbol{\lambda}$ faithfully capture the target distribution when the cost function

$$\mathcal{C}_{\boldsymbol{\lambda}} = \langle \mathcal{L}_{\boldsymbol{\lambda}} \rangle_{\mathcal{D}} + \mathbb{H}_{\mathcal{D}} \tag{2.70}$$

approaches zero. Since the target distribution is unknown, this is equivalent of saying that the negative log-likelihood

$$\langle \mathcal{L}_{\boldsymbol{\lambda}} \rangle_{\mathcal{D}} = -\|\mathcal{D}\|^{-1} \sum_{\boldsymbol{\sigma}_k \in \mathcal{D}} \log p_{\boldsymbol{\lambda}}(\boldsymbol{v}_k) \tag{2.71}$$

reaches its minimum value (equal to the negative entropy of the data). This evaluation criterion is somehow problematic, since to estimate the negative log-likelihood we need to know the RBM partition function $Z_{\boldsymbol{\lambda}}$. Depending on the size N of the system, and the insights we have on the target distribution, we choose one of the followings criteria for the grid-search selection of hyper-parameters:

1. *Exact NLL.* When the number of degrees of freedom N is small, we evaluate the partition function $Z_{\boldsymbol{\lambda}}$ exactly by full enumeration. This gives us access to the normalized RBM distribution, and thus to the exact NLL. The best set parameters $\boldsymbol{\lambda}^*$ is simply chosen as the one with minimum NLL computed on a validation set.

2. *Physical observables.* For larger systems, the partition function cannot be directly calculated. Since the objective of the learning are physical states, we have in general a set of physical observables we can probe. As long as such observables can be calculated directly on the input configurations, we can use this information to select the best parameters. Two simple examples are the activation profile over the visible layer, and the correlation function between visible units.

3. *Approximate NLL.* In some situations, the analysis of the physical observables was inconclusive. In this cases, an estimate of the NLL was obtained by using approximate techniques to compute the partition function, such as *parallel tempering* [123] or *annealed importance sampling* [124, 125]. In this case, several Markov chains are run in parallel, with increasing temperature. The bottom chain runs at temperature $\beta = 1$, and corresponds to the RBM distribution. The top chain should run at sufficiently high temperature, so that the partition function of the corresponding distribution can be approximated as $Z_{\boldsymbol{\lambda}}^{top} \simeq 2^{(N+n_h)}$. Provided there are enough chains, so that distributions of adjacent chains overlap, the algorithm can be used to provide an approximation to the partition function $Z_{\boldsymbol{\lambda}}$.

2.4 Spins at thermal equilibrium

We proceed to show a first implementation of generative modelling of a physical system. For now, we restrict to classical states, and examine a magnetic system at finite temperature. Because of the similar structure of the phase space, we consider a collection N magnetic Ising spins $\boldsymbol{\sigma}^z = (\sigma_1^z, \dots, \sigma_N^z)$ $(\sigma_j^z = \pm 1)$ on a lattice. The spins interacts with a Hamiltonian $H(\boldsymbol{\sigma}^z)$, and the system can exchange energy with a thermal bath at inverse temperature $\beta = 1/T$. For a fixed N, the spin system is described in the canonical ensemble by the Boltzmann distribution

$$p_\beta(\boldsymbol{\sigma}^z) = Z_\beta^{-1}\, e^{-\beta H(\boldsymbol{\sigma}^z)} \tag{2.72}$$

with partition function

$$Z_\beta = \sum_{\boldsymbol{\sigma}^z} p(\boldsymbol{\sigma}^z)\,. \tag{2.73}$$

At a given β, the equilibrium state of the system minimizes the free energy $F = U - TS$, where $U = \langle H \rangle$ is the average energy and S is the entropy. Depending on the dimensionality and the structure of the Hamiltonian, different types of orders can appear below a certain critical temperature T_c, dividing two distinct magnetic phases. Given the existence of a

phase transition, of particular importance is the universal behaviour near the critical point, which is quantified by a set of critical exponents. Now imagine, given such a magnetic system, we are capable of measuring the orientation of each spin. We can then build a dataset of spin configurations, where now the unknown distribution underlying the data is the physical distribution $q(\boldsymbol{\sigma}^z) = p_\beta(\boldsymbol{\sigma})$. Therefore, the generative modelling leads (upon faithful training) to an internal representation of the physical distribution. Equivalently, the RBM has learned the partition function of the spin system, an as such, it captures all the thermal physics and allows the scaling calculation of critical exponents.

2.4.1 The Ising model

In the spirit of a first test run, we examine the ferromagnetic Ising model, which is the simplest, yet non-trivial, model of magnetism exhibiting a continuous phase transition. It has been subject to intense studies because it provides a way to understand the emergence of a spontaneous magnetization in some metals (such as iron or nickel), when the temperature is lowered below a critical value, called *Curie temperature*. The Ising spins are placed on the vertices of a d-dimensional cubic lattice with periodic boundaries, and interacts with their nearest neighbours only. The Hamiltonian is

$$H(\boldsymbol{\sigma}^z) = -J \sum_{\langle ij \rangle} \sigma_i^z \sigma_j^z \quad , \quad J > 0 \, . \tag{2.74}$$

which is invariant under time-reversal symmetry (or $\mathbb{Z}_2$), i.e. $H(\boldsymbol{\sigma}^z) = H(-\boldsymbol{\sigma}^z)$. At high temperature, the spins are subject to strong thermal fluctuations, and their orientation is randomly distributed (respecting the symmetry of the Hamiltonian). This is called the *paramagnetic phase*. As we cool down the system, it becomes energetically favourable for the spins to align in the same direction. Below a critical temperature T_c, the system becomes ferromagnetic, which is characterized by a finite magnetization

$$M = \langle \sigma_j^z \rangle = Z_\beta^{-1} \mathrm{Tr} \Big[\sigma_j^z e^{-\beta H(\boldsymbol{\sigma}^z)} \Big] \neq 0 \, , \tag{2.75}$$

In the ground state ($T \to 0$), the system will be in one of the two lowest energy states, where all the spins point either up ($\uparrow, \uparrow, \uparrow, \dots$) or down ($\downarrow, \downarrow, \downarrow, \dots$), breaking time-reversal symmetry. When approaching the critical point from above, one of the two lowest-energy states is randomly selected by thermal fluctuation, and the symmetry is spontaneously broken. The emergence of magnetic order is detected by the order parameter, which is here the magnetization M. In the paramagnetic phase the system has $M \simeq 0$, while $|M| \to 1$ when the system approaches the ferromagnetic ground state.

Let us first consider the simplest case of a 1d chain. In this case, the free energy density can be calculated using a transfer matrix, exactly solving the model. The solution shows that the $1d$ Ising model does not magnetize at any finite temperature (in the thermodynamic limit). The same result can be obtained from considering the entropic contribution of thermal fluctuations [126]. Take either one of the two ferromagnetic ground states at zero temperature, with energy U_0. Now insert a minority droplet of L anti-aligned spins, within the ferromagnetic background (Fig. 2.6a). The new energy of the chain is $U = U_0 + 4J$ (independent of L). When the temperature is non-zero, there is an entropic contribution to the free energy $F = U - TS$, which can be estimated just by counting the number of micro-states where a given spin σ_j^z belongs to the droplet. This number is equal to the droplet size L. Thus, the change in free energy caused by the a droplet of size L is

$$\Delta F = 4J - T \log L \tag{2.76}$$

Therefore, for any $T \neq 0$ there exist a large enough L so that entropy dominates over the energy, leading to a proliferation of droplets. As such, the ferromagnetic order is destroyed for $T > 0$, and the critical temperature of the $1d$ Ising model is then $T_c = 0$. We point out that for a finite size system, there is a range of temperatures (near T_c) where the spin chain exhibits a finite magnetization. This cross-over region becomes smaller when the system size is increased, and disappear in the thermodynamic limit ($T_c \to 0$)

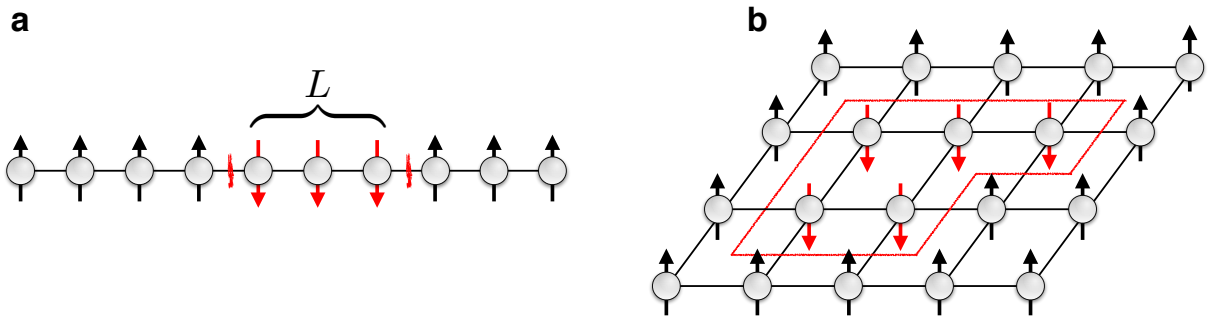

Figure 2.6: **Peierls droplet in the Ising model**. Minority droplets of anti-aligned spins within a ferromagnetic background for a 1d chain (**a**) and 2d square lattice (**b**).

For the case of a $2d$ square lattice, the existence of a phase transition was first established by duality principles [127]. The full solution of the model was given then given by Onsager in 1944 [128], who calculated the exact value of the critical temperature

$$\frac{T_c}{J} = \frac{2}{\log(1+\sqrt{2})} \approx 2.2692\,. \tag{2.77}$$

His solution proved that the free energy can become non-analytic at the critical point, a remarkable milestone for statistical mechanics and the study of critical phenomena. In analogy with the $1d$ chain, we consider again a minority droplet in in the ferromagnetic background. The important difference is that now the energy cost for creating a droplet with a perimeter L is $\Delta U = 2JL$, as we need to break L bonds in the lattice. The bigger the droplet, the higher is the energy cost the system has to pay. The entropic contribution is obtained considering a random walk on the lattice. Under the restrictions of no-backtracking and a closed walk, the free energy difference is

$$\Delta F \simeq 2JL - TL\log C = L(2J - T\log C) \tag{2.78}$$

where $2 < C < 3$ is a constant related to structure of the random walk. We can see now that at sufficiently low (but finite) temperature ($T < T_c = 2J/\log C$), the droplets are suppressed and the ferromagnetic order survives.

2.4.2 Learning thermodynamics

The objective of the experiment is to implement a RBM to learn the ferromagnetic Ising model from spin measurement [73]. Given a fixed temperature, we want to encode the equilibrium distribution of the spins system

$$q_\beta(\boldsymbol{\sigma}^z) = Z_\beta^{-1}\, e^{-\beta H(\boldsymbol{\sigma}^z)} = Z_\beta^{-1}\, e^{-\beta J \sum_{\langle ij\rangle} \sigma_i^z \sigma_j^z} \tag{2.79}$$

into the internal parameters $\boldsymbol{\lambda}$ of the RBM. To do so, we minimized the distance with the RBM distribution

$$p_{\boldsymbol{\lambda}}(\boldsymbol{\sigma}) = Z_{\boldsymbol{\lambda}}^{-1}\, e^{-\mathcal{E}_{\boldsymbol{\lambda}}(\boldsymbol{\sigma})}\,. \tag{2.80}$$

Note that we will always convert the spin degrees of freedom $\sigma^z = \pm 1$ to RBM variables $\sigma = 0, 1$. This is done without loss of generalization, as the former representation can be found simply by rescaling the parameters. It is clear that the problem we are considering is particularly suited for a RBM. This is also suggested by the parametric form of the effective visible energy

$$\mathcal{E}_{\boldsymbol{\lambda}}(\boldsymbol{\sigma}) = -\sum_{j=1}^{N} b_j \sigma_j - \sum_{i=1}^{n_h} \log\left(1 + e^{\sum_j W_{ij}\sigma_j + c_i}\right) \tag{2.81}$$

which contains spin-spin correlations (mediated by the hidden layer) on top of a mean-field contribution $\sum_j b_j \sigma_j$. Due to the similar from of the two distributions, a successful training implies that the RBM has learned an internal representation of the free energy $F_\beta = -\log Z_\beta$ of the spin model.

As an instructive example we begin by training a single network on a $1d$ chain with only $N = 6$ spins at $\beta = 1$. For such a small system it is possible to compute the partition function, and thus the full thermal distribution, exactly. We prepare a dataset of spin configurations by exact sampling of the target distribution $q(\boldsymbol{\sigma}^z)$ and then train a RBM using CD_5. Again, because of the limited number of degrees of freedom, it is feasible to compute the partition function $Z_{\boldsymbol{\lambda}}$ of the RBM by exact enumeration. As such, we have access to the exact KL divergence, which we can calculate at various time-steps of the

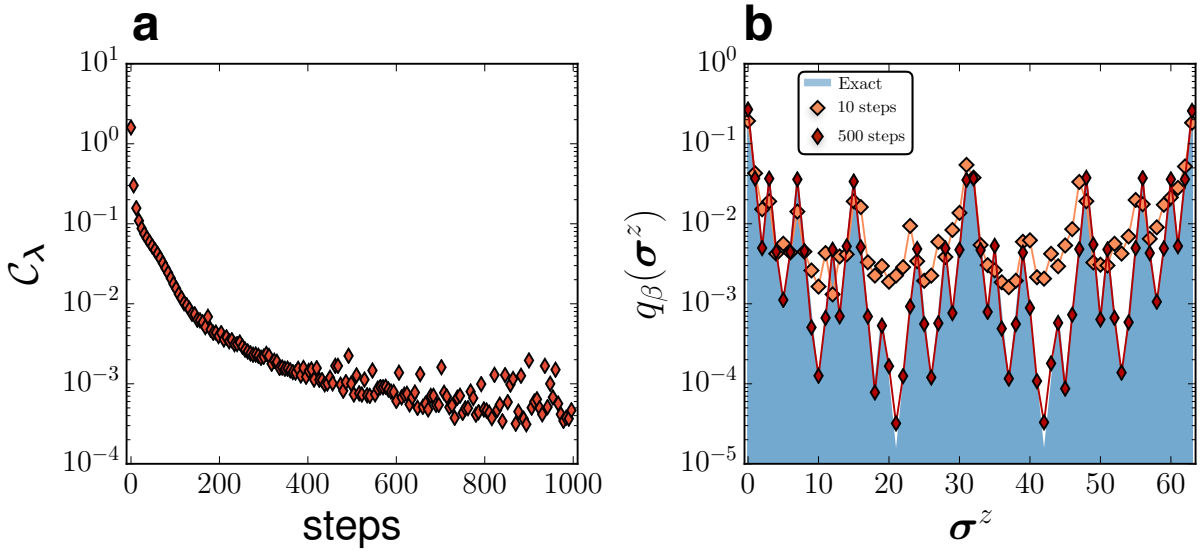

Figure 2.7: **Learning the Ising chain**. **a**) KL divergence as a function of training step. **b**) Probability distributions for a $1d$ Ising model with $N = 6$ spins. We show the comparison between the exact probability distribution (blue) and the approximate distribution produced by the Boltzmann machine after 10 (orange) and 500 (red) training steps for all of the 2^6 states $\boldsymbol{\sigma}^z$.

training. In Fig. 2.7a we show the KL divergence as a function of the training steps. The KL divergence rapidly decreases in the early transient of the optimization, and subsequently reaches a plateau where it starts fluctuating, a sign that the optimization has reached one of the local minima. Since we can evaluate the partition function, we can directly compared the two distributions. In Fig. 2.7b we show this comparison by plotting the probability for each of the 2^6 states $\boldsymbol{\sigma}^z$ of the phase space, measured at two different steps of the training. We observe that, after enough parameters updates, the RBM distribution matches exactly the target distribution.

Next, we consider the more interesting case where the spins are arranged on a $2d$ square lattice. Instead of exact sampling, we use a MC simulation to importance sample spin configurations from the Ising partition function Z_β. We use both the Metropolis-Hastings algorithm in the form o local spin-flips, as well as the Wolff cluster update [129], which reduces autocorrelation time near criticality. Moreover, the cluster update allows

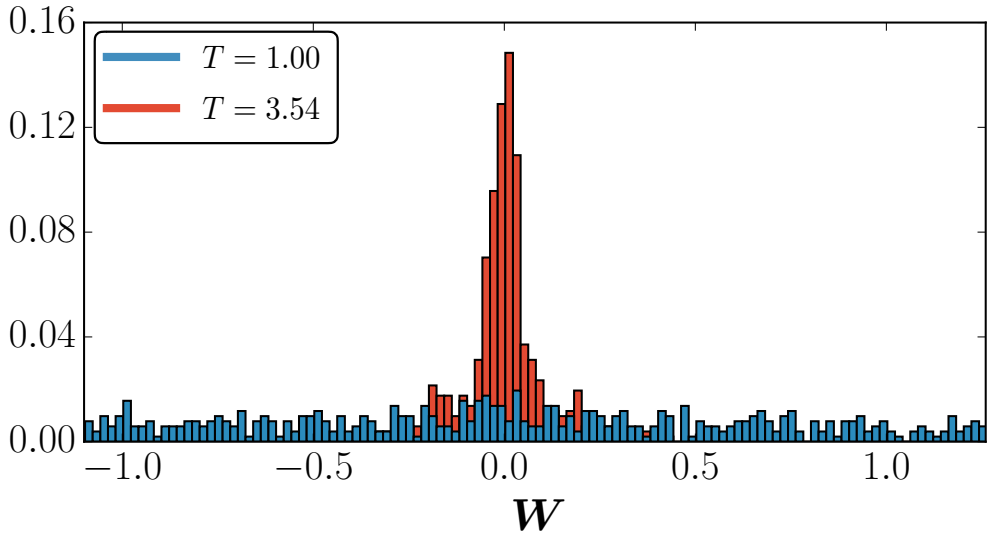

Figure 2.8: **Learned weights for the 2d Ising model**. Histogram of the relative frequency of appearance of the weight amplitudes for two Boltzmann machines with $n_h = 32$ hidden nodes, trained at low and high T for the $d = 2$ Ising model with $N = 64$ spins.

us to generate both symmetry-broken states in the ferromagnetic phase. We prepare a dataset $\mathcal{D}_\beta$ with $\|\mathcal{D}_\beta\| = 10^5$ spin configurations for several inverse temperatures in a range centered around $\beta_c = 1/T_c$. For each β we train a different RBM with CD_{20}, using SGD with a learning rate $\eta = 0.01$, a batch size $M = 50$, an initial weight distribution of $w_0 = 0.001$. The training runs for approximately 2000 epochs [3].

It is natural to ask how the performance of each RBM is affected when the training samples are taken at high or low temperature. Moreover, we are interested in whether or not a RBM is able to properly capture the fluctuations that the system undergoes at criticality. Before discussing the quantitative analysis of the thermodynamics, we give an insight into the functioning of these machines by showing the histogram of the matrix elements of $\boldsymbol{W}$ (Fig. 2.8) after training at low and high temperature. In these two limits we know what the probability distribution $q_\beta(\boldsymbol{\sigma})$ looks like and we can thus obtain a qualitative understanding of the training and sampling processes of the machines. At very

[3] One epoch corresponds to an entire sweep of the dataset, for a total of $\|\mathcal{D}\|/M$ updates.

high temperature $\beta \ll 1$, the spins are completely random, so $q_\beta(\boldsymbol{\sigma}^z) \simeq 2^{-N}$. In this case the weights histogram of the high temperature machine ($T = 3.54$) displays a sharp peak centered around zero, which means that the visible and hidden layers are quasi-decoupled. Thus, for a given hidden configuration $\boldsymbol{h}$, the probability of activating the j-th spin in the visible layer is

$$p_{\boldsymbol{\lambda},\beta}(\sigma_j^z = 1 \,|\, \boldsymbol{h}) = \frac{1}{1 + e^{-\sum_i W_{ij} h_i - b_j}} \simeq \frac{1}{2}\,, \tag{2.82}$$

where we introduced the subscript β in the RBM distribution. Thus, for high temperature, the visible layer is then randomly distributed. On the other hand, at low temperature the two polarized states $\boldsymbol{\sigma} = \mathbf{0}, \mathbf{1}$ are most probable and this causes the histogram to be wide and flat. When we start the sampling we initialize both visible and hidden layers randomly. There is a spontaneous symmetry breaking and the machine chooses one of the two polarizations. If the machine chooses the visible state $\boldsymbol{\sigma} = \mathbf{1}$ after equilibration, we find, by inspecting the hidden states driving the spins, that the hidden layer is arranged such that only the nodes connected to the positive weights are active (and similarly for the opposite state). The activations will be in this case large and positive and thus $p_{\boldsymbol{\lambda}}(\boldsymbol{\sigma}^z = \mathbf{1} \,|\, \boldsymbol{h}) \simeq 1$. Note that, even though the dataset is ergodic, once the visible layer has equilibrated into one polarization state, it is unlikely to switch to the other. This ergodicity issue is analogous to that faced by local Metropolis-Hastings updates in MC simulations of the low-temperature ferromagnet.

We turn now to discuss performance on the full range of temperatures. Since, for general spin Hamiltonians with a large N, it is very challenging to compute the partition function and thus the KL divergence, we instead characterize the performance of the machine using Ising thermodynamics observables. Given an observable $\mathcal{O}_\beta$ defined on the spin system, we can compare its expectation value computed on the spins in the dataset at inverse temperature β,

$$\langle \mathcal{O}_\beta \rangle_{\mathcal{D}_\beta} = \|\mathcal{D}_\beta\|^{-1} \sum_{\boldsymbol{\sigma}_k^z \in \mathcal{D}_\beta} \mathcal{O}_\beta(\boldsymbol{\sigma}_k^z), \tag{2.83}$$

with the average computed on the spin samples produced by the RBM trained on $\mathcal{D}_\beta$. After training, we can initialize this machine with a random configuration and perform block Gibbs sampling until equilibration. The expectation value of the observable $\mathcal{O}_\beta$

evaluated on the RBM is given by

$$\langle \mathcal{O}_\beta \rangle_{p_{\boldsymbol{\lambda},\beta}} = Z^{-1}_{\boldsymbol{\lambda},\beta} \sum_{\boldsymbol{\sigma}} \mathcal{O}_\beta(\boldsymbol{\sigma})\, \mathrm{e}^{-\mathcal{E}_{\boldsymbol{\lambda},\beta}(\boldsymbol{\sigma})} \simeq M_C^{-1} \sum_{k=1}^{M_C} \mathcal{O}_\beta(\boldsymbol{\sigma}_k)\,, \tag{2.84}$$

where the configurations $\boldsymbol{\sigma}_k$ are drawn from the conditional RBM distributions via block Gibbs sampling. If the machine is properly trained we expect the deviations $\delta\mathcal{O}_\beta = |\langle \mathcal{O}_\beta \rangle_{\mathcal{D}_\beta} - \langle \mathcal{O}_\beta \rangle_{p_{\boldsymbol{\lambda},\beta}}|$ to be small. Here we examine the magnetization $M = \langle \sum_i \sigma_i \rangle / N$, the energy U/N and their corresponding fluctuations: the susceptibility $\chi = (\langle M^2 \rangle - \langle M \rangle^2)/(NT)$ and the specific heat $C_V = (\langle E^2 \rangle - \langle E \rangle^2)/(NT^2)$. The thermodynamic observables, measured from different system size, provide enough information to extract the critical exponents of the model (and thus the universality class), fully characterizing its critical behaviour. We plot the comparison of these observables in Fig. 2.9. For the magnetization, we find that even with a number of hidden units as low as two (not shown), the machine is able to reproduce the exact behaviour within statistical error. This may be expected as it is the spin state itself that the Boltzmann machine is being trained on. One may therefore expect a similar accuracy on the correlation length ξ, which can be defined in terms of the Fourier transform of the correlation function

$$S(\boldsymbol{k}) = \sum_{\boldsymbol{r}} \cos(\boldsymbol{k} \cdot \boldsymbol{r}_{ij}) \langle \sigma_i^z \sigma_j^z \rangle\,, \tag{2.85}$$

also called *static structure factor*. In particular, the correlation length can be constructed from the structure factor at $\boldsymbol{q}_0 = (0, 0)$ (i.e. the wave-vector of the dominant correlation for a ferromagnet) and $\boldsymbol{q}_1 = (2\pi/L, 0)$ as follows [130] :

$$\xi = \frac{L}{2\pi} \sqrt{\frac{S(\boldsymbol{q}_0)}{S(\boldsymbol{q}_1)} - 1} \tag{2.86}$$

As shown in the inset of Fig. 2.9b, the accuracy for the correlation length is indeed very high – only slightly worse than the magnetization for a small number of hidden units. In contrast, in the case of the energy, even though we are computing its value using the Ising

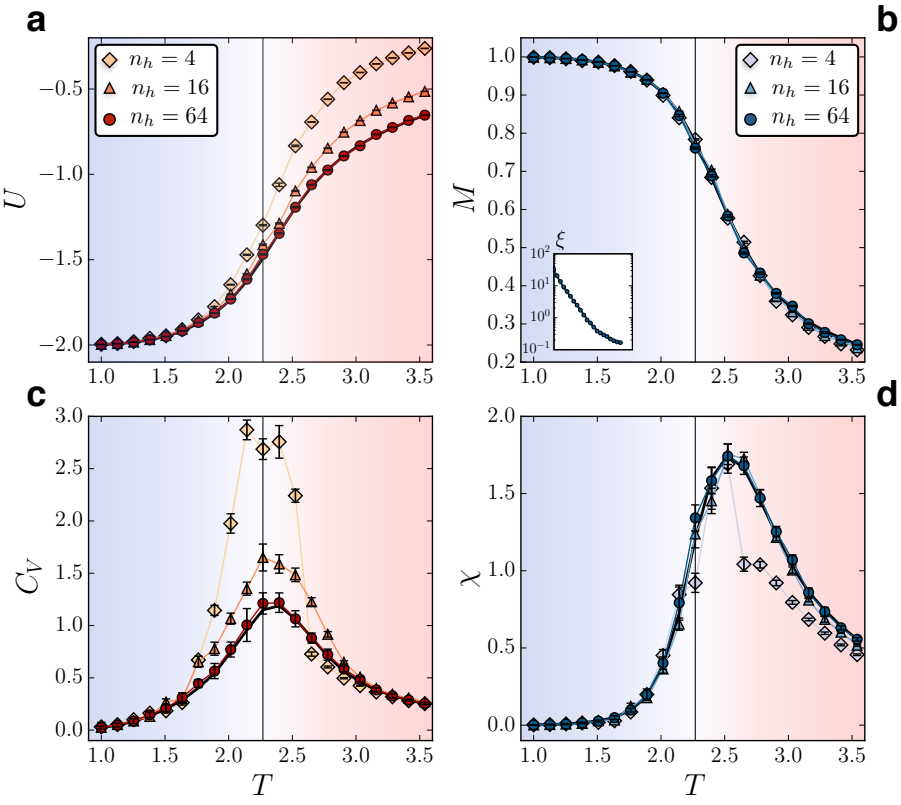

Figure 2.9: **Thermodynamic observables**. Comparison of the observables generated with the Boltzmann machine with the exact values calculated from the dataset (black) for a $d = 2$ Ising system with $N = 64$ spins. The observables considered are energy (**a**), magnetization (**b**), specific heat (**c**) and magnetic susceptibility (**d**). We show the results for Boltzmann machines with hidden nodes $n_H = 4$, $n_H = 16$ and $n_H = 64$.

Hamiltonian (applied to the visible units), information about the local energy constraints is not directly included in the training dataset. This results in a larger discrepancy between the physical value of the energy and that generated with the RBM.

Most interestingly, it appears that for a given physical system size N, the Boltzmann machine with a fixed n_h learns best away from criticality. In Fig. 2.10a we plot the scaling of the specific heat with the number of hidden units in the machine for five different temperatures. When the system is in an ordered or a disordered state, the machines trained

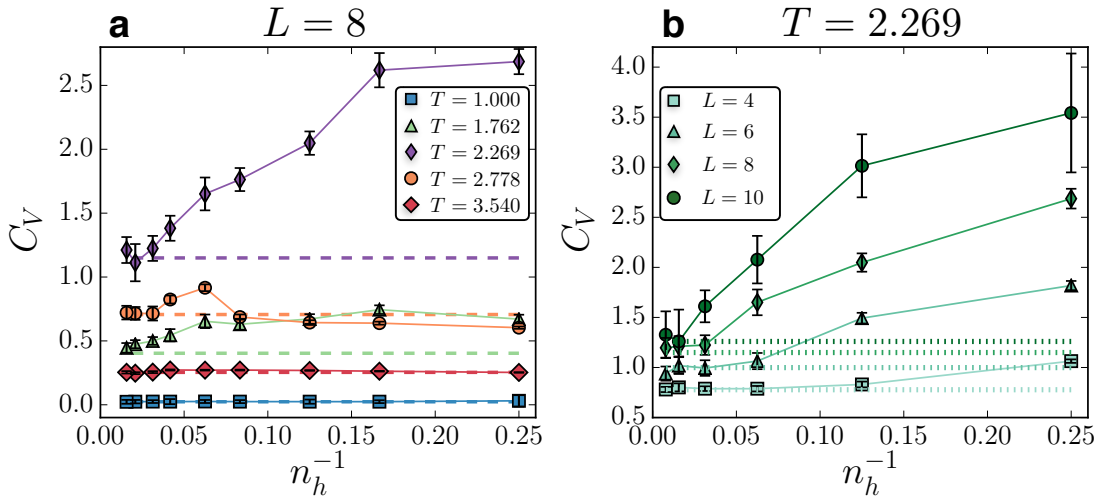

Figure 2.10: **Finite-size scaling for the specific heat**. C_V with the number of hidden nodes n_H. In (**a**) we show scaling at different temperatures T, when the system is ordered, disordered and critical. In (**b**) we show the scaling at criticality for different systems sizes L. Dotted lines represent the exact value computed on the training dataset.

on the spins of the corresponding datasets are able to reproduce the exact values within statistical error, irrespective to n_h. This is consistent with the weight histograms in Fig. 2.8. At high temperature this follows from the two layers being quasi decoupled. For low temperatures we have seen that only the hidden nodes that connect to positive weights (or negative weights, depending on the polarization of the visible layer) are set to 1; increasing the number of hidden nodes will not affect the activation of the visible units. Finally, when the system is at criticality, it is still possible to obtain an accurate approximation of the physical distribution, however a clear dependency on the finite number of hidden units appears. As illustrated in Fig. 2.10a, in order to converge the specific heat at the critical point, the required n_h is significantly larger than for T far above or below the transition. We also note that the same scaling plot for the magnetization shows no clear dependencies on n_h. Finally, we show in Fig. 2.10b the scaling curves at criticality for different system sizes. As expected, the threshold in the number of hidden units required for faithful learning of the specific heat grows with increasing N.

2.5 Conclusions

In this Chapter, we have overviewed a stochastic neural network called a restricted Boltzmann machine. We have shown various favourable properties of this probabilistic graphical model, such as the the intra-layer conditional independence, which allows an efficient stochastic sampling of neurons configurations. Most importantly, the natural interpretation of a RBM within the framework of statistical mechanics makes the network very appealing for exploring the physics of many-particle systems. From the perspective of the representational power of the network, a RBM is capable of describing any function, provided enough number of neurons [131], analogously to the bond dimension of a MPS. Regarding unsupervised learning, there exist efficient training algorithms, such as the CD, to build internal representations of unknown distribution from raw data.

We selected the easiest non-trivial model of magnetism, the Ising model, and performed numerical simulations where we trained a set of RBMs on Ising data generated with MC at various temperatures. For a small system in $1d$, we confirmed through an exact calculation that the RBM converges to the physical probability distribution with sufficient training steps. For the case of $2d$, where exact calculations are not feasible, we compared thermodynamic observables produced by the RBM to those calculated directly by MC. We observed that away from criticality, the RBM is able to capture the thermodynamics with only a few hidden units, while the large fluctuations near the critical points required a large amount of units to learn the distribution. Finally, the performance of a RBM may be evaluated using a comparison of thermodynamic observables calculated from both the physical and modelled distribution. The conceptual elimination of reliance on the KL divergence may suggest alternatives to evaluating the performance of such machines in other applications.

The RBM easily learned a non-trivial distribution of classical spins. The result of this proof-of-principle experiment is most welcomed, as we can now entertain the possibility of learning harder distributions underlying quantum data. In order to capture quantum mechanical properties with RBMs, we first need to introduce an appropriate parametrization of quantum states of matter.

Chapter 3

Neural-network quantum states

In this Chapter, we introduce a class of quantum states based on the natural connection between generative models and the probabilistic nature of quantum mechanics. We will start with the simple case of positive wavefunction in Sec. 3.1, and discuss the presence of phase structures in Sec. 3.1.1. We will then consider in Sec. 3.2 the case of mixed states, and build a neural-network representation of a density matrix, using on a purification of the physical Hilbert space. We discuss in Sec. 3.3 the measurement process of generic observables, and the calculation of entanglement entropy using a two replicated RBMs.

3.1 Neural wavefunctions

We start by considering pure quantum states, described by a wave-function $|\Psi\rangle$. We choose a reference basis $|\boldsymbol{\sigma}\rangle = |\sigma_1, \dots, \sigma_N\rangle$ of the full Hilbert space $\boldsymbol{\mathcal{H}}$, where $\sigma_j = 0, 1$ [1]. The goal is to build a parametric representation $\psi_{\boldsymbol{\lambda}}(\boldsymbol{\sigma})$ of the wavefunction, with $\boldsymbol{\lambda}$ the parameters of a RBM. The specific choice of neural network is not unique, and many other configurations, such as connectivity and type of activations, could be considered. We will restrict to a RBM since we believe it to be an appropriate representation to capture physical states.

[1] This could for instance corresponds to the quantum number $\sigma^z = \pm 1$.

More importantly, the probabilistic model defined by the RBM provides a natural way to define a quantum state, which follows from the inherently probabilistic nature of the measurement process in quantum mechanics.

The RBM representation of a pure quantum state is defined in terms of the outcomes of a projective measurement on the state $|\Psi\rangle$. In this process, each degrees of freedom σ_j is independently measured, with outcome $\sigma_j = 0, 1$. The output of a measurement is a N-bit string $\boldsymbol{\sigma}$, and the probability of this outcome is given by the Born rule:

$$P(\boldsymbol{\sigma}) = |\langle \boldsymbol{\sigma}|\Psi\rangle|^2 = |\Psi(\boldsymbol{\sigma})|^2 \,. \tag{3.1}$$

We make now the assumption that the wavefunction is (or can be gauged to be) positive:

$$\Psi(\boldsymbol{\sigma}) \in \mathbb{R} \qquad , \qquad \Psi(\boldsymbol{\sigma}) \geq 0 \;\; \forall\, |\boldsymbol{\sigma}\rangle \in \boldsymbol{\mathcal{H}} \,. \tag{3.2}$$

While this condition might feel quite restrictive, it is nonetheless true for several cases of interest, such as many quantum spin models on bipartite lattices and bosonic systems. In this case, where there is no sign or phase structure, the probability distribution $P(\boldsymbol{\sigma})$ fully characterizes the quantum state:

$$\Psi(\boldsymbol{\sigma}) = \sqrt{P(\boldsymbol{\sigma})} \,. \tag{3.3}$$

This property allows us to establish the connection between the quantum state and the generative model, defining the *positive neural wavefunction* as the square root of the generative model probability distribution:

Definition 3.1. *Given a Hilbert space $\boldsymbol{\mathcal{H}}$ of a N-particle system spanned by the basis $|\boldsymbol{\sigma}\rangle = |\sigma_1, \ldots, \sigma_N\rangle$, the* **positive neural wavefunction** *is a mapping $\boldsymbol{\psi_\lambda} : \boldsymbol{\mathcal{H}} \to \mathbb{R}$ that associates to each basis state $|\boldsymbol{\sigma}\rangle \in \boldsymbol{\mathcal{H}}$ a real and positive value:*

$$\psi_{\boldsymbol{\lambda}}(\boldsymbol{\sigma}) = \sqrt{p_{\boldsymbol{\lambda}}(\boldsymbol{\sigma})} = Z_{\boldsymbol{\lambda}}^{-\frac{1}{2}}\, e^{-\mathcal{E}_{\boldsymbol{\lambda}}(\boldsymbol{\sigma})/2} \,. \tag{3.4}$$

The wavefunction parameters are $\boldsymbol{\lambda} = \{\boldsymbol{W}, \boldsymbol{b}, \boldsymbol{c}\}$.

Now that the neural-network representation of a wavefunction is well defined, a natural question is what classes of quantum states can be efficiently represented with this ansatz. As far for the efficient representation, a RBM with enough number of hidden units can represent any probability distribution [131]. Clearly, the worst case scenario corresponds to a neural wavefunction with $n_h = 2^N$ hidden units, which exactly encodes each coefficient $\Psi(\boldsymbol{\sigma})$ in the parameters $\boldsymbol{\lambda}$. Therefore, depending on the complexity of the quantum state to be encoded, an increasing amount of network connections are required to faithfully capture its properties. Consequently, the number of total parameter ($\dim[\boldsymbol{\lambda}]$) naturally quantifies the convergence of this representation. We will loosely refer to efficient representation when the number of hidden units required to achieve a certain performance threshold (i.e. fidelity) is of the same order of magnitude of the number of physical degrees of freedom N.

3.1.1 Phase structure

The positivity of the wavefunction $|\Psi\rangle$ leads to a simple interpretation of the quantum state in terms of a classical probabilistic model. This implies that, if there exists a set of parameters $\boldsymbol{\lambda}$ of a RBM such that $\psi_{\boldsymbol{\lambda}}(\boldsymbol{\sigma}) \simeq \Psi(\boldsymbol{\sigma})$ (according to some chosen metric), then all the quantum correlations of the state $|\Psi\rangle$ have been encoded into a classical Ising Hamiltonian $E_{\boldsymbol{\lambda}}$. This also implies that the physical properties of the system described by $|\Psi\rangle$ can be computed equivalently by considering the corresponding classical Ising system at thermal equilibrium (this will be discuss later in the chapter). In fact, the distribution of the states visited by the classical system at equilibrium matches the distribution of the outcome of a set of projective measurements on the quantum state. In other words, if such set of parameters exists, the two system are essentially indistinguishable, as far as measurements of physical observables are concerned. This scenario drastically when the state $|\Psi\rangle$ has a sign or a phase structure [2]. We can now write the state as

$$|\Psi\rangle = \sum_{\boldsymbol{\sigma}} \Phi(\boldsymbol{\sigma})\, e^{i\phi(\boldsymbol{\sigma})} |\boldsymbol{\sigma}\rangle \,, \tag{3.5}$$

[2] As seen in Chapter 1, this means the the coefficients $\Psi(\boldsymbol{\sigma})$ of the quantum state can be either positive or negative, as well as complex.

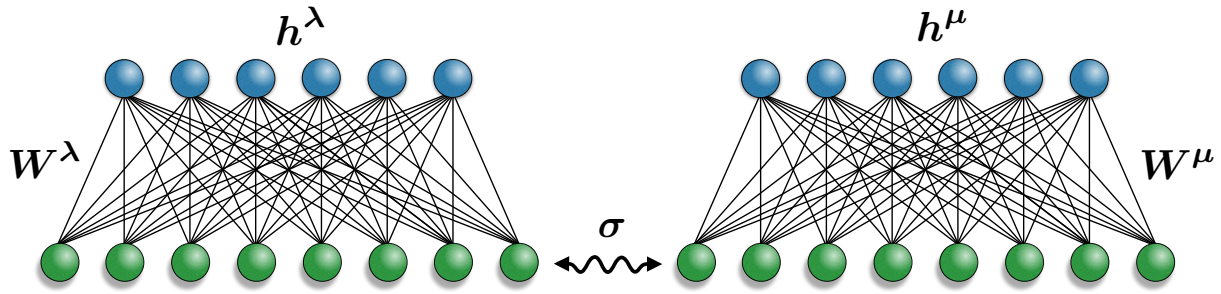

Figure 3.1: **Complex neural wavefunction**. The parametrization of a wavefunction with complex coefficient is realized using two RBMs with different parameters $\boldsymbol{\lambda}$ and $\boldsymbol{\mu}$, parametrizing the modulus and phase respectively. At any point of the simulation/training, the state of the visible layer is synched between the two networks.

which is the most general pure quantum state. This is the case for quantum systems undergoing unitary dynamics, ground state of frustrated magnets and output states of generic quantum circuits.

The introduction of a phase structure increases the classical resources required to represent the state. Now a single RBM with distribution $p_{\boldsymbol{\lambda}}(\boldsymbol{\sigma})$ is clearly unable to encode the state $|\Psi\rangle$. One possible solution to introduce the complex nature of the coefficients is to use a RBM with complex weights. The resulting ansatz has been proposed by Carleo in the context of the variational optimization of ground state energy of quantum spin models [74]. While this representation has proved to be efficient in many cases of study, it would lead to complex-valued conditional probabilities, disabling the block Gibbs sampling mechanism. In light of this, we decide to keep the RBM parameters $\boldsymbol{\lambda}$ real, and add an additional set of (real) parameters $\boldsymbol{\mu}$ to represent the phase. In particular, we choose a parametrization using the effective visible energy as the phase of the wavefunction: $\phi_{\boldsymbol{\mu}}(\boldsymbol{\sigma}) = -\mathcal{E}_{\boldsymbol{\mu}}(\boldsymbol{\sigma})$. The resulting neural wavefunction is built with the $\boldsymbol{\lambda}$-RBM parametrizing the amplitudes, coupled with the $\boldsymbol{\mu}$-RBM capturing the phase structure. The state of the visible layer of the two machines is always kept in synch between one another (Fig. 3.1).

Definition 3.2. *Given a Hilbert space $\boldsymbol{\mathcal{H}}$ of a N-particle system spanned by the basis $|\boldsymbol{\sigma}\rangle = |\sigma_1, \ldots, \sigma_N\rangle$, the* **neural wavefunction** *is a mapping $\boldsymbol{\psi_{\lambda\mu}} : \boldsymbol{\mathcal{H}} \to \mathbb{C}$ that associates to each basis state $|\boldsymbol{\sigma}\rangle \in \boldsymbol{\mathcal{H}}$ an amplitude $\sqrt{p_{\boldsymbol{\lambda}}(\boldsymbol{\sigma})}$ and a phase $\mathcal{E}_{\boldsymbol{\mu}}(\boldsymbol{\sigma})$:*

$$\psi_{\boldsymbol{\lambda\mu}}(\boldsymbol{\sigma}) \equiv \sqrt{p_{\boldsymbol{\lambda}}(\boldsymbol{\sigma})}\, e^{i\phi_{\boldsymbol{\mu}}(\boldsymbol{\sigma})} = Z_{\boldsymbol{\lambda}}^{-\frac{1}{2}}\, e^{-(\mathcal{E}_{\boldsymbol{\lambda}}(\boldsymbol{\sigma}) + i\mathcal{E}_{\boldsymbol{\mu}}(\boldsymbol{\sigma}))/2} \tag{3.6}$$

The network parameters are $(\boldsymbol{\lambda}, \boldsymbol{\mu}) = \{\boldsymbol{W^{\lambda}}, \boldsymbol{b^{\lambda}}, \boldsymbol{c^{\lambda}}, \boldsymbol{W^{\mu}}, \boldsymbol{b^{\mu}}, \boldsymbol{c^{\mu}}\}$.

An important property of this parametrization is that the the probability distribution of a projective measurements only depends on one sub-set of the RBMs parameters:

$$P(\boldsymbol{\sigma}) = \left| \sqrt{p_{\boldsymbol{\lambda}}(\boldsymbol{\sigma})}\, e^{i\phi_{\boldsymbol{\mu}}(\boldsymbol{\sigma})} \right|^2 = p_{\boldsymbol{\lambda}}(\boldsymbol{\sigma})\,. \tag{3.7}$$

This means that when running the Markov chain to sample configurations $|\boldsymbol{\sigma}\rangle$, whether during the training or to generate sample in a new simulation, only the hidden layer connected with $\boldsymbol{\lambda}$ parameters is updated by block Gibbs sampling, while the other hidden layer (with parameters $\boldsymbol{\mu}$) is left idling.

3.2 Neural density operators

We now turn to the case of quantum states whose purity cannot be assumed. We wish to extend to neural-network approach applied to the wavefunction to the general case of mixed states. In other words, we want to find a neural-network representation of a density matrix, which we will refer to as *neural density operator* (NDO). In analogy to the pure state, we define the NDO as a mapping $\hat{\rho}_{\boldsymbol{\theta}}$ that, given two input states $|\boldsymbol{\sigma}\rangle$ and $|\boldsymbol{\sigma}'\rangle$, returns the matrix element $\rho_{\boldsymbol{\theta}}(\boldsymbol{\sigma}, \boldsymbol{\sigma}')$. The set $\boldsymbol{\theta}$ contains the network parameters, and is yet to be determined. Contrary to the case of the neural wavefunction, we have several requirements that the NDO must satisfy in order to describe physical states, which leads in turn to a more complex parametrization. In fact, the neural wavefunction representation

is essentially unconstrained, with the only requirement being the normalization

$$\sum_{\boldsymbol{\sigma}} |\psi_{\boldsymbol{\lambda}\boldsymbol{\mu}}(\boldsymbol{\sigma})|^2 = 1 \,. \tag{3.8}$$

Clearly, this condition can be trivially enforced with a NDO defined as

$$\rho_{\boldsymbol{\theta}}(\boldsymbol{\sigma}, \boldsymbol{\sigma}') = Z_{\boldsymbol{\theta}}^{-1} \tilde{\rho}_{\boldsymbol{\theta}}(\boldsymbol{\sigma}, \boldsymbol{\sigma}') \,, \tag{3.9}$$

where $Z_{\boldsymbol{\theta}} = \sum_{\boldsymbol{\sigma}} \tilde{\rho}_{\boldsymbol{\theta}}(\boldsymbol{\sigma}, \boldsymbol{\sigma})$ is the normalization constant. The self-adjoint condition is also easy to encode. For instance, we could define the NDO using two real matrices $\boldsymbol{f} = \boldsymbol{f}^\top$ and $\boldsymbol{g} = -\boldsymbol{g}^\top$, in the following way:

$$\tilde{\rho}_{\boldsymbol{\theta}}(\boldsymbol{\sigma}, \boldsymbol{\sigma}') = f_{\boldsymbol{\theta}}(\boldsymbol{\sigma}, \boldsymbol{\sigma}') + i g_{\boldsymbol{\theta}}(\boldsymbol{\sigma}, \boldsymbol{\sigma}') \,, \tag{3.10}$$

which clearly respect the hermitian condition. In contrast, there is no simple strategy to constrain these two matrices in such a way that the density matrix result to be positive semi-definite. A simple and straightforward solution consists of introducing an hermitian operator $\hat{T} = \hat{T}^\dagger$ and define the density operator as $\tilde{\rho} = \hat{T}^\dagger \hat{T}$, which is positive semi-definite by construction. In fact, for any state $|\boldsymbol{\xi}\rangle$ one obtains

$$\langle \boldsymbol{\xi} | \hat{\rho} | \boldsymbol{\xi} \rangle = \langle \boldsymbol{\xi} | \hat{T}^\dagger \hat{T} | \boldsymbol{\xi} \rangle = |\hat{T} | \boldsymbol{\xi} \rangle|^2 \geq 0 \tag{3.11}$$

Unfortunately, this representation is not efficient, in the sense that the evaluation of one element of the density matrix

$$\tilde{\rho}_{\boldsymbol{\theta}}(\boldsymbol{\sigma}, \boldsymbol{\sigma}') = \sum_{\boldsymbol{\sigma}''} T^*_{\boldsymbol{\sigma}''\boldsymbol{\sigma}} T_{\boldsymbol{\sigma}''\boldsymbol{\sigma}'} \tag{3.12}$$

requires a summation over an exponentially large number of states. Since our goal is to obtain a scalable representation of the state, we instead consider a purification procedure, where we enlarge the Hilbert space so that the state of the new system is pure, and the density operator is simply obtained by tracing out the additional degrees of freedom.

Lemma 3.1. *Given a density operator $\hat{\rho}_A$ living on the Hilbert space $\mathcal{H}_A$, it is always possible to introduce an auxiliary Hilbert space $\mathcal{H}_B$ with dimension $dim(\mathcal{H}_B) \geq dim(\mathcal{H}_A)$, and a pure state $|\Psi\rangle \in \mathcal{H}_A \otimes \mathcal{H}_B$, such that*

$$\hat{\rho}_A = Tr_B(|\Psi\rangle\langle\Psi|) \tag{3.13}$$

The state $|\Psi\rangle$ is not unique and it is called a purification of the physical state $\hat{\rho}_A$, while the auxiliary Hilbert space has in general no physical meaning. [18]

In the context of the neural-network representation, the purification procedure translates into adding a new set of variables $\boldsymbol{a} = (a_1, \ldots, a_{n_a})$ with $n_a \geq N$ (in general), so that we can parametrize the composite system with a neural wavefunction

$$|\psi_{\boldsymbol{\theta}}\rangle = \sum_{\boldsymbol{\sigma a}} \psi_{\boldsymbol{\theta}}(\boldsymbol{\sigma}, \boldsymbol{a})|\boldsymbol{\sigma}\rangle \otimes |\boldsymbol{a}\rangle \,, \tag{3.14}$$

for some set of internal parameters $\boldsymbol{\theta}$. The NDO is then obtained by tracing out the auxiliary units from the composite density operator $\hat{\rho}_{\boldsymbol{\theta}}^{\boldsymbol{\sigma}\oplus\boldsymbol{a}} = |\psi_{\boldsymbol{\theta}}\rangle\langle\psi_{\boldsymbol{\theta}}|$

$$\hat{\rho}_{\boldsymbol{\theta}} = \mathrm{Tr}_{\boldsymbol{a}}(|\psi_{\boldsymbol{\theta}}\rangle\langle\psi_{\boldsymbol{\theta}}|) \,, \tag{3.15}$$

obtaining the density matrix

$$\rho_{\boldsymbol{\theta}}(\boldsymbol{\sigma}, \boldsymbol{\sigma}') = \sum_{\boldsymbol{a}} \psi_{\boldsymbol{\theta}}(\boldsymbol{\sigma}, \boldsymbol{a})\psi^*_{\boldsymbol{\theta}}(\boldsymbol{\sigma}', \boldsymbol{a}) \,. \tag{3.16}$$

Note that in this expression we have once again a summation running over 2^{n_a} states, and thus, it is in still not scalable. But the RBM provides a very convenient method to overcome this problem. The summation over the auxiliary units can be in fact performed analytically if we encode these degrees of freedom in the latent space of the neural network, leading to a scalable parametrization of the density operator.

3.2.1 Latent space purification

Let us consider again a neural wavefunction defined previously

$$\psi_{\boldsymbol{\lambda}\boldsymbol{\mu}}(\boldsymbol{\sigma}) \equiv \sqrt{p_{\boldsymbol{\lambda}}(\boldsymbol{\sigma})}\, e^{i\phi_{\boldsymbol{\mu}}(\boldsymbol{\sigma})} = Z_{\boldsymbol{\lambda}}^{-\frac{1}{2}}\, e^{-(\mathcal{E}_{\boldsymbol{\lambda}}(\boldsymbol{\sigma})+i\mathcal{E}_{\boldsymbol{\mu}}(\boldsymbol{\sigma}))/2}\,, \tag{3.17}$$

consisting of two RBMs synched with each other, one to parametrize the modulus and the other the phase. In particular, recall the parametric form of the effective energy:

$$\mathcal{E}_{\boldsymbol{\theta}}(\boldsymbol{\sigma}) = -\log p_{\boldsymbol{\theta}}(\boldsymbol{\sigma}) = -\sum_j b_j^{\boldsymbol{\theta}}\sigma_j - \sum_i \log\left[1+\exp\Big(\sum_j W_{ij}^{\boldsymbol{\theta}}\sigma_j + c_i^{\boldsymbol{\theta}}\Big)\right], \tag{3.18}$$

for any of the two RBMs, $\boldsymbol{\theta} = \boldsymbol{\lambda}, \boldsymbol{\mu}$. We now add the new hidden layer $\boldsymbol{a} = (a_1, \ldots, a_{n_a})$ to each of the two RBMs in the neural wavefunction. These unit represent the additional (unphysical) degrees of freedom used in the purification. We couple this new layers with the visible layers using a new interaction term $\boldsymbol{V}^{\boldsymbol{\theta}}$ (and a new bias $\boldsymbol{d}^{\boldsymbol{\theta}}$). With this modification, we obtain the new new effective energy

$$\Upsilon_{\boldsymbol{\theta}}(\boldsymbol{\sigma}, \boldsymbol{a}) = -\sum_j b_j^{\boldsymbol{\theta}}\sigma_j - \sum_i \log\left[1+\exp\Big(\sum_j W_{ij}^{\boldsymbol{\theta}}\sigma_j + c_i^{\boldsymbol{\theta}}\Big)\right] - \sum_{jk} V_{jk}^{\boldsymbol{\theta}} a_k \sigma_j - \sum_k d_k^{\boldsymbol{\theta}} a_k\,. \tag{3.19}$$

This is equivalent of simply enlarging the latent space of the network, and as such, all the fundamental properties of the RBMs are left unchanged. Therefore, the NDO is built using two RBMS, as a neural wavefunction, where now the latent space has been increased with the auxiliary degrees of freedom (Fig. 3.2).

We introduced additional units in the neural network, so that we can now describe the quantum state of the composite system with a pure state wavefunction. In our case, we implement a neural wavefunction with a set of parameters $(\boldsymbol{\lambda}, \boldsymbol{\mu})$, defined as

$$\psi_{\boldsymbol{\lambda}\boldsymbol{\mu}}(\boldsymbol{\sigma}, \boldsymbol{a}) = Z_{\boldsymbol{\lambda}}^{-\frac{1}{2}}\, e^{-(\Upsilon_{\boldsymbol{\lambda}}(\boldsymbol{\sigma},\boldsymbol{a})+i\Upsilon_{\boldsymbol{\mu}}(\boldsymbol{\sigma},\boldsymbol{a}))/2} \tag{3.20}$$

where $Z_{\boldsymbol{\lambda}} = \sum_{\boldsymbol{\sigma}\boldsymbol{a}} p_{\boldsymbol{\lambda}}(\boldsymbol{\sigma}, \boldsymbol{a})$ is the partition function of the $\boldsymbol{\lambda}$-RBM.

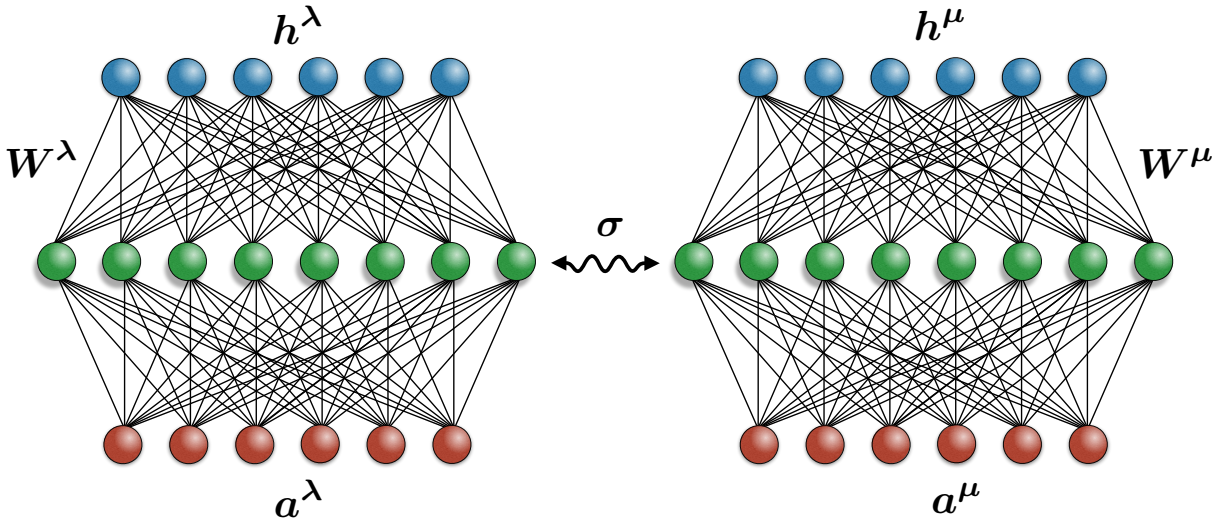

Figure 3.2: **Neural density operator**. Graphical representation of the neural density operator. The visible layer (green) encodes the state of the physical system $\boldsymbol{\sigma}$, which is shared between the modulus and the phase machines. The hidden layers (blue) are implemented to represent the correlation within the physical system, while the purification layer (red) is used to capture the coupling with the environment.

We now proceed to derive an analytical expression for the NDO, so that we can access the matrix elements efficiently. From the definition above, the NDO is given by

$$\hat{\rho}_{\boldsymbol{\lambda}\boldsymbol{\mu}} = \sum_{\boldsymbol{\sigma}\boldsymbol{\sigma}'} \rho_{\boldsymbol{\lambda}\boldsymbol{\mu}}(\boldsymbol{\sigma},\boldsymbol{\sigma}')|\boldsymbol{\sigma}\rangle\langle\boldsymbol{\sigma}'|\,, \tag{3.21}$$

with matrix elements

$$\begin{aligned}\rho_{\boldsymbol{\lambda}\boldsymbol{\mu}}(\boldsymbol{\sigma},\boldsymbol{\sigma}') &= \sum_{\boldsymbol{a}} \psi_{\boldsymbol{\lambda}\boldsymbol{\mu}}(\boldsymbol{\sigma},\boldsymbol{a})\psi^{*}_{\boldsymbol{\lambda}\boldsymbol{\mu}}(\boldsymbol{\sigma}',\boldsymbol{a}) \\ &= Z_{\boldsymbol{\lambda}}^{-1}\sum_{\boldsymbol{a}} e^{-(\Upsilon_{\boldsymbol{\lambda}}(\boldsymbol{\sigma},\boldsymbol{a})+\Upsilon_{\boldsymbol{\lambda}}(\boldsymbol{\sigma}',\boldsymbol{a}))/2}\, e^{-i(\Upsilon_{\boldsymbol{\mu}}(\boldsymbol{\sigma},\boldsymbol{a})-\Upsilon_{\boldsymbol{\mu}}(\boldsymbol{\sigma}',\boldsymbol{a}))/2} \\ &\equiv Z_{\boldsymbol{\lambda}}^{-1}\widetilde{\rho}_{\boldsymbol{\lambda}\boldsymbol{\mu}}(\boldsymbol{\sigma},\boldsymbol{\sigma}')\,.\end{aligned} \tag{3.22}$$

By plugging in the expression for the effective energies we obtain

$$\begin{aligned}\widetilde{\rho}_{\boldsymbol{\lambda}\boldsymbol{\mu}}(\boldsymbol{\sigma},\boldsymbol{\sigma}') = \sum_{\boldsymbol{a}} \exp\Bigg\{ & \sum_j b_j^{\boldsymbol{\lambda}}(\sigma_j+\sigma_j') + \sum_i \log\Big[1+\exp\Big(\sum_j W_{ij}^{\boldsymbol{\lambda}}\sigma_j + c_i^{\boldsymbol{\lambda}}\Big)\Big] \\ & + \sum_i \log\Big[1+\exp\Big(\sum_j W_{ij}^{\boldsymbol{\lambda}}\sigma_j' + c_i^{\boldsymbol{\lambda}}\Big)\Big] \\ & + \sum_k a_k\Big(\sum_j V_{jk}^{\boldsymbol{\lambda}} a_k(\sigma_j+\sigma_j') + 2d_k^{\boldsymbol{\lambda}}\Big) \\ & + i\Bigg[\sum_j b_j^{\boldsymbol{\lambda}}(\sigma_j-\sigma_j') + \sum_i \log\Big[1+\exp\Big(\sum_j W_{ij}^{\boldsymbol{\lambda}}\sigma_j + c_i^{\boldsymbol{\lambda}}\Big)\Big] \\ & - \sum_i \log\Big[1+\exp\Big(\sum_j W_{ij}^{\boldsymbol{\lambda}}\sigma_j' + c_i^{\boldsymbol{\lambda}}\Big)\Big] \\ & + \sum_k a_k\Big(\sum_j V_{jk}^{\boldsymbol{\lambda}} a_k(\sigma_j-\sigma_j') + 2d_k^{\boldsymbol{\lambda}}\Big)\Bigg]\Bigg\}/2\,.\end{aligned} \tag{3.23}$$

We now write the NDO in terms of the matrices $\boldsymbol{\Gamma}_{\boldsymbol{\theta}}^{[\pm]}$ defined as follows:

$$\begin{aligned}\Gamma_{\boldsymbol{\theta}}^{[\pm]}(\boldsymbol{\sigma},\boldsymbol{\sigma}') &= -\frac{1}{2}\sum_j b_j^{\boldsymbol{\theta}}(\sigma_j \pm \sigma_j') - \frac{1}{2}\sum_i \log\Big[1+\exp\Big(\sum_j W_{ij}^{\boldsymbol{\theta}}\sigma_j + \frac{1}{2}c_i^{\boldsymbol{\theta}}\Big)\Big] \\ &\quad \mp \sum_i \log\Big[1+\exp\Big(\sum_j W_{ij}^{\boldsymbol{\theta}}\sigma_j' + c_i^{\boldsymbol{\theta}}\Big)\Big] \\ &= \frac{1}{2}\big[\mathcal{E}_{\boldsymbol{\theta}}(\boldsymbol{\sigma}) \pm \mathcal{E}_{\boldsymbol{\theta}}(\boldsymbol{\sigma}')\big]\end{aligned} \tag{3.24}$$

obtaining the following:

$$\widetilde{\rho}_{\boldsymbol{\lambda}\boldsymbol{\mu}}(\boldsymbol{\sigma},\boldsymbol{\sigma}') = e^{-\Gamma_{\boldsymbol{\lambda}}^{[+]}(\boldsymbol{\sigma},\boldsymbol{\sigma}')-i\Gamma_{\boldsymbol{\mu}}^{[-]}(\boldsymbol{\sigma},\boldsymbol{\sigma}')}\sum_{\boldsymbol{a}}\exp\Bigg\{\frac{1}{2}\sum_k a_k\Big[d_k^{\boldsymbol{\lambda}} + \sum_j V_{jk}^{\boldsymbol{\lambda}}(\sigma_j+\sigma_j') + i\sum_j V_{jk}^{\boldsymbol{\mu}}(\sigma_j-\sigma_j')\Big]\Bigg\}\,. \tag{3.25}$$

Since the above expression factorizes over the auxiliary units a_k, we can carry out the sum exactly:

$$\begin{aligned}
&\sum_{\boldsymbol{a}} \exp\Bigg\{ \sum_k a_k \Big[d_k^{\boldsymbol{\lambda}} + \sum_j V_{jk}^{\boldsymbol{\lambda}} (\sigma_j - \sigma_j') \Big] \Bigg\} \\
&= \prod_k \sum_{a_k} \exp\Bigg\{ a_k \Big[d_k^{\boldsymbol{\lambda}} + \sum_j V_{jk}^{\boldsymbol{\lambda}} (\sigma_j - \sigma_j') \Big] \Bigg\} \\
&= \exp\Bigg\{ \sum_k \log \Big[1 + \exp\Big(\frac{1}{2} \sum_j V_{jk}^{\boldsymbol{\lambda}} (\sigma_j + \sigma_j') + \frac{i}{2} \sum_j V_{jk}^{\boldsymbol{\mu}} (\sigma_j - \sigma_j') + d_k^{\boldsymbol{\lambda}} \Big) \Big] \Bigg\}
\end{aligned} \tag{3.26}$$

By defining a new matrix

$$\Pi_{\boldsymbol{\lambda}\boldsymbol{\mu}}(\boldsymbol{\sigma}, \boldsymbol{\sigma}') = -\sum_k \log \Big[1 + \exp\Big(\frac{1}{2} \sum_j V_{jk}^{\boldsymbol{\lambda}} (\sigma_j + \sigma_j') + \frac{i}{2} \sum_j V_{jk}^{\boldsymbol{\mu}} (\sigma_j - \sigma_j') + d_k^{\boldsymbol{\lambda}} \Big) \Big] \tag{3.27}$$

we can give a compact definition for the NDO:

Definition 3.3. *Given a Hilbert space $\boldsymbol{\mathcal{H}}$ of a N-particle system spanned by the basis $|\boldsymbol{\sigma}\rangle = |\sigma_1, \dots, \sigma_N\rangle$, we define an auxiliary system $|\boldsymbol{a}\rangle = |a_1, \dots, a_{n_a}\rangle$ embedded in the latent space of a RBM, which purifies the physical system. The* **neural density operator** *is obtained by tracing out the auxiliary units $|\boldsymbol{a}\rangle$ from the composite pure state. The matrix elements of the NDO are given by:*

$$\rho_{\boldsymbol{\lambda}\boldsymbol{\mu}}(\boldsymbol{\sigma}, \boldsymbol{\sigma}') = Z_{\boldsymbol{\lambda}}^{-1} e^{-\Lambda_{\boldsymbol{\lambda}\boldsymbol{\mu}}(\boldsymbol{\sigma}, \boldsymbol{\sigma}')} \tag{3.28}$$

where

$$\boldsymbol{\Lambda}_{\boldsymbol{\lambda}\boldsymbol{\mu}} = \boldsymbol{\Gamma}_{\boldsymbol{\lambda}}^{[+]} + i \boldsymbol{\Gamma}_{\boldsymbol{\mu}}^{[-]} + \boldsymbol{\Pi}_{\boldsymbol{\lambda}\boldsymbol{\mu}} \,, \tag{3.29}$$

and the matrices $\boldsymbol{\Gamma}_{\boldsymbol{\lambda}/\boldsymbol{\mu}}^{[\pm]}$ and $\boldsymbol{\Pi}_{\boldsymbol{\lambda},\boldsymbol{\mu}}$ are given above. The network parameters are $(\boldsymbol{\lambda}, \boldsymbol{\mu}) = \{\boldsymbol{W}^{\boldsymbol{\lambda}}, \boldsymbol{V}^{\boldsymbol{\lambda}}, \boldsymbol{b}^{\boldsymbol{\lambda}}, \boldsymbol{c}^{\boldsymbol{\lambda}}, \boldsymbol{d}^{\boldsymbol{\lambda}}, \boldsymbol{W}^{\boldsymbol{\mu}}, \boldsymbol{V}^{\boldsymbol{\mu}}, \boldsymbol{b}^{\boldsymbol{\mu}}, \boldsymbol{c}^{\boldsymbol{\mu}}\}$. The two matrices $\boldsymbol{V}_{\boldsymbol{\lambda}}$ and $\boldsymbol{V}_{\boldsymbol{\mu}}$ encode the mixing of the physical system with the auxiliary system[3].

[3] In the case where both are set to zero, the state $\psi_{\boldsymbol{\lambda}\boldsymbol{\mu}}(\boldsymbol{\sigma}, \boldsymbol{a})$ becomes separable and the resulting NDO describes a pure state, $\hat{\rho}_{\boldsymbol{\lambda}\boldsymbol{\mu}} = |\psi_{\boldsymbol{\lambda}\boldsymbol{\mu}}(\boldsymbol{\sigma})\rangle\langle\psi_{\boldsymbol{\lambda}\boldsymbol{\mu}}(\boldsymbol{\sigma})|$.

3.3 Measurements

We turn now to the measurement process, and assume that a set of network parameters $(\boldsymbol{\lambda}, \boldsymbol{\mu})$ is given. We first examine the case of generic observables and discuss how they can be calculated using stochastic sampling (for both a neural wavefunction $|\psi_{\boldsymbol{\lambda\mu}}\rangle$ and a neural density operator $\hat{\rho}_{\boldsymbol{\lambda\mu}}$). Then, assuming a positive state $\psi_{\boldsymbol{\lambda}}(\boldsymbol{\sigma}) > 0$, we use a replicated copy of the RBM and apply the ratio trick from QMC to estimate the second Renyi entanglement entropy, with important implications for experiments.

3.3.1 Observables

We start with a neural wavefunction $|\psi_{\boldsymbol{\lambda\mu}}\rangle$ and consider a generic observable $\hat{\mathcal{O}}$, with matrix elements $\mathcal{O}_{\boldsymbol{\sigma\sigma'}} = \langle \boldsymbol{\sigma}|\hat{\mathcal{O}}|\boldsymbol{\sigma'}\rangle$ in the reference basis. We wish to compute the following expectation value

$$\langle \hat{\mathcal{O}} \rangle = \langle \psi_{\boldsymbol{\lambda\mu}}|\hat{\mathcal{O}}|\psi_{\boldsymbol{\lambda\mu}}\rangle = \sum_{\boldsymbol{\sigma\sigma'}} \psi^*_{\boldsymbol{\lambda\mu}}(\boldsymbol{\sigma'})\mathcal{O}_{\boldsymbol{\sigma'\sigma}}\psi_{\boldsymbol{\lambda\mu}}(\boldsymbol{\sigma}) \, . \tag{3.30}$$

The simplest class of observables are operators $\hat{\mathcal{O}}$ that are diagonal in the reference basis, i.e. $\mathcal{O}_{\boldsymbol{\sigma\sigma'}} = \mathcal{O}_{\boldsymbol{\sigma\sigma}}\delta_{\boldsymbol{\sigma'\sigma}}$. In this case, the expectation value simply becomes

$$\langle \hat{\mathcal{O}} \rangle = \sum_{\boldsymbol{\sigma}} |\psi_{\boldsymbol{\lambda\mu}}(\boldsymbol{\sigma})|^2 \mathcal{O}_{\boldsymbol{\sigma\sigma}} = \frac{1}{Z_{\boldsymbol{\lambda}}} \sum_{\boldsymbol{\sigma}} p_{\boldsymbol{\lambda}}(\boldsymbol{\sigma})\mathcal{O}_{\boldsymbol{\sigma\sigma}} \, , \tag{3.31}$$

which does not depends on the set of parameters $\boldsymbol{\mu}$ parametrizing the phase structure. The expectation value is computed as a MC average from samples of the visible layer generated by the RBM. Recall that this process is particularly efficient for this choice of neural network, due to the block Gibbs sampling. We can draw configurations $\{\boldsymbol{\sigma}_k\}$ from $p_{\boldsymbol{\lambda}}(\boldsymbol{\sigma})$ by repeatedly sampling the distributions $p_{\boldsymbol{\lambda}}(\boldsymbol{h} \mid \boldsymbol{\sigma})$ and $p_{\boldsymbol{\lambda}}(\boldsymbol{\sigma} \mid \boldsymbol{h})$ and compute the average value of the observable as

$$\langle \hat{\mathcal{O}} \rangle \simeq \frac{1}{n_{\mathrm{MC}}} \sum_{k=1}^{n_{\mathrm{MC}}} \mathcal{O}_{\boldsymbol{\sigma}_k} \tag{3.32}$$

with error scaling as $\delta\hat{\mathcal{O}} \sim 1/\sqrt{n_{\mathrm{MC}}}$. Relevant examples of diagonal measurements are the magnetization $m^z = \langle\hat{\sigma}^z\rangle$, the spin-spin correlation function $C^z_{ij} = \langle\hat{\sigma}^z_i\hat{\sigma}^z_j\rangle$ for quantum spin systems, the occupation number $n_j = \langle\hat{c}^\dagger_j\hat{c}_j\rangle$ for bosons/fermions on lattice, and the up/down polarization of photons. All of these observables can be evaluated directly from the configurations of the visible layer of the $\boldsymbol{\lambda}$-RBM.

Next, we examine the case of off-diagonal observables $\mathcal{O}_{\boldsymbol{\sigma}\boldsymbol{\sigma}'} \neq 0$. As this observables depends on the coherences of the system, their computation involves now the phases of the state, and thus the $\boldsymbol{\mu}$-RBM. We write again the expectation value of the observable:

$$\begin{aligned}\langle\hat{\mathcal{O}}\rangle &= \sum_{\boldsymbol{\sigma}\boldsymbol{\sigma}'} \psi_{\boldsymbol{\lambda}\boldsymbol{\mu}}(\boldsymbol{\sigma})\psi^*_{\boldsymbol{\lambda}\boldsymbol{\mu}}(\boldsymbol{\sigma}')\mathcal{O}_{\boldsymbol{\sigma}'\boldsymbol{\sigma}} \\ &= \sum_{\boldsymbol{\sigma}} |\psi_{\boldsymbol{\lambda}\boldsymbol{\mu}}(\boldsymbol{\sigma})|^2 \sum_{\boldsymbol{\sigma}'} \frac{\psi^*_{\boldsymbol{\lambda}\boldsymbol{\mu}}(\boldsymbol{\sigma}')}{\psi^*_{\boldsymbol{\lambda}\boldsymbol{\mu}}(\boldsymbol{\sigma})}\mathcal{O}_{\boldsymbol{\sigma}'\boldsymbol{\sigma}} = \frac{1}{Z_{\boldsymbol{\lambda}}}\sum_{\boldsymbol{\sigma}} p_{\boldsymbol{\lambda}}(\boldsymbol{\sigma})\mathcal{O}^{[L]}_{\boldsymbol{\sigma}} \\ &\simeq \frac{1}{n_{\mathrm{MC}}}\sum_{k=1}^{n_{\mathrm{MC}}} \mathcal{O}^{[L]}_{\boldsymbol{\sigma}_k}\,.\end{aligned} \tag{3.33}$$

The expectation value $\langle\hat{\mathcal{O}}\rangle$ reduces to the MC average of the so-called "local estimate" of the observables:

$$\mathcal{O}^{[L]}_{\boldsymbol{\sigma}} = \sum_{\boldsymbol{\sigma}'} \frac{\psi^*_{\boldsymbol{\lambda}\boldsymbol{\mu}}(\boldsymbol{\sigma}')}{\psi^*_{\boldsymbol{\lambda}\boldsymbol{\mu}}(\boldsymbol{\sigma})}\mathcal{O}_{\boldsymbol{\sigma}'\boldsymbol{\sigma}} = \sum_{\boldsymbol{\sigma}'} e^{(\mathcal{E}_{\boldsymbol{\lambda}}(\boldsymbol{\sigma})-\mathcal{E}_{\boldsymbol{\lambda}}(\boldsymbol{\sigma}'))/2}\, e^{-i(\mathcal{E}_{\boldsymbol{\mu}}(\boldsymbol{\sigma})-\mathcal{E}_{\boldsymbol{\mu}}(\boldsymbol{\sigma}'))/2}\, \mathcal{O}_{\boldsymbol{\sigma}'\boldsymbol{\sigma}}, \tag{3.34}$$

which does not involve the unknown partition function $Z_{\boldsymbol{\lambda}}$. Note that to evaluate this sum, we require that the matrix representation of the observable in the reference basis is sparse[4]. In practice, the average is carried out again on the samples from the distribution $p_{\boldsymbol{\lambda}}(\boldsymbol{\sigma})$. As an example, consider the expectation value of the transverse magnetization $\hat{\sigma}^x_j$ on site j, computed with neural wavefunction in the reference basis $\{\boldsymbol{\sigma} = \boldsymbol{\sigma}^z\}$. Then matrix element of the operator is

$$[\hat{\sigma}^x_j]_{\boldsymbol{\sigma}\boldsymbol{\sigma}'} = \langle\boldsymbol{\sigma}|\sigma^x_j|\boldsymbol{\sigma}'\rangle = \delta_{\sigma_j,1-\sigma'_j}\prod_{i\neq j}\delta_{\sigma_i,\sigma'_i}\,, \tag{3.35}$$

[4]The number of non-zero elements of the matrix should scale sub-exponentially with the size N

and the local estimate of the observable is

$$[\hat{\sigma}_j^x]_{\boldsymbol{\sigma}}^{[L]} = \frac{\psi_{\boldsymbol{\lambda}}(\sigma_1, \dots, 1-\sigma_j, \sigma_{j+1}, \dots)}{\psi_{\boldsymbol{\lambda}}(\sigma_1, \dots, \sigma_j, \sigma_{j+1}, \dots)} \tag{3.36}$$

Using the bloc Gibbs sampling, the expectation value is approximated as a MC average:

$$\langle \hat{\sigma}_j^x \rangle \simeq \frac{1}{n_{\mathrm{MC}}} \sum_{k=1}^{n_{\mathrm{MC}}} [\hat{\sigma}_j^x]_{\boldsymbol{\sigma}_k}^{[L]} \,. \tag{3.37}$$

A similar procedure is used to estimate observables using a NDO $\hat{\rho}_{\boldsymbol{\lambda\mu}}$. Now the expectation value of an observable $\hat{\mathcal{O}}$ is given by:

$$\langle \hat{\mathcal{O}} \rangle = \mathrm{Tr}_{\boldsymbol{\sigma}} \{ \hat{\rho}_{\boldsymbol{\lambda\mu}} \hat{\mathcal{O}} \} \tag{3.38}$$

which is calculated considering the full pure neural wavefunction on the composite system:

$$\begin{aligned} \langle \hat{\mathcal{O}} \rangle &= \langle \psi_{\boldsymbol{\lambda\mu}} | \hat{\mathcal{O}} \otimes \hat{\mathcal{I}}_{\boldsymbol{a}} | \psi_{\boldsymbol{\lambda\mu}} \rangle = \sum_{\boldsymbol{\sigma\sigma'}} \sum_{\boldsymbol{a}} \psi_{\boldsymbol{\lambda\mu}}(\boldsymbol{\sigma}, \boldsymbol{a}) \psi^*_{\boldsymbol{\lambda\mu}}(\boldsymbol{\sigma'}, \boldsymbol{a}) \mathcal{O}_{\boldsymbol{\sigma'\sigma}} \\ &= \sum_{\boldsymbol{\sigma a}} |\psi_{\boldsymbol{\lambda\mu}}(\boldsymbol{\sigma}, \boldsymbol{a})|^2 \sum_{\boldsymbol{\sigma'}} \frac{\psi^*_{\boldsymbol{\lambda\mu}}(\boldsymbol{\sigma'}, \boldsymbol{a})}{\psi^*_{\boldsymbol{\lambda\mu}}(\boldsymbol{\sigma}, \boldsymbol{a})} \mathcal{O}_{\boldsymbol{\sigma'\sigma}} \end{aligned} \tag{3.39}$$

where $\hat{\mathcal{I}}_{\boldsymbol{a}}$ is the identity operator acting on the Hilbert space of the auxiliary degrees of freedom. Therefore, we can approximate the expectation value of $\hat{\mathcal{O}}$ with a MC average of

$$\mathcal{O}^{[L]}_{(\boldsymbol{\sigma},\boldsymbol{a})} = \sum_{\boldsymbol{\sigma'}} e^{[\Upsilon_{\boldsymbol{\lambda}}(\boldsymbol{\sigma},\boldsymbol{a}) - \Upsilon_{\boldsymbol{\lambda}}(\boldsymbol{\sigma'},\boldsymbol{a})]/2} e^{-i\Upsilon_{\boldsymbol{\mu}}(\boldsymbol{\sigma},\boldsymbol{a}) - \Upsilon_{\boldsymbol{\mu}}(\boldsymbol{\sigma'},\boldsymbol{a})]/2} \mathcal{O}_{\boldsymbol{\sigma'\sigma}} \tag{3.40}$$

over a collection of samples drawn from the distribution $|\psi_{\boldsymbol{\lambda\mu}}(\boldsymbol{\sigma}, \boldsymbol{a})|^2 = Z_{\boldsymbol{\lambda}}^{-1} p_{\boldsymbol{\lambda}}(\boldsymbol{\sigma}, \boldsymbol{a})$. For the case of NDO, sampling the distribution $p_{\boldsymbol{\lambda}}(\boldsymbol{\sigma}, \boldsymbol{a})$ is equivalent to sampling the conditional distributions $p_{\boldsymbol{\lambda}}(\boldsymbol{\sigma} \mid \boldsymbol{h}, \boldsymbol{a})$, $p_{\boldsymbol{\lambda}}(\boldsymbol{h} \mid \boldsymbol{\sigma})$ and $p_{\boldsymbol{\lambda}}(\boldsymbol{a} \mid \boldsymbol{\sigma})$, which, also does not require the knowledge of the partition function.

3.3.2 Entanglement entropy

We finally turn to the measurement of entanglement. For the case of pure states, the entanglement can be quantified by the Renyi entanglement entropy S_α. In particular we consider the second Renyi entropy, which was successfully calculated in QMC simulations. Given a partition of the physical system into a region A and its complement $A^\perp$, the entropy of region A with reduced density matrix $\hat{\rho}_A = \mathrm{Tr}_{A^\perp}(\hat{\rho}) = \mathrm{Tr}_{A^\perp}(|\Psi\rangle\langle\Psi|)$ is given by

$$S_2(\hat{\rho}_A) = -\log(\mathrm{Tr}_A(\hat{\rho}_A^2))\,. \tag{3.41}$$

The algorithm, proposed by Hastings and Melko in 2010 [132] in the framework of stochastic series expansion, uses two replicated copies of the system to estimate the second Renyi entropy $S_2(\hat{\rho}_A)$.

We will now show how this algorithms can be easily translated in the framework of neural wavefunction. The algorithm, since it was framed on QMC, assumes a real and positive state, which in our case is parametrized as

$$|\psi_{\boldsymbol{\lambda}}\rangle = \sum_{\boldsymbol{\sigma},\boldsymbol{\sigma}^\perp} \psi_{\boldsymbol{\lambda}}(\boldsymbol{\sigma},\boldsymbol{\sigma}^\perp)|\boldsymbol{\sigma}\rangle \otimes |\boldsymbol{\sigma}^\perp\rangle, \tag{3.42}$$

where $\{|\boldsymbol{\sigma}\rangle\}$ and $\{|\boldsymbol{\sigma}^\perp\rangle\}$ are basis state for the regions A and $A^\perp$ respectively and $\psi_{\boldsymbol{\lambda}}(\boldsymbol{\sigma},\boldsymbol{\sigma}^\perp) = (\langle\boldsymbol{\sigma}| \otimes \langle\boldsymbol{\sigma}^\perp|)|\psi_{\boldsymbol{\lambda}}\rangle$. The reduced density matrix for sub-region A is

$$\begin{aligned}\hat{\rho}_A &= \mathrm{Tr}_{A^\perp}\left[\sum_{\boldsymbol{\sigma},\boldsymbol{\sigma}^\perp}\sum_{\boldsymbol{\sigma}',\boldsymbol{\sigma}'^\perp} \psi(\boldsymbol{\sigma},\boldsymbol{\sigma}^\perp)\psi(\boldsymbol{\sigma}',\boldsymbol{\sigma}'^\perp)|\boldsymbol{\sigma}\rangle\langle\boldsymbol{\sigma}'| \otimes |\boldsymbol{\sigma}^\perp\rangle\langle\boldsymbol{\sigma}'^\perp|\right] \\ &= \sum_{\boldsymbol{\sigma},\boldsymbol{\sigma}'}\left[\sum_{\boldsymbol{\sigma}^\perp}\psi(\boldsymbol{\sigma},\boldsymbol{\sigma}^\perp)\psi(\boldsymbol{\sigma}',\boldsymbol{\sigma}^\perp)\right]|\boldsymbol{\sigma}\rangle\langle\boldsymbol{\sigma}'|\end{aligned} \tag{3.43}$$

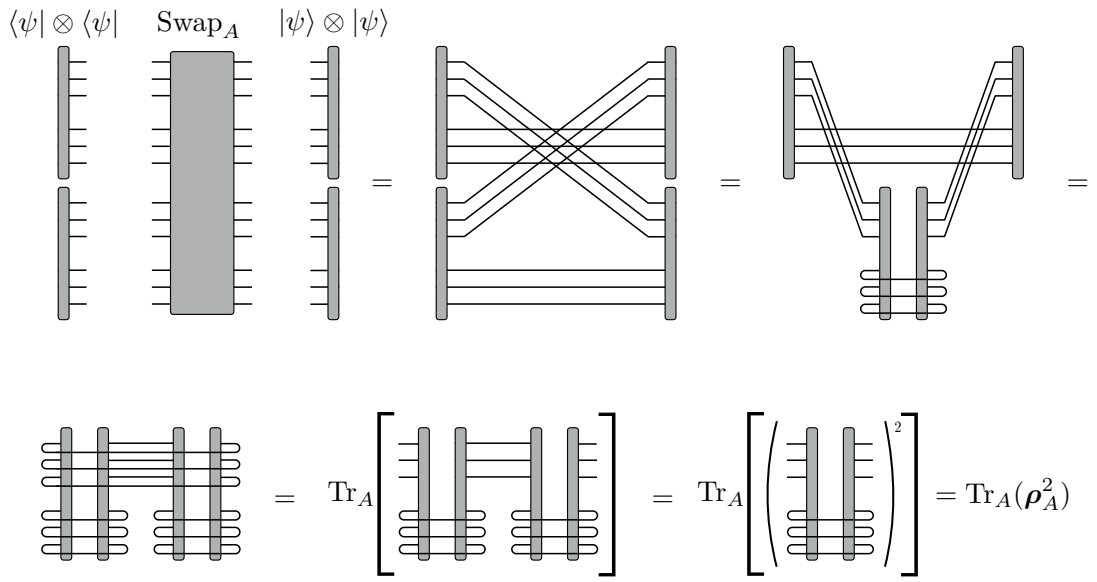

Figure 3.3: **Replica trick for entanglement entropy**. The expectation value of the swap operator computed between two replicated copies of the system. The swap operator exchanges the tensor indices for the subregion A, and corresponds to computing the trace of the reduced density matrix squared for sub-region A.

The trace of $\hat{\rho}_A^2$ is then:

$$\begin{aligned}\mathrm{Tr}_A(\hat{\rho}_A^2) &= \mathrm{Tr}_A\left[\sum_{\boldsymbol{\sigma}_1,\boldsymbol{\sigma}_2}\sum_{\boldsymbol{\sigma}_1^\perp,\boldsymbol{\sigma}_2^\perp}\sum_{\boldsymbol{\sigma}_2'}\psi(\boldsymbol{\sigma}_1,\boldsymbol{\sigma}_1^\perp)\psi(\boldsymbol{\sigma}_2,\boldsymbol{\sigma}_1^\perp)\psi(\boldsymbol{\sigma}_2,\boldsymbol{\sigma}_2^\perp)\psi(\boldsymbol{\sigma}_2',\boldsymbol{\sigma}_2^\perp)|\boldsymbol{\sigma}_1\rangle\langle\boldsymbol{\sigma}_2'|\right] \\ &= \sum_{\boldsymbol{\sigma}_1,\boldsymbol{\sigma}_2}\sum_{\boldsymbol{\sigma}_1^\perp,\boldsymbol{\sigma}_2^\perp}\psi(\boldsymbol{\sigma}_1,\boldsymbol{\sigma}_1^\perp)\psi(\boldsymbol{\sigma}_2,\boldsymbol{\sigma}_1^\perp)\psi(\boldsymbol{\sigma}_1,\boldsymbol{\sigma}_2^\perp)\psi(\boldsymbol{\sigma}_2,\boldsymbol{\sigma}_2^\perp)\,.\end{aligned} \tag{3.44}$$

We can re-weight this expression to obtain

$$\begin{aligned}\mathrm{Tr}_A(\hat{\rho}_A^2) &= \sum_{\boldsymbol{\sigma}_1,\boldsymbol{\sigma}_2}\sum_{\boldsymbol{\sigma}_1^\perp,\boldsymbol{\sigma}_2^\perp}|\psi(\boldsymbol{\sigma}_1,\boldsymbol{\sigma}_1^\perp)\psi(\boldsymbol{\sigma}_2,\boldsymbol{\sigma}_2^\perp)|^2\frac{\psi(\boldsymbol{\sigma}_2,\boldsymbol{\sigma}_1^\perp)\psi(\boldsymbol{\sigma}_1,\boldsymbol{\sigma}_2^\perp)}{\psi(\boldsymbol{\sigma}_1,\boldsymbol{\sigma}_1^\perp)\psi(\boldsymbol{\sigma}_2,\boldsymbol{\sigma}_2^\perp)} \\ &= \left\langle\frac{\psi(\boldsymbol{\sigma}_2,\boldsymbol{\sigma}_1^\perp)\psi(\boldsymbol{\sigma}_1,\boldsymbol{\sigma}_2^\perp)}{\psi(\boldsymbol{\sigma}_1,\boldsymbol{\sigma}_1^\perp)\psi(\boldsymbol{\sigma}_2,\boldsymbol{\sigma}_2^\perp)}\right\rangle_{|\psi_1\psi_2|^2},\end{aligned} \tag{3.45}$$

and evaluate it using MC sampling of the distribution $|\psi(\boldsymbol{\sigma}_1, \boldsymbol{\sigma}_1^\perp)\psi(\boldsymbol{\sigma}_2, \boldsymbol{\sigma}_2^\perp)|^2$. Note that the expectation value in the above equation corresponds to measure the expectation value of an operator that swaps the configurations of the two replicas for region A:

$$\begin{aligned} S_2(\hat{\rho}_A) &= -\log(\mathrm{Tr}_A(\hat{\rho}_A^2)) = -\log\left\langle \frac{\psi(\boldsymbol{\sigma}_2, \boldsymbol{\sigma}_1^\perp)\psi(\boldsymbol{\sigma}_1, \boldsymbol{\sigma}_2^\perp)}{\psi(\boldsymbol{\sigma}_1, \boldsymbol{\sigma}_1^\perp)\psi(\boldsymbol{\sigma}_2, \boldsymbol{\sigma}_2^\perp)} \right\rangle_{|\psi_1\psi_2|^2} \\ &= -\log\left[\langle\psi_2| \otimes \langle\psi_1|\mathrm{Swap}_A\,|\psi_1\rangle \otimes |\psi_2\rangle\right] \end{aligned} \tag{3.46}$$

A more intuitive explanation is obtained by adopting a TN notation (Fig. 3.3) [133]. Starting from the two copies of the system we take the expectation value of the swap operator (a). This acts on the two copies by swapping the configuration of region A (b). The TN can be re-arranged (c-f), resulting in the trace of the reduced density matrix squared, which is directly related to the second Renyi entropy.

Improved ratio trick

The replica trick shown above provides a way to compute the entanglement entropy, but in practice the expectation value of the Swap operator becomes very small when the subregion size grows large, leading to very high sampling noise. To avoid this issue, we instead implement the *improved ratio trick*, also proposed in [132]. Assuming we are dealing with a one-dimensional chain of N sites, the entanglement entropy for a subregion A containing a block of M sites can be computed as

$$S_2(\rho_A) = -\sum_{j=0}^{M-1} \log \frac{\langle \mathrm{Swap}_{A^{j+1}} \rangle}{\langle \mathrm{Swap}_{A^j} \rangle}, \tag{3.47}$$

where A^j contains j sites and $\langle \mathrm{Swap}_{A^0} \rangle = 1$. The state of the two copies is

$$|\psi_{\boldsymbol{\lambda}}\rangle \otimes |\psi_{\boldsymbol{\lambda}}\rangle = \sum_{\boldsymbol{\sigma}_1}\sum_{\boldsymbol{\sigma}_2} \psi_{\boldsymbol{\lambda}}(\boldsymbol{\sigma}_1)\psi_{\boldsymbol{\lambda}}(\boldsymbol{\sigma}_2)|\boldsymbol{\sigma}_1\rangle \otimes |\boldsymbol{\sigma}_2\rangle, \tag{3.48}$$

and the expectation value of the Swap operator is

$$\langle \text{Swap}_A^j \rangle = \sum_{\boldsymbol{\sigma}_1} \sum_{\boldsymbol{\sigma}_2} \psi_{\boldsymbol{\lambda}}(\boldsymbol{\sigma}_1)\psi_{\boldsymbol{\lambda}}(\boldsymbol{\sigma}_2)\psi_{\boldsymbol{\lambda}}(\boldsymbol{\sigma}_{12}^j)\psi_{\boldsymbol{\lambda}}(\boldsymbol{\sigma}_{21}^j), \tag{3.49}$$

where we defined $\boldsymbol{\sigma}_{12}^j = (\sigma_1^1, \sigma_1^2, \ldots, \sigma_1^{j-1}, \sigma_2^j, \ldots, \sigma_2^N)$ and $\boldsymbol{\sigma}_{21}^j = (\sigma_2^1, \sigma_2^2, \ldots, \sigma_2^{j-1}, \sigma_1^j, \ldots, \sigma_1^N)$. The ratio of expectation values then can be rewritten as

$$\begin{aligned}
\frac{\langle \text{Swap}_A^{j+1} \rangle}{\langle \text{Swap}_A^j \rangle} &= \frac{\sum_{\boldsymbol{\sigma}_1} \sum_{\boldsymbol{\sigma}_2} \psi_{\boldsymbol{\lambda}}(\boldsymbol{\sigma}_1)\psi_{\boldsymbol{\lambda}}(\boldsymbol{\sigma}_2)\psi_{\boldsymbol{\lambda}}(\boldsymbol{\sigma}_{12}^{j+1})\psi_{\boldsymbol{\lambda}}(\boldsymbol{\sigma}_{21}^{j+1})}{\sum_{\boldsymbol{\sigma}_1} \sum_{\boldsymbol{\sigma}_2} \psi_{\boldsymbol{\lambda}}(\boldsymbol{\sigma}_1)\psi_{\boldsymbol{\lambda}}(\boldsymbol{\sigma}_2)\psi_{\boldsymbol{\lambda}}(\boldsymbol{\sigma}_{12}^j)\psi_{\boldsymbol{\lambda}}(\boldsymbol{\sigma}_{21}^j)} \\
&= \frac{\sum_{\boldsymbol{\sigma}_1} \sum_{\boldsymbol{\sigma}_2} \psi_{\boldsymbol{\lambda}}(\boldsymbol{\sigma}_1)\psi_{\boldsymbol{\lambda}}(\boldsymbol{\sigma}_2)\psi_{\boldsymbol{\lambda}}(\boldsymbol{\sigma}_{12}^j)\psi_{\boldsymbol{\lambda}}(\boldsymbol{\sigma}_{21}^j)\frac{\psi_{\boldsymbol{\lambda}}(\boldsymbol{\sigma}_{12}^{j+1})\psi_{\boldsymbol{\lambda}}(\boldsymbol{\sigma}_{21}^{j+1})}{\psi_{\boldsymbol{\lambda}}(\boldsymbol{\sigma}_{12}^j)\psi_{\boldsymbol{\lambda}}(\boldsymbol{\sigma}_{21}^j)}}{\sum_{\boldsymbol{\sigma}_1} \sum_{\boldsymbol{\sigma}_2} \psi_{\boldsymbol{\lambda}}(\boldsymbol{\sigma}_1)\psi_{\boldsymbol{\lambda}}(\boldsymbol{\sigma}_2)\psi_{\boldsymbol{\lambda}}(\boldsymbol{\sigma}_{12}^j)\psi_{\boldsymbol{\lambda}}(\boldsymbol{\sigma}_{21}^j)} \\
&= \frac{\sum_{\boldsymbol{\sigma}_1} \sum_{\boldsymbol{\sigma}_2} Q^j(\boldsymbol{\sigma}_1, \boldsymbol{\sigma}_2)\mathcal{R}^j(\boldsymbol{\sigma}_1, \boldsymbol{\sigma}_2)}{\sum_{\boldsymbol{\sigma}_1} \sum_{\boldsymbol{\sigma}_2} Q^j(\boldsymbol{\sigma}_1, \boldsymbol{\sigma}_2)} \\
&= \langle \mathcal{R}^j(\boldsymbol{\sigma}_1, \boldsymbol{\sigma}_2) \rangle_{Q^j},
\end{aligned} \tag{3.50}$$

where we defined the probability distribution $Q^j(\boldsymbol{\sigma}_1, \boldsymbol{\sigma}_2) = \psi_{\boldsymbol{\lambda}}(\boldsymbol{\sigma}_1)\psi_{\boldsymbol{\lambda}}(\boldsymbol{\sigma}_2)\psi_{\boldsymbol{\lambda}}(\boldsymbol{\sigma}_{12}^j)\psi_{\boldsymbol{\lambda}}(\boldsymbol{\sigma}_{21}^j)$ and the observable

$$\mathcal{R}^j(\boldsymbol{\sigma}_1, \boldsymbol{\sigma}_2) = \frac{\psi_{\boldsymbol{\lambda}}(\boldsymbol{\sigma}_{12}^{j+1})\psi_{\boldsymbol{\lambda}}(\boldsymbol{\sigma}_{21}^{j+1})}{\psi_{\boldsymbol{\lambda}}(\boldsymbol{\sigma}_{12}^j)\psi_{\boldsymbol{\lambda}}(\boldsymbol{\sigma}_{21}^j)}. \tag{3.51}$$

To compute the expectation value $\langle \mathcal{R}^j(\boldsymbol{\sigma}_1, \boldsymbol{\sigma}_2) \rangle_{P^j}$ we employ standard MC simulation, where spin configurations $(\boldsymbol{\sigma}_1, \boldsymbol{\sigma}_2)$ for the two copies are sampled from the probability distribution $Q^j(\boldsymbol{\sigma}_1, \boldsymbol{\sigma}_2)$. If the system is a one-dimensional chain with sub-region A containing M out of the N physical degrees of freedom, the entanglement entropy for sub-region A can be computed with the improved ratio trick as

$$S_2(\rho_M) = -\sum_{j=0}^{M-1} \log \langle \mathcal{R}^j(\boldsymbol{\sigma_1}, \boldsymbol{\sigma_2}) \rangle_{Q^j}. \tag{3.52}$$

where the M different simulations can be ran in parallel.

3.4 Conclusion

We have introduced a new representation for quantum many-body states based on a RBM. The representation is built upon the link between the probabilistic interpretation of quantum states in terms of the measurement process, and the probabilistic nature of generative models. For closed and isolated physical systems, described by a quantum wavefunction, the quantum state is compressed into the set of network parameters $\boldsymbol{\lambda}$ of the RBM, provided the quantum state is positive, $\Psi(\boldsymbol{\sigma}) \geq 0$. If however the state features a non-trivial sign (phase) structure, an additional RBM with parameters $\boldsymbol{\mu}$ is required to parametrize the state, while retaining all the probabilistic properties of the neural network, such as block Gibbs sampling. In turn, for open quantum systems described by mixed states, we have shown that a neural-network representation of a density operator can be constructed using a purification scheme, where additional degrees of freedom are added to the physical system such that the quantum state of the composite system is pure. The physical density operator is then obtained by tracing out the auxiliary units. While this process would usually be computationally cumbersome, the RBM provides a natural way of carrying out the task. By embedding the ancillary units in the latent space of the neural network, we can efficiently obtain a possibly compact representation of the physical density operator. We point out that, even though we have only considered $2d$ local Hilbert spaces, the generalization to higher-dimensional spaces (e.g. higher spins, bosonic states, etc) can be easily obtained by using a multinomial RBM [134].

Given a set of network parameters for either a neural wavefunction or a NDO, expectation values of physical observables can be estimated approximately as a MC average on the sample obtained from the RBM. The process results to be efficient, as long as the matrix representation of the observable in the reference basis is sufficiently sparse. More interestingly, the replica trick used to compute entanglement entropy in QMC has a straightforward implementation with neural wavefunction, where the second Renyi entropy is obtained by sampling the swap operator between two replicated RBMs. As we will see in Chapter 4, this property has important consequences for both computational and experimental studies of strongly-correlated materials.

Chapter 4

Quantum data reconstruction

Quantum states of many-body systems, whether pure of mixed, can be represented by a RBM, or a combination thereof. The representation relies on a classical probability distributions $p_{\boldsymbol{\theta}}(\boldsymbol{\sigma})$, parametrized by a set of network weights $\boldsymbol{\theta}$. Provided enough number of parameters, any quantum state can be expressed in this form, but yet this number might increase exponentially with the size N of the physical system. Now that the neural-network representation, as well as the measurement procedures, have been defined, we turn to the problem of discovering the parameters $\boldsymbol{\theta}$. For the case of pure quantum states, for example, given a target state $|\Psi\rangle$, we wish to discover a set of parameters $\boldsymbol{\theta}^*$, such that the neural wavefunction is a good approximation of the target state, $|\psi_{\boldsymbol{\theta}^*}\rangle \simeq |\Psi\rangle$.

There are two main paradigm to learn a quantum state. In a *knowledge-driven* approach, we have some a priori information about the underlying microscopic physical laws governing the system. For ground state problem of an Hamiltonian $\hat{H}$, in some special cases it is possible to find an exact neural-network representations [135, 136]. In general, however, given a neural-network ansatz $|\psi_{\boldsymbol{\theta}}\rangle$, this problem should be tackled numerically, for example through VMC. According to the variational principle, the energy functional

$$E_{\boldsymbol{\theta}} = \langle\psi_{\boldsymbol{\theta}}|\hat{H}|\psi_{\boldsymbol{\theta}}\rangle \tag{4.1}$$

is greater or equal to the ground state energy E_0, and $E_{\boldsymbol{\theta}} = E_0$ if and only if the neural-

network state is the ground state of the Hamiltonian, $|\psi_{\boldsymbol{\theta}}\rangle = |\Psi_0\rangle$. One can then optimize the variational wavefunction $\psi_{\boldsymbol{\theta}}$ with respect to the parameters $\boldsymbol{\theta}$ to minimize $E_{\boldsymbol{\theta}}$ until convergence. Many different variational ansatz have been proposed for VMC, depending on the a priori insights we possess on the quantum state we are trying to learn. ML has entered the picture of VMC optimization only recently [74], using a modified RBM with complex weights. This representation is being investigated from the perspective of representational power [80], its relation with the entanglement of quantum states [137] and its performance in the numerical simulations [138, 139].

The second path to discover the network parameters, which we develop in this Thesis, assumes minimal knowledge about the quantum state under consideration, and rather follows a *data-driven* approach to construct the neural-network state. More precisely, we wish to reconstruct the quantum state from a set of data consisting of appropriate measurements performed on the physical systems. This process of quantum state reconstruction (QSR), also known as quantum state tomography [140, 141, 142], is becoming increasingly important to validate and characterize the new generation of quantum hardware, including both quantum circuit and synthetic quantum matter.

The Chapter is organized as follows. In Sec. 4.1 we present a set of algorithms to reconstruct positive and complex wavefunctions, as well as density matrices for mixed states. In Sec. 4.2 we demonstrate the power of neural-network QSR for a variety of quantum states, with datasets containing either synthetic measurements generated with classical computers, or real measurement taken in laboratories. In Sec. 4.3 we conclude by discussing the limitations and reconstruction errors of the proposed technique. The Chapter covers material from the following references: [143, 144, 145].

4.1 Neural-network quantum reconstruction

Let us consider an unknown target quantum state we wish to reconstruct, such as a density operator $\hat{\varrho}$. Within an experimental scenario, the first assumptions is the ability to reliably prepare many identical copies of the state $\hat{\varrho}$. Furthermore, one requires a set of observables $\{\hat{\mathcal{O}}_{\boldsymbol{\gamma}}\}$ whose measurements are feasible in the experimental setup. Then, traditional quantum state tomography aims to learn the full density matrix $\varrho(\boldsymbol{\sigma}, \boldsymbol{\sigma}')$ (in some given reference basis $\{\boldsymbol{\sigma}\}$), from the experimental measurements outcomes of the observables, $\mathcal{O}_{\boldsymbol{\gamma}}^{\text{exp}} = \langle \hat{\mathcal{O}}_{\boldsymbol{\gamma}} \rangle_{\text{exp}}$. Among traditional techniques we find the *linear inversion method*, which approximates the probability of obtaining outcome $\boldsymbol{\gamma}$ from measuring the observable $\hat{\mathcal{O}}_{\boldsymbol{\gamma}}$ with an experimental histogram [146]

$$P(\boldsymbol{\gamma} \mid \hat{\varrho}) = \text{Tr}\big(\hat{\varrho}\, \hat{\mathcal{O}}_{\boldsymbol{\gamma}}\big) \simeq P_{\text{hist}}(\boldsymbol{\gamma})\,, \tag{4.2}$$

and finds a reconstructed density matrix $\hat{\rho}_{\text{LI}}$ by inverting Eq. 4.2:

$$\hat{\rho}_{\text{LI}} = \sum_{\boldsymbol{\gamma}\boldsymbol{\gamma}'} T^{-1}_{\boldsymbol{\gamma}\boldsymbol{\gamma}'} P_{\text{hist}}(\boldsymbol{\gamma}') \hat{\mathcal{O}}_{\boldsymbol{\gamma}}\,, \tag{4.3}$$

where $T_{\boldsymbol{\gamma}\boldsymbol{\gamma}'} = \text{Tr}(\hat{\mathcal{O}}_{\boldsymbol{\gamma}} \hat{\mathcal{O}}_{\boldsymbol{\gamma}'})$. The simplicity of this approach comes with important shortcomings. First, the reconstructed density matrix can become non positive semi-definite, and thus unphysical. Second, the density matrix $\hat{\rho}_{\text{LI}}$ is represented explicitly, leading to an exponential overhead in the quantum state representation. Third, the reconstruction process requires the matrix inversion, which also scales exponentially with system size.

A different approach, also related to our proposed technique, is the *maximum-likelihood estimation* (MLE). Given a parametrization $\hat{\rho}_{\text{MLE}}$ of the density matrix, the reconstruction is carried out by maximizing the probability that the density matrix $\hat{\rho}_{\text{MLE}}$ would generate the measurement data:

$$\hat{\rho}_{\text{MLE}} = \text{argmax}\left[P(\boldsymbol{\gamma} \mid \hat{\varrho})\right]\,. \tag{4.4}$$

In this case, the reconstruction can be in principle carried out without exponential overhead and the density matrix can be built to be positive semi-definite by construction. A typical

choice of representation is given by

$$\hat{\rho}_{\mathrm{MLE}} = \frac{\hat{T}^{\dagger}\hat{T}}{\mathrm{Tr}(\hat{T}^{\dagger}\hat{T})} \tag{4.5}$$

where $\hat{T}$ is a tri-diagonal hermitian matrix. The drawback of this approach is however still in the representation, which is clearly not scalable.

Traditional brute-force approaches to QSR pose a high demand on computational resources, and thus allow the reconstruction of physical systems with a only small number of degrees of freedom [147, 148]. While there are techniques to reduce the experimental burden, such as compressed sensing [149], large systems can be studied only through techniques requiring a feasible number of measurements. For example, permutationally-invariant tomography [150], makes efficient use of the symmetries of prototypical quantum optics states, and can be amenable to a large number of qubits. However, the QSR of a general many-body systems remains very challenging. In this context, the efficient representation of quantum states with MPS, makes TNs the state-of-the-art tool for the reconstruction of low-entangled states [151, 152]. For highly-entangled quantum states resulting either from deep quantum circuits or high-dimensional physical systems, alternative representations are required.

In the task of reconstructing an unknown quantum state from measurements, there are two fundamental complexity issues. First, we need an efficient parametrization of the state. That is, irrespective to the procedure to learn the state coefficients, we need to be able to store the quantum state in a compact way. In this regard, we have shown that neural network have the potential to provide such a representation. Second, we need algorithm that extracts as much information as possible from a limited set of measurements data. Thus, QSR is a data-driven problem, so it is very natural to implement ML algorithms to design a QSR framework for quantum many-body states. In the following Subsections, we will introduce and discuss in detail the reconstruction algorithms for pure positive/complex quantum wavefunction and mixed density operators.

4.1.1 Positive wavefunctions

We begin with the simplest scenario where the target quantum state we wish to reconstruct is pure and described by a quantum wavefunction $|\Psi\rangle$ with positive coefficients:

$$\Psi(\boldsymbol{\sigma}) \geq 0 \qquad \forall |\boldsymbol{\sigma}\rangle \in \boldsymbol{\mathcal{H}} \,. \tag{4.6}$$

Under the assumption of purity, the projective measurements in the reference basis $\{|\boldsymbol{\sigma}\rangle\}$ provides sufficient information to carry out QSR reliably. The outcomes of the measurements are distributed according the probability distribution $P(\boldsymbol{\sigma}) = |\Psi(\boldsymbol{\sigma})|^2$, obtained by projecting the target wavefunction into the reference basis. We adopt the neural wavefunction representation without any phase structure:

$$\psi_{\boldsymbol{\lambda}}(\boldsymbol{\sigma}) \equiv \sqrt{p_{\boldsymbol{\lambda}}(\boldsymbol{\sigma})} = Z_{\boldsymbol{\lambda}}^{-\frac{1}{2}} \, e^{-\mathcal{E}_{\boldsymbol{\lambda}}(\boldsymbol{\sigma})/2} \,. \tag{4.7}$$

The strategy is to carry out the reconstruction using unsupervised learning on the distribution $P(\boldsymbol{\sigma})$. This corresponds to minimize the KL divergence

$$\mathcal{C}_{\boldsymbol{\lambda}}^{P} = \sum_{\boldsymbol{\sigma}} P(\boldsymbol{\sigma}) \log \frac{P(\boldsymbol{\sigma})}{|\psi_{\boldsymbol{\lambda}}(\boldsymbol{\sigma})|^2} \,. \tag{4.8}$$

Upon nearly perfect training $C_{\boldsymbol{\lambda}} \simeq 0$, the following condition holds:

$$P(\boldsymbol{\sigma}) = |\Psi(\boldsymbol{\sigma})|^2 \simeq |\psi_{\boldsymbol{\lambda}}(\boldsymbol{\sigma})|^2 \,, \tag{4.9}$$

and, since both states are positive, we have discovered the quantum state $\psi_{\boldsymbol{\lambda}}(\boldsymbol{\sigma}) \simeq \Psi(\boldsymbol{\sigma})$, as required. As the quantum state, and so the probability P is unknown, we assume we possess a dataset $\mathcal{D}$ of training samples distributed according to $P(\boldsymbol{\sigma})$, where each sample in the dataset consists of a N-bit measurement string $(\sigma_1, \ldots, \sigma_N)$. As previously covered in Chapter 2, the cost function is approximated by

$$\mathcal{C}_{\boldsymbol{\lambda}}^{P} \simeq \mathcal{C}_{\boldsymbol{\lambda}}^{\mathcal{D}} = -\|\mathcal{D}\|^{-1} \sum_{\boldsymbol{\sigma} \in \mathcal{D}} \log |\psi_{\boldsymbol{\lambda}}(\boldsymbol{\sigma})|^2 = \log Z_{\boldsymbol{\lambda}} + \|\mathcal{D}\|^{-1} \sum_{\boldsymbol{\sigma} \in \mathcal{D}} \mathcal{E}_{\boldsymbol{\lambda}}(\boldsymbol{\sigma}) \tag{4.10}$$

where we omit the constant data entropy term $\mathbb{H}_{\mathcal{D}}$. This is equivalent to the case of classical data, with the gradient of the cost function equal to

$$\nabla_{\boldsymbol{\lambda}} \mathcal{C}_{\boldsymbol{\lambda}}^{\mathcal{D}} = \|\mathcal{D}\|^{-1} \sum_{\boldsymbol{\sigma} \in \mathcal{D}} \nabla_{\boldsymbol{\lambda}} \mathcal{E}_{\boldsymbol{\lambda}}(\boldsymbol{\sigma}) - \sum_{\boldsymbol{\sigma}} p_{\boldsymbol{\lambda}}(\boldsymbol{\sigma}) \nabla_{\boldsymbol{\lambda}} \mathcal{E}_{\boldsymbol{\lambda}}(\boldsymbol{\sigma}) \,. \tag{4.11}$$

The parameters of the neural wavefunction (i.e. the single RBM) are updated following one of the optimization algorithms covered in Chapter 3, the simplest one being SGD:

$$\boldsymbol{\lambda} \longrightarrow \boldsymbol{\lambda} - \eta \, \nabla_{\boldsymbol{\lambda}} \mathcal{C}_{\boldsymbol{\lambda}}^{\mathcal{D}} \,. \tag{4.12}$$

Therefore, the reconstruction problem here reduces to learning a classical probability distribution $P(\boldsymbol{\sigma}) = \Psi^2(\boldsymbol{\sigma})$, over an exponentially large phase space.

4.1.2 Complex wavefunctions

We now turn to the case of a complex target wavefunction, which requires a more careful approach. Now the quantum states has coefficients

$$\Psi(\boldsymbol{\sigma}) = |\Psi(\boldsymbol{\sigma})| e^{i\phi(\boldsymbol{\sigma})} \,. \tag{4.13}$$

Aside purely complex states, these also include the class of real-valued wavefunctions with a sign structure, where the the real-valued coefficients can assume different signs ($\phi(\boldsymbol{\sigma}) = 0, \pi$). If we now perform projective measurements on the reference basis, the various configurations obtained by collapsing the state are distributed again with probability $P(\boldsymbol{\sigma}) = |\Psi(\boldsymbol{\sigma})|^2$, and thus all the information about the phase $\phi(\boldsymbol{\sigma})$ is lost. Instead, after state preparation, we apply a unitary transformation $\boldsymbol{\mathcal{U}}$ to the state $|\Psi\rangle$ (before the projective measurements), so that the distribution of the measurements $P'(\boldsymbol{\sigma}) = |\Psi'(\boldsymbol{\sigma})|^2$ of the resulting state $\Psi'(\boldsymbol{\sigma}) = \langle \boldsymbol{\sigma} | \boldsymbol{\mathcal{U}} | \Psi \rangle$ contains "some" information about the phases. In general, many different unitaries must be independently applied to extract the full phase structure $\phi(\boldsymbol{\sigma})$. The amount of measurement settings (i.e. number of unitaries) and the structure thereof depends on the particular type of quantum state under reconstruction.

The unitary transformations performed before the projective measurements are equivalent to a change of basis of the many-body Hilbert space $\boldsymbol{\mathcal{H}}$. At this point, we make the assumption of a tensor product structure $\boldsymbol{\mathcal{U}} = \bigotimes_{j=1}^{N} \hat{\mathcal{U}}_j$. This implies that only local changes of bases are allowed. We now introduce the following notation for a generic basis $\{|\boldsymbol{\sigma}^{\boldsymbol{b}}\rangle = |\sigma_1^{b_1}, \ldots, \sigma_N^{b_N}\rangle\}$, where the vector $\boldsymbol{b}$ identifies the local basis b_j for each site j. For instance, if the j-th site is left unchanged by $\boldsymbol{\mathcal{U}}$ (i.e. it is measured in the reference basis), then $b_j = z$. In contrast, if for example $b_j = x$, then the site j is measured in the σ^x basis, and so on. In the new basis, the target wavefunction becomes

$$\begin{aligned} \Psi(\boldsymbol{\sigma}^{\boldsymbol{b}}) = \langle \boldsymbol{\sigma}^{\boldsymbol{b}} | \Psi \rangle &= \sum_{\boldsymbol{\sigma}} \langle \boldsymbol{\sigma}^{\boldsymbol{b}} | \boldsymbol{\sigma} \rangle \langle \boldsymbol{\sigma} | \Psi \rangle \\ &= \sum_{\boldsymbol{\sigma}} \mathcal{U}(\boldsymbol{\sigma}^{\boldsymbol{b}}, \boldsymbol{\sigma}) \Psi(\boldsymbol{\sigma}) \, , \end{aligned} \tag{4.14}$$

and the distribution of the projective measurements is

$$P_{\boldsymbol{b}}(\boldsymbol{\sigma}^{\boldsymbol{b}}) = \left| \sum_{\boldsymbol{\sigma}} \mathcal{U}(\boldsymbol{\sigma}^{\boldsymbol{b}}, \boldsymbol{\sigma}) \Psi(\boldsymbol{\sigma}) \right|^2 \tag{4.15}$$

If the unitary transformations are properly chosen, the distribution in the new basis contains fingerprints of the phase structure, which can be learned by the neural network. Note that $\hat{\mathcal{U}}_j$ quantifies the overlap between the new and old local bases $|\sigma_j^{b_j}\rangle$ and $|\sigma_j^z\rangle$ respectively. The matrix elements of $\boldsymbol{\mathcal{U}}$ are given by the product

$$\mathcal{U}(\boldsymbol{\sigma}^{\boldsymbol{b}}, \boldsymbol{\sigma}) = \prod_{j=1}^{N} \hat{\mathcal{U}}_j(\sigma_j^{b_j}, \sigma_j) = \prod_{j=1}^{N} \langle \sigma_j^{b_j} | \sigma_j \rangle \, . \tag{4.16}$$

Before proceeding to discuss the reconstruction algorithm, let us consider a simple example with only two binary degrees of freedom, and show the importance of the choice of bases, and how it affects the reconstruction procedure.

Unitary transformation for 2 qubits

Let us consider the following generic 2-qubits state:

$$|\Psi\rangle = \sum_{\sigma_0,\sigma_1} \Psi(\sigma_0,\sigma_1)|\sigma_0\sigma_1\rangle \equiv \sum_{\sigma_0,\sigma_1} \Psi_{\sigma_0\sigma_1} e^{i\phi_{\sigma_0\sigma_1}}|\sigma_0\sigma_1\rangle \tag{4.17}$$

with $\Psi_{\sigma_0\sigma_1} \in \mathbb{R}$. By learning the distribution underlying the projective measurements in the reference basis, we can discover the value of the coefficients $\Psi_{\sigma_0\sigma_1}$. To extract the phase, let us perform a simple transformation, changing the local basis of the qubit $j = 0$ from σ_0 to σ_0^x [1]. This is realized by applying the unitary rotation $\boldsymbol{\mathcal{U}} = \hat{\mathrm{H}}_0 \otimes \hat{\mathcal{I}}_1$, where

$$\hat{\mathrm{H}}_j = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \tag{4.18}$$

is called the *Hadamard gate*. A simple way to picture the transformation consists of interpreting the unitary $\boldsymbol{\mathcal{U}}$ as a quantum circuit acting on the state $|\Psi\rangle$ (Fig. 4.1). Given this local change of basis, the wavefunction becomes

$$\begin{aligned} \Psi(\boldsymbol{\sigma^b}) = \Psi(\sigma_0^x,\sigma_1) &= \sum_{\sigma_0} \langle \sigma_0^x|\sigma_0^z\rangle \Psi(\sigma_0,\sigma_1) = \sum_{\sigma_0} \mathrm{H}_0(\sigma_0^x,\sigma_0)\Psi(\sigma_0,\sigma_1) \\ &= \frac{1}{\sqrt{2}}\Big[\Psi(\sigma_0=0,\sigma_1) + (1-2\sigma_0^x)\Psi(\sigma_0=1,\sigma_1)\Big] \\ &= \frac{1}{\sqrt{2}}\Big[\Psi_{0\sigma_1} e^{i\phi_{0\sigma_1}} + (1-2\sigma_0^x)\Psi_{1\sigma_1} e^{i\phi_{1\sigma_1}}\Big]\,. \end{aligned} \tag{4.19}$$

[1] Unless otherwise stated, we always imply $\sigma_j = \sigma_j^z$.

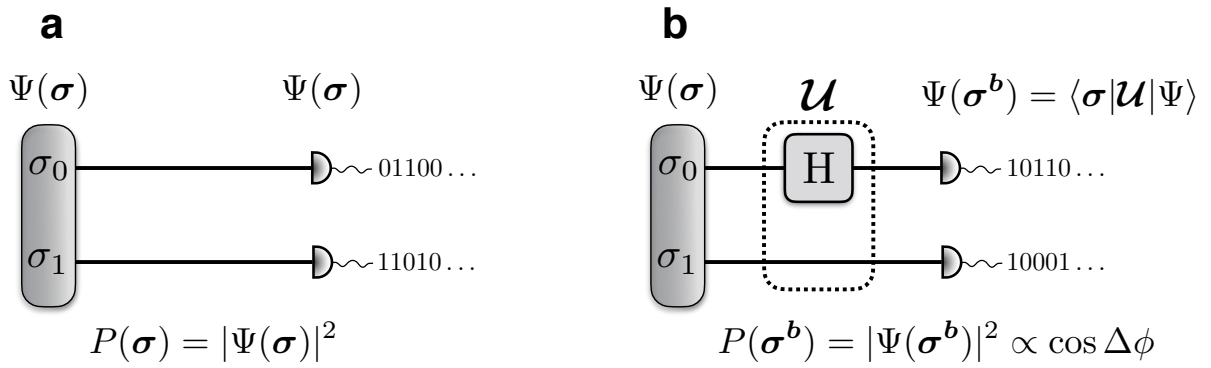

Figure 4.1: **Unitary rotations for 2 qubits**. **a**) Measurements on the reference basis. **b**) Measurement in the rotated basis. The unitary rotation (the Hadamard gate on qubit 0) is applied after state preparation and before the projective measurement.

If we now perform a projective measurement on the quantum state after this rotation, the measurement outcomes are distributed according to the probability

$$\begin{aligned} P_{\boldsymbol{b}}(\sigma_0^x, \sigma_1) = |\Psi(\sigma_0^x, \sigma_1)|^2 &= \frac{1}{4}\left|\Psi_{0\sigma_1} e^{i\phi_{0\sigma_1}} + (1-2\sigma_0^x)\Psi_{1\sigma_1} e^{i\phi_{1\sigma_1}}\right|^2 \\ &= \frac{1}{4}\left[\Psi_{0\sigma_1}^2 + \Psi_{1\sigma_1}^2 + (1-2\sigma_0^x)\Psi_{0\sigma_1}\Psi_{1\sigma_1}\left(e^{i(\phi_{0\sigma_1}-\phi_{1\sigma_1})} + e^{-i(\phi_{0\sigma_1}-\phi_{1\sigma_1})}\right)\right] \\ &= \frac{\Psi_{0\sigma_1}^2 + \Psi_{1\sigma_1}^2}{4} + \frac{1-2\sigma_0^x}{2}\Psi_{0\sigma_1}\Psi_{1\sigma_1}\cos(\Delta\phi)\,, \end{aligned} \tag{4.20}$$

where $\Delta\phi = \phi_{0\sigma_1} - \phi_{1\sigma_1}$. By performing the same rotation on the other qubit we obtain a similar expression, with $\Delta\phi = \phi_{\sigma_0 0} - \phi_{\sigma_0 1}$. We can see that these rotations encode a partial information about the phase structure in the measurement outcomes in the new basis. However, note that the "signal" $\cos(\phi_{0\sigma_1} - \phi_{1\sigma_1})$ is only processed by the RBM if both $\Psi_{1\sigma_1}$ and $\Psi_{0\sigma_1}$ are different from zero. In other words, for the transformation to convey information about the phases, the rotated wavefunction must have a non-zero overlap with the target wavefunction in the reference basis. In fact, if the state under consideration is the singlet state $|\Psi\rangle = (|01\rangle - |10\rangle)/\sqrt{2}$, for example, then the probability in Eq. 4.20 does not depend on the phase difference $\Delta\phi$. In this case, to retrieve the phase structure we

should implement the transformation $\boldsymbol{\mathcal{U}} = \hat{\mathrm{H}}_0 \otimes \hat{\mathrm{H}}_1$. Further, note that the information provided by the cosines of the phase differences is not sufficient, and additional rotations need to implemented. For example, the rotation from σ_j to σ_j^y, using the elementary rotation/gate

$$\hat{\mathrm{K}}_j = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & -i \\ 1 & i \end{bmatrix} , \tag{4.21}$$

leads to a measurement distribution proportional to $\sin \Delta\phi$. This rotation, coupled with the Hadamard gate, provides sufficient information to reconstruct both the amplitudes and phases of the 2-qubits state $|\Psi\rangle$, as long as all coefficients $\Psi_{\sigma_0\sigma_1}$ are different from zero.

Reconstruction algorithm

We present now the reconstruction procedure, assuming that we have prepared a training dataset $\mathcal{D}$ using a combination of different bases. Each instance in the dataset contains a configuration $|\boldsymbol{\sigma}^{\boldsymbol{b}}\rangle$ and the corresponding basis $\boldsymbol{b}$ where the system was measured, such as:

$$|\boldsymbol{\sigma}^{\boldsymbol{b}}\rangle = (1, 0, 1, 0, 0, \ldots, 1, 0, 1) \qquad , \qquad \boldsymbol{b} = (\mathrm{z}, \mathrm{x}, \mathrm{z}, \mathrm{z}, \mathrm{y}, \ldots, \mathrm{z}, \mathrm{x}, \mathrm{z}) \, . \tag{4.22}$$

Similarly to the case of positive wavefunctions, we aim to minimize the statistical divergence between the data and the neural-network distribution. Since we have additional bases, we now require the simultaneous minimization of the KL divergence between the distribution $P_{\boldsymbol{b}}(\boldsymbol{\sigma}^{\boldsymbol{b}})$ and the distribution $|\psi_{\boldsymbol{\lambda\mu}}(\boldsymbol{\sigma}^{\boldsymbol{b}})|^2$ for any measurement basis. The neural wavefunction in the rotated basis is simply obtained as

$$\psi_{\boldsymbol{\lambda\mu}}(\boldsymbol{\sigma}^{\boldsymbol{b}}) = \sum_{\boldsymbol{\sigma}} \mathcal{U}(\boldsymbol{\sigma}^{\boldsymbol{b}}, \boldsymbol{\sigma}) \psi_{\boldsymbol{\lambda\mu}}(\boldsymbol{\sigma}) \, , \tag{4.23}$$

with

$$\psi_{\boldsymbol{\lambda\mu}}(\boldsymbol{\sigma}) = Z_{\boldsymbol{\lambda}}^{-1/2} \widetilde{\psi}_{\boldsymbol{\lambda\mu}}(\boldsymbol{\sigma}) = Z_{\boldsymbol{\lambda}}^{-1/2} \, e^{-[\mathcal{E}_{\boldsymbol{\lambda}}(\boldsymbol{\sigma}) + i\mathcal{E}_{\boldsymbol{\mu}}(\boldsymbol{\sigma})]/2} \tag{4.24}$$

The full cost function is given by

$$\mathcal{C}^{P}_{\boldsymbol{\lambda\mu}} = \sum_{\boldsymbol{b}} \sum_{\boldsymbol{\sigma^b}} P_{\boldsymbol{b}}(\boldsymbol{\sigma^b}) \log \frac{P_{\boldsymbol{b}}(\boldsymbol{\sigma^b})}{|\psi_{\boldsymbol{\lambda\mu}}(\boldsymbol{\sigma^b})|^2} \tag{4.25}$$

and approximated with the dataset as

$$\mathcal{C}^{\mathcal{D}}_{\boldsymbol{\lambda\mu}} = -\|\mathcal{D}\|^{-1} \sum_{\boldsymbol{\sigma^b}\in\mathcal{D}} \log |\psi_{\boldsymbol{\lambda\mu}}(\boldsymbol{\sigma^b})|^2 \tag{4.26}$$

We now plug in the expression of the neural wavefunction:

$$\begin{aligned}
\mathcal{C}^{\mathcal{D}}_{\boldsymbol{\lambda\mu}} &= -\|\mathcal{D}\|^{-1} \sum_{\boldsymbol{\sigma^b}\in\mathcal{D}} \log |\psi_{\boldsymbol{\lambda\mu}}(\boldsymbol{\sigma^b})|^2 \\
&= \log Z_{\boldsymbol{\lambda}} - \|\mathcal{D}\|^{-1} \sum_{\boldsymbol{\sigma^b}\in\mathcal{D}} \Big[\log \widetilde{\psi}_{\boldsymbol{\lambda\mu}}(\boldsymbol{\sigma^b}) + c.c \Big] \\
&= \log Z_{\boldsymbol{\lambda}} - \|\mathcal{D}\|^{-1} \sum_{\boldsymbol{\sigma^b}\in\mathcal{D}} \left[\log \left(\sum_{\boldsymbol{\sigma}} \mathcal{U}(\boldsymbol{\sigma^b}, \boldsymbol{\sigma}) \widetilde{\psi}_{\boldsymbol{\lambda\mu}}(\boldsymbol{\sigma}) \right) + c.c \right] .
\end{aligned} \tag{4.27}$$

The gradients of the unnormalized wavefunction are

$$\nabla_{\boldsymbol{\lambda}} \widetilde{\psi}_{\boldsymbol{\lambda\mu}}(\boldsymbol{\sigma}) = -\frac{1}{2} \widetilde{\psi}_{\boldsymbol{\lambda\mu}}(\boldsymbol{\sigma}) \nabla_{\boldsymbol{\lambda}} \mathcal{E}_{\boldsymbol{\lambda}}(\boldsymbol{\sigma}) \tag{4.28}$$

$$\nabla_{\boldsymbol{\mu}} \widetilde{\psi}_{\boldsymbol{\lambda\mu}}(\boldsymbol{\sigma}) = -\frac{i}{2} \widetilde{\psi}_{\boldsymbol{\lambda\mu}}(\boldsymbol{\sigma}) \nabla_{\boldsymbol{\mu}} \mathcal{E}_{\boldsymbol{\mu}}(\boldsymbol{\sigma}) \tag{4.29}$$

Consequently, we obtain for the full gradient:

$$\begin{aligned}
\nabla_{\boldsymbol{\lambda}}\mathcal{C}^{\mathcal{D}}_{\boldsymbol{\lambda\mu}} &= \nabla_{\boldsymbol{\lambda}}\log Z_{\boldsymbol{\lambda}} - \|\mathcal{D}\|^{-1}\sum_{\boldsymbol{\sigma^b}\in\mathcal{D}}\left[\frac{\sum_{\boldsymbol{\sigma}}\mathcal{U}(\boldsymbol{\sigma^b},\boldsymbol{\sigma})\nabla_{\boldsymbol{\lambda}}\widetilde{\psi}_{\boldsymbol{\lambda\mu}}(\boldsymbol{\sigma})}{\sum_{\boldsymbol{\sigma}}\mathcal{U}(\boldsymbol{\sigma^b},\boldsymbol{\sigma})\widetilde{\psi}_{\boldsymbol{\lambda\mu}}(\boldsymbol{\sigma})} + c.c\right] \\
&= \frac{1}{2}\|\mathcal{D}\|^{-1}\sum_{\boldsymbol{\sigma^b}\in\mathcal{D}}\left[\frac{\sum_{\boldsymbol{\sigma}}\mathcal{U}(\boldsymbol{\sigma^b},\boldsymbol{\sigma})\widetilde{\psi}_{\boldsymbol{\lambda\mu}}(\boldsymbol{\sigma})\nabla_{\boldsymbol{\lambda}}\mathcal{E}_{\boldsymbol{\lambda}}(\boldsymbol{\sigma})}{\sum_{\boldsymbol{\sigma}}\mathcal{U}(\boldsymbol{\sigma^b},\boldsymbol{\sigma})\widetilde{\psi}_{\boldsymbol{\lambda\mu}}(\boldsymbol{\sigma})} + c.c\right] - Z_{\boldsymbol{\lambda}}^{-1}\sum_{\boldsymbol{\sigma}}p_{\boldsymbol{\lambda}}(\boldsymbol{\sigma})\nabla_{\boldsymbol{\lambda}}\mathcal{E}_{\boldsymbol{\lambda}}(\boldsymbol{\sigma}) \\
&= \frac{1}{2}\|\mathcal{D}\|^{-1}\sum_{\boldsymbol{\sigma^b}\in\mathcal{D}}\left[\left\langle\nabla_{\boldsymbol{\lambda}}\mathcal{E}_{\boldsymbol{\lambda}}(\boldsymbol{\sigma})\right\rangle_{Q^{\boldsymbol{b}}} + \left\langle\nabla_{\boldsymbol{\lambda}}\mathcal{E}_{\boldsymbol{\lambda}}(\boldsymbol{\sigma})\right\rangle^{*}_{Q^{\boldsymbol{b}}}\right] - \left\langle\mathcal{E}_{\boldsymbol{\lambda}}(\boldsymbol{\sigma})\right\rangle_{p_{\boldsymbol{\lambda}}} \\
&= \|\mathcal{D}\|^{-1}\sum_{\boldsymbol{\sigma^b}\in\mathcal{D}}\mathrm{Re}\left[\left\langle\nabla_{\boldsymbol{\lambda}}\mathcal{E}_{\boldsymbol{\lambda}}(\boldsymbol{\sigma})\right\rangle_{Q^{\boldsymbol{b}}}\right] - \left\langle\mathcal{E}_{\boldsymbol{\lambda}}(\boldsymbol{\sigma})\right\rangle_{p_{\boldsymbol{\lambda}}},
\end{aligned} \tag{4.30}$$

and

$$\begin{aligned}
\nabla_{\boldsymbol{\mu}}\mathcal{C}^{\mathcal{D}}_{\boldsymbol{\lambda\mu}} &= \frac{i}{2}\|\mathcal{D}\|^{-1}\sum_{\boldsymbol{\sigma^b}\in\mathcal{D}}\left[\left\langle\nabla_{\boldsymbol{\mu}}\mathcal{E}_{\boldsymbol{\mu}}(\boldsymbol{\sigma})\right\rangle_{Q^{\boldsymbol{b}}} - \left\langle\nabla_{\boldsymbol{\mu}}\mathcal{E}_{\boldsymbol{\mu}}(\boldsymbol{\sigma})\right\rangle^{*}_{Q^{\boldsymbol{b}}}\right] \\
&= -\|\mathcal{D}\|^{-1}\sum_{\boldsymbol{\sigma^b}\in\mathcal{D}}\mathrm{Im}\left[\left\langle\nabla_{\boldsymbol{\mu}}\mathcal{E}_{\boldsymbol{\mu}}(\boldsymbol{\sigma})\right\rangle_{Q^{\boldsymbol{b}}}\right].
\end{aligned} \tag{4.31}$$

We first need a function that, given some configuration $|\boldsymbol{\sigma^b}\rangle$ in the dataset and the set of network parameters $\boldsymbol{\theta} = \boldsymbol{\lambda}, \boldsymbol{\mu}$, it returns the value of the rotated gradient

$$\left\langle\nabla_{\boldsymbol{\theta}}\mathcal{E}_{\boldsymbol{\theta}}(\boldsymbol{\sigma})\right\rangle_{Q^{\boldsymbol{b}}} = \frac{\sum_{\boldsymbol{\sigma}}Q^{\boldsymbol{b}}(\boldsymbol{\sigma})\nabla_{\boldsymbol{\theta}}\mathcal{E}_{\boldsymbol{\theta}}(\boldsymbol{\sigma})}{\sum_{\boldsymbol{\sigma}}Q^{\boldsymbol{b}}(\boldsymbol{\sigma})} = \frac{\sum_{\boldsymbol{\sigma}}\mathcal{U}(\boldsymbol{\sigma^b},\boldsymbol{\sigma})\widetilde{\psi}_{\boldsymbol{\lambda\mu}}(\boldsymbol{\sigma})\nabla_{\boldsymbol{\theta}}\mathcal{E}_{\boldsymbol{\theta}}(\boldsymbol{\sigma})}{\sum_{\boldsymbol{\sigma}}\mathcal{U}(\boldsymbol{\sigma^b},\boldsymbol{\sigma})\widetilde{\psi}_{\boldsymbol{\lambda\mu}}(\boldsymbol{\sigma})} \tag{4.32}$$

We start by identifying the set of sites where the unitary rotation is non-trivial (different from the identity). Let us assume that the local basis b_ℓ is different from the reference basis ($\hat{\mathcal{U}}_\ell \neq \hat{\mathcal{I}}_\ell$) for the sites $(\tau_1, \ldots, \tau_{N_U})$(with a total of N_U non-trivial unitaries). For instance, if the configuration $|\boldsymbol{\sigma}^{\boldsymbol{b}}_k\rangle$ was measured in the basis $\boldsymbol{b} = (\mathrm{z}, \mathrm{x}, \mathrm{z}, \mathrm{x})$, the site-set

would be $\boldsymbol{\tau} = (2, 4)$. The full unitary matrix $\boldsymbol{\mathcal{U}}$ is then

$$\boldsymbol{\mathcal{U}} = \left[\bigotimes_{\ell \in \boldsymbol{\tau}} \hat{\mathcal{U}}_\ell \right] \otimes \left[\bigotimes_{\ell \notin \boldsymbol{\tau}} \hat{\mathcal{I}}_\ell \right] . \tag{4.33}$$

Note that this unitary acts non-trivially on subspace $\mathcal{H}_U$ of the full Hilbert space, with dimension $\dim(\mathcal{H}_U) = 2^{N_U}$, and it has the following matrix elements:

$$\mathcal{U}(\boldsymbol{\sigma}^{\boldsymbol{b}}, \boldsymbol{\sigma}) = \prod_{j=1}^{N_U} \langle \hat{\sigma}_{\tau_j}^{b_{\tau_j}} | \sigma_{\tau_j} \rangle \prod_{\ell \notin \boldsymbol{\tau}} \delta_{\hat{\sigma}_\ell^{b_\ell}, \sigma_\ell} . \tag{4.34}$$

Let's define an orthonormal basis $|\boldsymbol{S}\rangle = |S_1, \dots, S_{N_U}\rangle$ for the sub-space $\mathcal{H}_U$, with $|S_j\rangle = |\sigma_{\tau_j}\rangle$. Then, the summation reduces to:

$$\sum_{\boldsymbol{\sigma}} \mathcal{U}(\boldsymbol{\sigma}^{\boldsymbol{b}}, \boldsymbol{\sigma}) \widetilde{\psi}_{\boldsymbol{\lambda}\boldsymbol{\mu}}(\boldsymbol{\sigma}) = \sum_{\boldsymbol{S}} \prod_{j=1}^{N_U} \langle \sigma_{\tau_j}^{b_{\tau_j}} | S_j \rangle \left[\left(\bigotimes_{\ell \notin \boldsymbol{\tau}} \langle \sigma_\ell^{b_\ell = z} | \right) \otimes \langle \boldsymbol{S} | \right] | \widetilde{\psi}_{\boldsymbol{\lambda}\boldsymbol{\mu}} \rangle , \tag{4.35}$$

which can be carried out efficiently for moderate values of N_U.

4.1.3 Density operators

Let us now consider now the problem of reconstructing an unknown mixed quantum state described by a density operator $\hat{\varrho}$. Given a basis $\boldsymbol{b}$, the measurements are now distributed according to the probability distribution $P(\boldsymbol{\sigma}^{\boldsymbol{b}}) = \varrho(\boldsymbol{\sigma}^{\boldsymbol{b}}, \boldsymbol{\sigma}^{\boldsymbol{b}})$. Analogously to the case of pure complex wavefunctions, we train the NDO to minimize the total divergence for all the bases:

$$\mathcal{C}^P_{\boldsymbol{\lambda}\boldsymbol{\mu}} = \sum_{\boldsymbol{b}} \sum_{\boldsymbol{\sigma}^{\boldsymbol{b}}} P(\boldsymbol{\sigma}^{\boldsymbol{b}}) \log \frac{P(\boldsymbol{\sigma}^{\boldsymbol{b}})}{\rho_{\boldsymbol{\lambda}\boldsymbol{\mu}}(\boldsymbol{\sigma}^{\boldsymbol{b}}, \boldsymbol{\sigma}^{\boldsymbol{b}})}. \tag{4.36}$$

Under the same assumption of a datasets $\mathcal{D}$ containing snapshots $\boldsymbol{\sigma}^{\boldsymbol{b}}$ taken in different bases $\boldsymbol{b}$, the cost function is approximated as

$$\mathcal{C}^{P}_{\boldsymbol{\lambda\mu}} \simeq \mathcal{C}^{\mathcal{D}}_{\boldsymbol{\lambda\mu}} = \langle \mathcal{L}_{\boldsymbol{\lambda},\boldsymbol{\mu}} \rangle_{\mathcal{D}} = -\|\mathcal{D}\|^{-1} \sum_{\boldsymbol{\sigma}^{\boldsymbol{b}}_k \in \mathcal{D}} \log \rho_{\boldsymbol{\lambda\mu}}(\boldsymbol{\sigma}^{\boldsymbol{b}}_k, \boldsymbol{\sigma}^{\boldsymbol{b}}_k) \tag{4.37}$$

where once again we omit the constant entropy of the data-set.

In order to take the derivative of Eq. (4.37), we first need to rotate the density operator from the original reference basis $\boldsymbol{\sigma}$ to the new basis, obtaining $\hat{\rho}^{\boldsymbol{b}}_{\boldsymbol{\lambda},\boldsymbol{\mu}} = \boldsymbol{\mathcal{U}} \hat{\rho}_{\boldsymbol{\lambda},\boldsymbol{\mu}} \boldsymbol{\mathcal{U}}^{\dagger}$. As in the previous section, the unitary rotation $\boldsymbol{\mathcal{U}}$ is simply given by the product of unitary matrices $\boldsymbol{\mathcal{U}} = \bigotimes_j \hat{\mathcal{U}}_j$, each performing a local change of basis $\hat{\mathcal{U}}_j = \langle \sigma_j^{b_j} | \sigma_j \rangle$. The gradients of the average negative log-likelihood $\langle \mathcal{L}_{\boldsymbol{\lambda},\boldsymbol{\mu}} \rangle_{\mathcal{D}}$ with respect to the network parameters become

$$\nabla_{\boldsymbol{\lambda}} \langle \mathcal{L}_{\boldsymbol{\lambda},\boldsymbol{\mu}} \rangle_{\mathcal{D}} = \|\mathcal{D}\|^{-1} \sum_{\boldsymbol{\sigma}^{\boldsymbol{b}} \in \mathcal{D}} \left[\langle \nabla_{\boldsymbol{\lambda}} \boldsymbol{\Gamma}^{[+]}_{\boldsymbol{\lambda}} + \nabla_{\boldsymbol{\lambda}} \boldsymbol{\Pi}_{\boldsymbol{\lambda\mu}} \rangle_{\boldsymbol{Q}_{\boldsymbol{\sigma}^{\boldsymbol{b}}}} - \langle \nabla_{\boldsymbol{\lambda}} \boldsymbol{\Gamma}^{[+]}_{\boldsymbol{\lambda}} + \nabla_{\boldsymbol{\lambda}} \boldsymbol{\Pi}_{\boldsymbol{\lambda\mu}} \rangle_{\rho_{\boldsymbol{\lambda},\boldsymbol{\mu}}} \right] \tag{4.38}$$

and

$$\nabla_{\boldsymbol{\mu}} \langle \mathcal{L}_{\boldsymbol{\lambda},\boldsymbol{\mu}} \rangle_{\mathcal{D}} = \|\mathcal{D}\|^{-1} \sum_{\boldsymbol{\sigma}^{\boldsymbol{b}} \in \mathcal{D}} \langle i \nabla_{\boldsymbol{\mu}} \boldsymbol{\Gamma}^{[-]}_{\boldsymbol{\mu}} + \nabla_{\boldsymbol{\mu}} \boldsymbol{\Pi}_{\boldsymbol{\lambda\mu}} \rangle_{\boldsymbol{Q}_{\boldsymbol{\sigma}^{\boldsymbol{b}}}} \tag{4.39}$$

The averages

$$\langle \hat{\mathcal{O}} \rangle_{Q^{\boldsymbol{b}}} = \frac{\sum_{\boldsymbol{\sigma}\boldsymbol{\sigma}'} Q^{\boldsymbol{b}}(\boldsymbol{\sigma}, \boldsymbol{\sigma}') \mathcal{O}(\boldsymbol{\sigma}, \boldsymbol{\sigma}')}{\sum_{\boldsymbol{\sigma}\boldsymbol{\sigma}'} Q^{\boldsymbol{b}}(\boldsymbol{\sigma}, \boldsymbol{\sigma}')}. \tag{4.40}$$

with respect to the quasi-probability distributions

$$Q^{\boldsymbol{b}}(\boldsymbol{\sigma}, \boldsymbol{\sigma}') = \mathcal{U}_{\boldsymbol{b}}(\boldsymbol{\sigma}^{\boldsymbol{b}}, \boldsymbol{\sigma}) \rho_{\boldsymbol{\lambda},\boldsymbol{\mu}}(\boldsymbol{\sigma}, \boldsymbol{\sigma}') \mathcal{U}^{*}_{\boldsymbol{b}}(\boldsymbol{\sigma}^{\boldsymbol{b}}, \boldsymbol{\sigma}') \tag{4.41}$$

can be evaluated directly on the samples in the datasets, with the double summation running over 4^t terms for a basis $\boldsymbol{b}$ where there are only t local unitaries $\mathcal{U}_j \neq \boldsymbol{\mathcal{I}}_j$. On the other hand, the average of the log-probability over the full model probability distribution $\langle \nabla_{\boldsymbol{\lambda}} \log \tilde{\rho}_{\boldsymbol{\lambda},\boldsymbol{\mu}}(\boldsymbol{\sigma}, \boldsymbol{\sigma}) \rangle_{\rho_{\boldsymbol{\lambda},\boldsymbol{\mu}}}$ appearing in Eq. (4.38) requires the knowledge of $Z_{\boldsymbol{\lambda}}$ and can be computed exactly only for very small system sizes. For larger N, it is possible to approximate this average by running a Markov chain MC on the distribution $\rho_{\boldsymbol{\lambda},\boldsymbol{\mu}}(\boldsymbol{\sigma}, \boldsymbol{\sigma})$, analogously to the calculation of the negative phase in the standard RBM training.

4.2 Numerical simulations

In this Section, we apply the neural-network QSR technique to a variety of quantum states, demonstrating its performance through numerical simulations. We will present trainings for positive/complex pure state and mixed states, as well as using synthetic data generated with classical simulations and experimental data.

4.2.1 W state

The first quantum state we consider is the W state, a paradigmatic N-qubit multipartite entangled state defined as

$$|\Psi\rangle = \frac{1}{\sqrt{N}}\big(|100\ldots\rangle + |010\ldots\rangle + \cdots + |\ldots 001\rangle\big). \tag{4.42}$$

Since all the coefficients in $|\Psi\rangle$ appear with the same sign, we will adopt the positive neural wavefunction as a parametrization for the quantum state in the reconstruction process. Because of the constrained structure of the state (only N basis states with equal probability $P(\boldsymbol{\sigma}) = 1/N$), it is possible to perform exact sampling of $|\Psi\rangle$ to generate training data. We generate several datasets with an increasing number of synthetic density measurements obtained by sampling the W state in the $\{|\boldsymbol{\sigma}\rangle\}$ basis. After convergence in the training process, we can probe the performance of the reconstruction by first examining the histogram of the frequency of the N components $\big(|100\ldots\rangle, |010\ldots\rangle\ldots\big)$. We use the trained parameters to sample configurations from $|\psi_{\boldsymbol{\lambda}}(\boldsymbol{\sigma})|^2 = p_{\boldsymbol{\lambda}}(\boldsymbol{\sigma})$ (by using standard block Gibbs sampling in the RBM) and compute the histograms of the counts. We show this in Fig. 4.2a for $N = 20$ and $n_h = N$. We show three histograms obtained with a different number of samples in the training dataset. From the histograms, we see that the N components converge to equal frequency as we increase the number of training samples.

This result qualitatively proves the success of the reconstruction. A crude (quantitative) estimation of the performances can be gained by calculating the variance of the average occupation number for each of the states appearing in $|\psi_{\boldsymbol{\lambda}}\rangle$. However, if we want a more

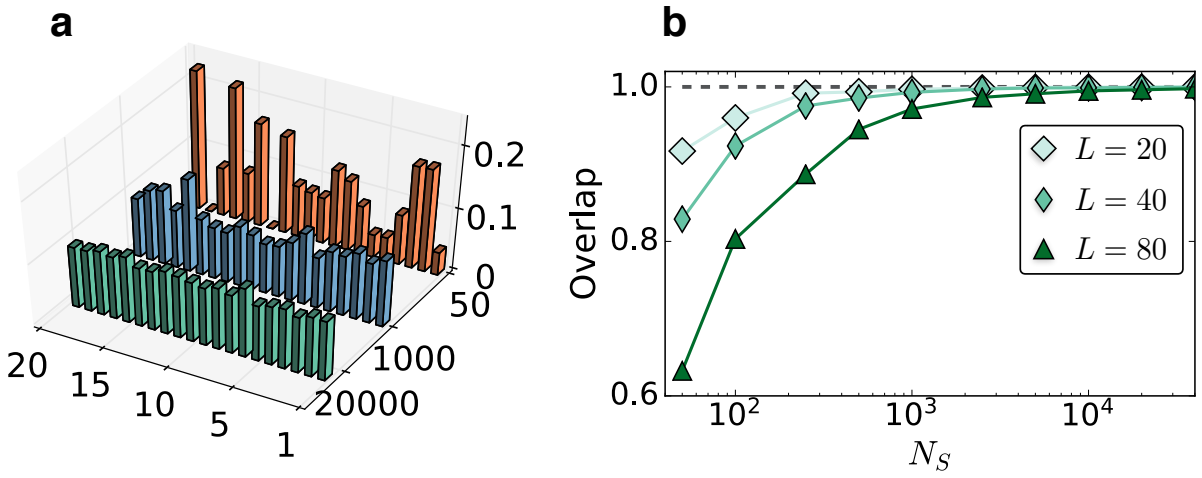

Figure 4.2: **W state**. **a**) We plot, for a different number of training samples N_S, the histogram of the appearance of the N basis states found in the W state. As N_S is increased, the distribution flattens towards its true value. **b**) Overlap between the true and the reconstructed state as a function of the number of training samples.

rigorous criteria, we can instead consider the overlap of the neural wavefunction with the original W state

$$O = |\langle \Psi | \psi_{\boldsymbol{\lambda}} \rangle| = \left| \sum_{\boldsymbol{\sigma}} \Psi(\boldsymbol{\sigma})\, \psi_{\boldsymbol{\lambda}}(\boldsymbol{\sigma}) \right| = Z_{\boldsymbol{\lambda}}^{-\frac{1}{2}} \sum_{\boldsymbol{\sigma}} \Psi(\boldsymbol{\sigma})\, \sqrt{p_{\boldsymbol{\lambda}}(\boldsymbol{\sigma})}\,. \tag{4.43}$$

Unfortunately, in this form, the overlap cannot be directly estimated, since the partition function is not known. Thus, we instead consider the overlap squared (i.e. fidelity)

$$\begin{aligned} \mathcal{F} &= |\langle \Psi | \psi_{\boldsymbol{\lambda}} \rangle|^2 = \langle \Psi | \psi_{\boldsymbol{\lambda}} \rangle \langle \Psi | \psi_{\boldsymbol{\lambda}} \rangle \\ &= \left[\sum_{\boldsymbol{\sigma}} |\psi_{\boldsymbol{\lambda}}(\boldsymbol{\sigma})|^2 \, \frac{\Psi(\boldsymbol{\sigma})}{\psi_{\boldsymbol{\lambda}}(\boldsymbol{\sigma})} \right] \times \left[\sum_{\boldsymbol{\sigma}} |\Psi(\boldsymbol{\sigma})|^2 \, \frac{\psi_{\boldsymbol{\lambda}}(\boldsymbol{\sigma})}{\Psi(\boldsymbol{\sigma})} \right] \\ &= \left\langle \frac{\Psi(\boldsymbol{\sigma})}{\psi_{\boldsymbol{\lambda}}(\boldsymbol{\sigma})} \right\rangle_{|\psi_{\boldsymbol{\lambda}}(\boldsymbol{\sigma})|^2} \times \left\langle \frac{\psi_{\boldsymbol{\lambda}}(\boldsymbol{\sigma})}{\Psi(\boldsymbol{\sigma})} \right\rangle_{|\Psi(\boldsymbol{\sigma})|^2} . \end{aligned} \tag{4.44}$$

In this way, the two contributions of the partition function cancel out and the estimator

can be evaluated simply by sampling both the neural wavefunction and the target state. For this specific case, the target wavefunction has a very simple representation

$$\Psi(\boldsymbol{\sigma}) = \frac{\delta(\boldsymbol{\sigma} - 2^k)}{\sqrt{N}}\,, \tag{4.45}$$

with $k \in [0, \dots, N-1]$. The fidelity reduces to

$$\mathcal{F} = \left[\frac{1}{n}\sum_{k=1}^{n} e^{\mathcal{E}_{\boldsymbol{\lambda}}(\boldsymbol{\sigma}_k)/2} \sum_{q=0}^{N-1} \frac{\delta(\boldsymbol{\sigma}_k - 2^q)}{\sqrt{N}}\right] \times \left[\sum_{q=0}^{N-1} \frac{e^{-\mathcal{E}_{\boldsymbol{\lambda}}(\boldsymbol{\sigma}=2^q)/2}}{\sqrt{N}}\right], \tag{4.46}$$

where the qubits configurations $\boldsymbol{\sigma}_k$ are drawn directly from the trained neural wavefunction's distribution $|\psi_{\boldsymbol{\lambda}}(\boldsymbol{\sigma})|^2 = p_{\boldsymbol{\lambda}}(\boldsymbol{\sigma})$ by performing block Gibbs sampling from the two conditional distributions $p_{\boldsymbol{\lambda}}(\boldsymbol{\sigma}\,|\,\boldsymbol{h})$ and $p_{\boldsymbol{\lambda}}(\boldsymbol{h}\,|\,\boldsymbol{\sigma})$. In Fig. 4.2b we show the overlap $O = \sqrt{\mathcal{F}}$ as a function of the number of samples in the training datasets for three different values of N. For a system size substantially larger than what is currently available in experiments [153], an overlap $O \sim 1$ can be achieved with a moderate number of samples. In particular, for $N = 8$ we found a faithful reconstruction with only about 100 N-bit measurements, a number comparable with state-of-the-art QST [149, 154, 155].

Phase-augmentation

We now include the effect of possible complex phases in the W state, and we do so by adding a local phase shift $\exp(i\theta(\boldsymbol{\sigma}))$ with a random phase $\theta(\boldsymbol{\sigma})$ to each qubit, obtaining

$$|\Psi\rangle = \frac{1}{\sqrt{N}}\Big(\mathrm{e}^{i\theta_1}|100\dots\rangle + \mathrm{e}^{i\theta_2}|010\dots\rangle + \dots + \mathrm{e}^{i\theta_N}|0\dots01\rangle\Big). \tag{4.47}$$

Given the specific structure of the state $|\Psi\rangle$, we require the $(N-1)$ supplementary basis

$$\boldsymbol{b} = (\mathrm{x},\mathrm{x},\mathrm{z},\mathrm{z},\dots)\,,\ (\mathrm{z},\mathrm{x},\mathrm{x},\mathrm{z},\dots)\,,\ (\mathrm{z},\mathrm{z},\mathrm{x},\mathrm{x},\dots),\ \dots \tag{4.48}$$

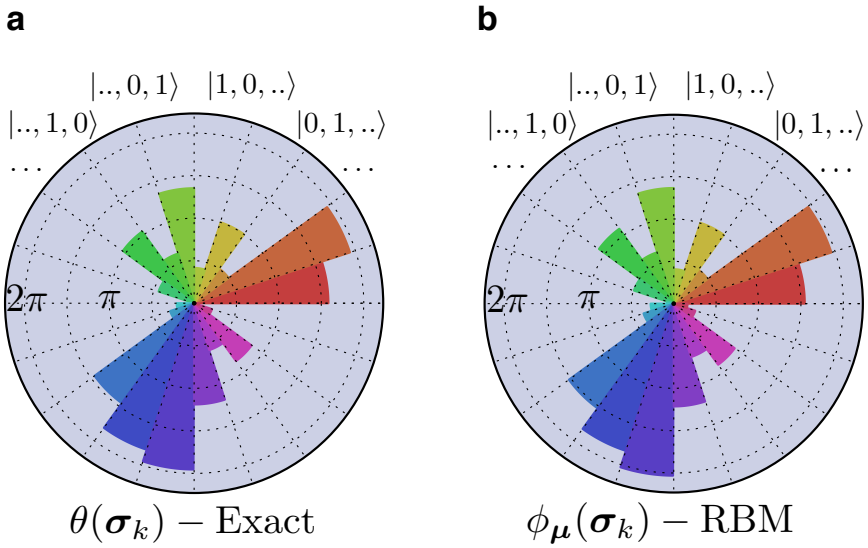

Figure 4.3: **Phase structure for the W state**. We compare the full phase structure between the generated target W state and the trained neural wavefunction for $N = 20$. Each colour represent a different phase (out of the total N), while the value of the phase is plotted along the radial direction.

where in the basis $\{\mathrm{x}_j, \mathrm{x}_{j+1}\}$ we have $|\Psi|^2 \propto \cos(\theta_{j+1} - \theta_j)$, and the $(N-1)$ supplementary basis

$$\boldsymbol{b} = (\mathrm{x}, \mathrm{y}, \mathrm{z}, \mathrm{z}, \dots)\,,\ (\mathrm{z}, \mathrm{x}, \mathrm{y}, \mathrm{z}, \dots)\,,\ (\mathrm{z}, \mathrm{z}, \mathrm{x}, \mathrm{y}, \dots),\ \dots \tag{4.49}$$

where in the basis $\{\mathrm{x}_j, \mathrm{y}_{j+1}\}$ we have $|\Psi|^2 \propto \sin(\theta_{j+1} - \theta_j)$. Therefore, the neural wavefunction is trained on a total of $2N-1$ bases (including the reference basis). The transformation matrices for the j-th basis ($\{X_j, X_{j+1}\}$) and ($\{X_j, Y_{j+1}\}$) are given by $\boldsymbol{\mathcal{U}}^{XX} = \hat{\mathrm{H}}_j \otimes \hat{\mathrm{H}}_{j+1}$ and $\boldsymbol{\mathcal{U}}^{XY} = \hat{\mathrm{H}}_j \otimes \hat{\mathrm{K}}_{j+1}$. We performed the training for $N = 20$ and found a very good agreement with an overlap of $O \simeq 0.997$. In Fig. 4.3 we plot a comparison between the exact phases (**a**) and the phases structure learned by the RBM (**b**). Each of the 20 phases is plotted with a different colour, and the value of the phase is plotted along the radial direction, ranging from from 0 to 2π.

Comparison with other tomographic techniques

We now show the comparison of performances between our QSR method and other tomographic approaches. We do so for the W state, as it is a common choice for benchmarking QSR techniques. The full brute-force tomography, as expected, requires a very large number of measurements (e.g. $N_S \sim 6 \times 10^5$ for an $N = 8$ state [156]). To reduce the measurements, more efficient methods exists, such as compressed sensing [149, 157] and permutationally invariant tomography [154, 158]. It has been shown that using compressed sensing, a drastic reduction of the number of measurements is possible (although still exponentially scaling with N), allowing an efficient reconstruction of the 8 sites W state [149]. Permutationally invariant tomography can in principle be also applied on this problem. Indeed this technique requires a number of measurements which scales as N^2 for the Dicke states [150, 159], and the W state is a particular class of such a states. Compressed sensing in the permutationally invariant subspace can also be combined together to further reduce the number of measurements for the Dicke states [160]. Large phaseless W states have also been reconstructed using MPS-based tomography [155]. In this case we can also quantitatively compare the number of measurement required for the tomography and show that neural network reconstruction is as efficient as such state-of-the art method for the W state (without phases). Following Ref. [155], for $N = 20$ the MPS tomography with $k = 2$ local reductions converges to a fidelity $\mathcal{F} \sim 0.99$ for infinite precision measurements, and drops to an average fidelity $\mathcal{F} \sim 0.96$ if assuming a realistic Gaussian noise on the measurements expectation values. If we consider that MPS tomography needs to reconstruct N 2-sites reduced density matrices, and assuming 100 measurements per basis, we might conclude that one needs roughly $100 \times 4 \times N = 8000$ measurements for an high-fidelity reconstruction. Neural network QSR achieve on the same system size a fidelity of $\mathcal{F} \sim 0.99$ using around 300 (see main text) $N-$qubit measurement, hence $N \times 300 = 6000$ single bit measurements, i.e. a number comparable to MPS tomography.

4.2.2 Quantum spin Hamiltonians

We turn now to the more complex (and interesting) scenario of quantum spin Hamiltonians. We consider again of a set of N interacting quantum spin-$\frac{1}{2}$ placed on the sites of a d-dimensional hyper-cubic lattice. The system is characterized by a set of quantum numbers $\{\sigma_j^z\}$, defining the basis of the Hilbert space $\{|\boldsymbol{\sigma}^z\rangle\}$. Our goal is to reconstruct the ground state wavefunction $\Psi(\boldsymbol{\sigma}^z)$ from a set of projective measurements. In particular, we consider two paradigmatic models of quantum magnetism, which both have a positive ground state.

The first Hamiltonian, briefly mentioned at the end of Chapter 2, is the *transverse-field Ising model* (TFIM). Initially proposed as a model for hydrogen-bonded ferroelectrics [161], the TFIM is a prototypical example of second order quantum phase transition. The Hamiltonian is obtained by simply adding a transverse magnetic field h to the classical Ising model Hamiltonian:

$$\hat{H} = -J \sum_{\langle ij \rangle} \hat{\sigma}_i^z \hat{\sigma}_j^z - h \sum_{i=1}^{N} \hat{\sigma}_i^x \, . \tag{4.50}$$

When the field is absent $h = 0$, the ground state is one of the two degenerate ferromagnetic states $|\uparrow\uparrow, \ldots, \uparrow\rangle$ or $|\downarrow\downarrow, \ldots, \downarrow\rangle$, while for very large fields $h \gg J$ the system is in a paramagnetic phase, with ground state $|\rightarrow\rightarrow, \ldots, \rightarrow\rangle$, where $|\rightarrow\rangle = (|0\rangle + |1\rangle)/\sqrt{2}$. For any dimension d, there exists a quantum critical point h_c separating the two magnetic phases, detected by the magnetization order parameter $\langle \hat{\sigma}^z \rangle$. The second Hamiltonian we consider is the XXZ model, parametrized by the anisotropy Δ of the interactions

$$\hat{H} = \sum_{\langle ij \rangle} \left[\hat{\sigma}_i^z \hat{\sigma}_j^z + \Delta \left(\hat{\sigma}_i^x \hat{\sigma}_j^x + \hat{\sigma}_i^y \hat{\sigma}_j^y \right) \right] . \tag{4.51}$$

For $\Delta = 0$ we find the classical Ising model, while for very large Δ the XXZ model reduces to the XY model, characterized by a $U(1)$ broken symmetry ground state. For $\Delta = 1$ we recover the $SU(2)$ invariant Heisenberg model, shown at the beginning of Chapter 1.

Data generation with QMC

Since we are interested in exploring regimes where exact brute-force approaches are not feasible, we cannot use ED to find the ground state and generate the training datasets. Instead, we employ a discrete time path-integral QCM, where the total imaginary time β is discretized in m_τ steps, so that the simulations are exact in the $\beta/m_\tau \to 0$ limit. As discussed in Chapter 1, the quantum configuration of the N-spin TFIM is mapped onto a configuration of $N \times m_\tau$ classical spin variables, with suitable interactions along the imaginary time direction (see Ref. [162] for details). Classical MC on this $(d+1)$-dimensional system can then be performed in order to collect samples of the quantum distribution in the $\{|\boldsymbol{\sigma}^z\rangle\}$ basis. Since we are interested in the ground state distribution, we use a sufficiently large inverse temperature, in the range $\beta = 10 - 20$ and a converged number of time slices $m_\tau = 1024 - 2048$. Statistically independent samples are collected only after waiting for a sufficiently large number of MC moves, i.e. larger than the autocorrelation time of the Markov chain. In order to decrease the autocorrelations between successive MC configurations we use cluster update algorithms. In the case of the TFIM we use the Wolff single cluster algorithm [129]. Here clusters can be un-restricted in the volume or restricted in such a way to extend only along the imaginary time direction [163, 164]. Both choices drastically improve the efficiency compared to the simple local update scheme. For the XXZ model we use a single cluster update version of the Loop algorithm [165].

Neural-network reconstruction

Using quantum QMC methods, we generate synthetic datasets by sampling the exact ground states of both the TFIM and the XXZ model. We generate a collection of datasets, each at a different value of transverse magnetic field h and the anisotropy Δ. We restrict to the case of a 1d chain with $N = 100$ spins and a 2d square lattice with linear size $L = 12$ (i.e. total number of spins $N = 144$). The neural wavefunctions are trained using CD learning with $k = 20$ sampling steps. The network parameters $\boldsymbol{\lambda}$ are optimized using SGD with a batch size of $M = 100$, a learning rate of $\eta = 0.01$ and a weight decay regularization of $l_2 = 10^{-4}$. The dataset sizes range around $\|\mathcal{D}\| \simeq 10^5$, and the training was carried out

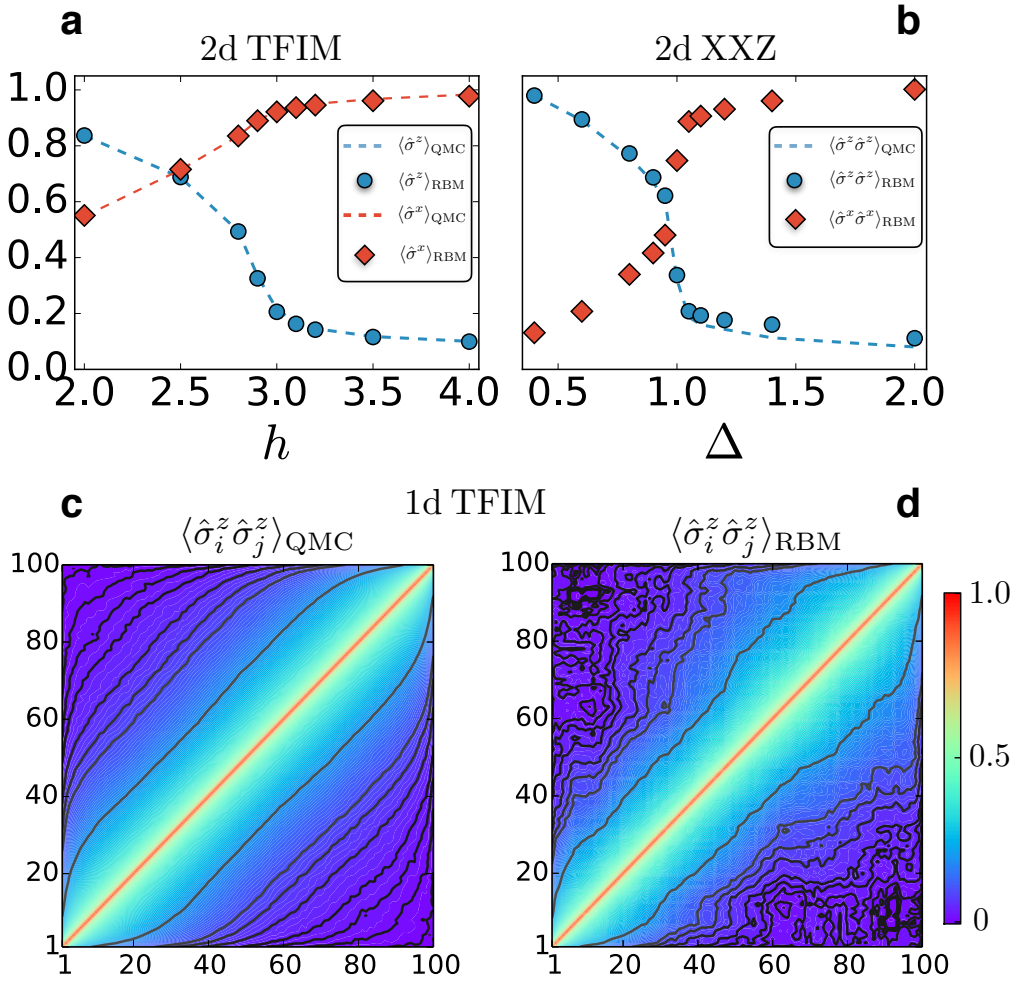

Figure 4.4: **Magnetization for XXZ and Ising models in one and two dimensions**. Diagonal and off-diagonal observables for ground states of quantum spin chains. **a)** Comparison of the average longitudinal $\langle\sigma^z\rangle$ and transverse $\langle\sigma^x\rangle$ magnetization between the trained neural wavefunction (markers) and QMC calculations (lines), for the 2d TFIM with $N = 144$ spins. **b)** Comparison of the average diagonal and off-diagonal correlation function between the spin at the top-left corner and its neighbour on the diagonal for the 2d XXZ model with $N = 144$ spins. Below, we show the comparison of the full spin-spin correlation function for the 1d TFIM with $N = 100$ spins, between the QMC results **(c)** and the reconstructed neural wavefunction **(d)**.

for 2000 epochs. The minimum number of hidden units required to faithfully learn the data distribution was always found to be less than the number of spins, and it was set to $n_h = N/2$ where otherwise stated.

After the training is converged and we discover the optimal set of parameters $\boldsymbol{\lambda}^*$, we proceed to evaluate the performance of the reconstruction. Given the system sizes of interest, we cannot evaluate the overlap between the true and reconstructed wavefunction in a feasible way. Instead, we evaluate the quality of the reconstruction by comparing different magnetic observables computed using the neural wavefunction, with results obtained from the QMC simulations. We show the numerical results of the neural-network QSR in Fig. 4.4. For the TFIM we compare the values of the longitudinal $\langle \hat{\sigma}^z \rangle = \sum_{i=1}^{N} \langle \hat{\sigma}_i^z \rangle$ and transverse (off-diagonal) magnetization $\langle \hat{\sigma}^x \rangle = \sum_{i=1}^{N} \langle \hat{\sigma}_i^x \rangle$, with the QMC estimate obtained with the path-integrals [2]. For a 2d model on a square lattice and $N = 144$ we observe a perfect agreement for either diagonal and off-diagonal magnetizations (Fig. 4.4a). The same agreement is also found for the TFIM in 1d (not shown). For this case however, we show instead the full spin-spin correlation function $\langle \sigma_i^z \sigma_j^z \rangle$, which involves non-local correlations. We compare the QMC result (Fig. 4.4c) with the correlation function reconstructed by the RBM (Fig. 4.4d), observing a good agreement. The deviations between the neural wavefunction and QMC are compatible with statistical uncertainty due to the finiteness of the training dataset.

We now turn to the important and highly-nonlocal quantum quantity that is perhaps the most challenging for direct experimental observation [167], the entanglement entropy. More specifically, we consider the second Renyi entanglement entropy

$$S_2(\hat{\rho}_A) = -\log(\mathrm{Tr}_A(\hat{\rho}_A^2)) \,. \tag{4.52}$$

where $\hat{\rho}_A = \mathrm{Tr}_{A^\perp}(\hat{\rho}) = \mathrm{Tr}_{A^\perp}(|\psi\rangle\langle\psi|)$ is the reduced density matrix for the sub-region A. We examine the same model Hamiltonian, but restrict ourselves in one dimension. We trained the neural wavefunctions with similar hyper-parameters on data generated from the TFIM at three different magnetic fields h and the XXZ at the Heisenberg point $\Delta = 1$. In both

[2] For the derivation of expectation values of off-diagonal operators, please refer to Ref. [166].

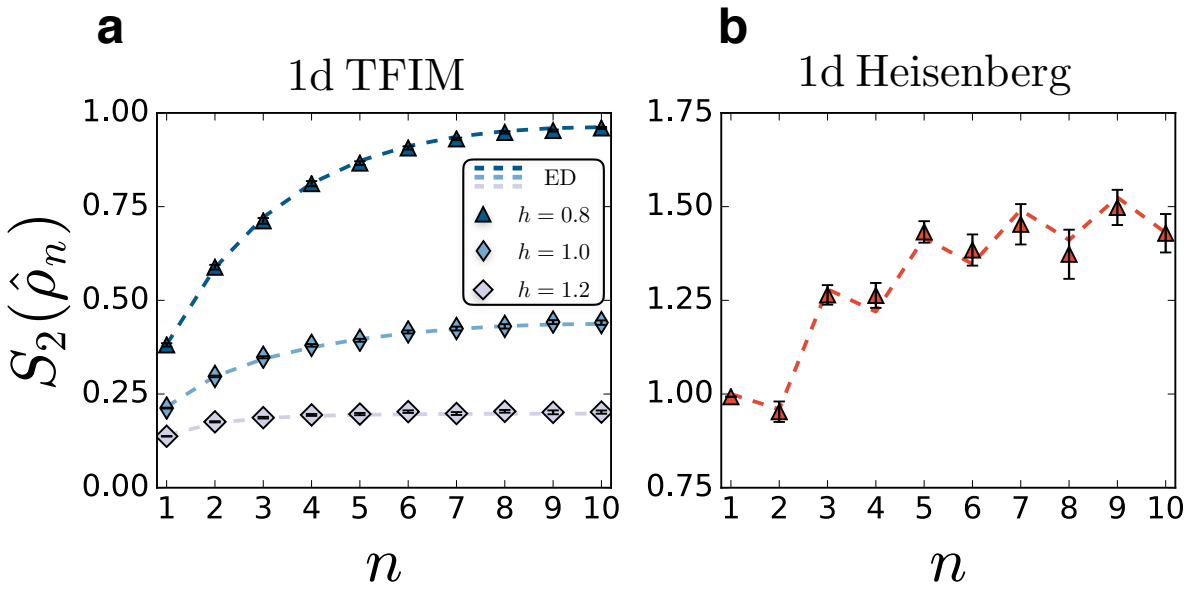

Figure 4.5: **Entanglement entropy via replicated RBMs**. We show the second Renyi entanglement entropy calculated using a replicated copy of the neural wavefunction and the improved ratio trick. We plot the scaling of the entanglement entropy with the size of the subregion A for the 1d TFIM at various strengths of the magnetic field h **(a)**, and for the 1d XXZ model at the Heisenberg point $\Delta = 1.0$ **(b)**. We compare our results against ED calculations (dashed lines).

instances, we consider a chain with $N = 20$ spins and open boundary conditions. After the training, we implement the improved ratio trick on the replicated RBMs (as discussed in Chapter 3), and compare our results with the exact entanglement entropy found with ED simulations. We show the results in Fig. 4.5, where we plot for both model Hamiltonians the entanglement entropy as a function of the size ℓ (i.e. number of sites) of the sub-region A with, $\ell \in [1, N/2]$. From this, we see that values generated from the neural wavefunction agree with the exact entanglement entropy to within statistical errors.

4.2.3 Quench dynamics

We now move beyond ground-state wavefunctions, and we consider quantum states originating from dynamics under unitary time evolution. We focus on the case of quench dynamics that is realizable in experiments with ultra-cold ions [168]. Specifically, we study a system of 1d Ising spins initially prepared in the state

$$|\Psi_0\rangle = |\rightarrow, \rightarrow, \dots, \rightarrow\rangle \,, \tag{4.53}$$

i.e. fully polarized in the $|\boldsymbol{\sigma}^x\rangle$ basis. The system is then time-evolved with unitary dynamics enforced by the Hamiltonian

$$\hat{H} = \sum_{j=1}^{N-1} J_{ij} \hat{\sigma}_i^z \hat{\sigma}_j^z \,, \tag{4.54}$$

with long-range anti-ferromagnetic interactions

$$J_{ij} = \frac{1}{|i-j|^\gamma} \,. \tag{4.55}$$

For a fixed time t, we obtain the time-evolved state using ED

$$|\Psi(t)\rangle = \mathrm{e}^{-\frac{i}{\hbar}\hat{H}t} |\Psi_0\rangle \,. \tag{4.56}$$

We then use the quantum state to build a dataset of spins density measurements from the distribution at time t

$$P_b(\boldsymbol{\sigma}^{\boldsymbol{b}}, t) = |\Psi(\boldsymbol{\sigma}^{\boldsymbol{b}}, t)|^2 = |\langle \boldsymbol{\sigma}^{\boldsymbol{b}} | \Psi(t) \rangle|^2 \tag{4.57}$$

in different bases $\{\boldsymbol{\sigma}^{\boldsymbol{b}}\}$. For this specific state, we employ the following bases configurations:

$$\boldsymbol{b} = (Y, Z, Z, Z, \dots) \,, \, (Z, Y, Z, Z, \dots) \,, \, (Z, Z, Y, Z, \dots), \, \dots \tag{4.58}$$

and

$$\boldsymbol{b} = (X, Z, Z, Z, \dots) \,, \, (Z, X, Z, Z, \dots) \,, \, (Z, Z, X, Z, \dots), \, \dots \tag{4.59}$$

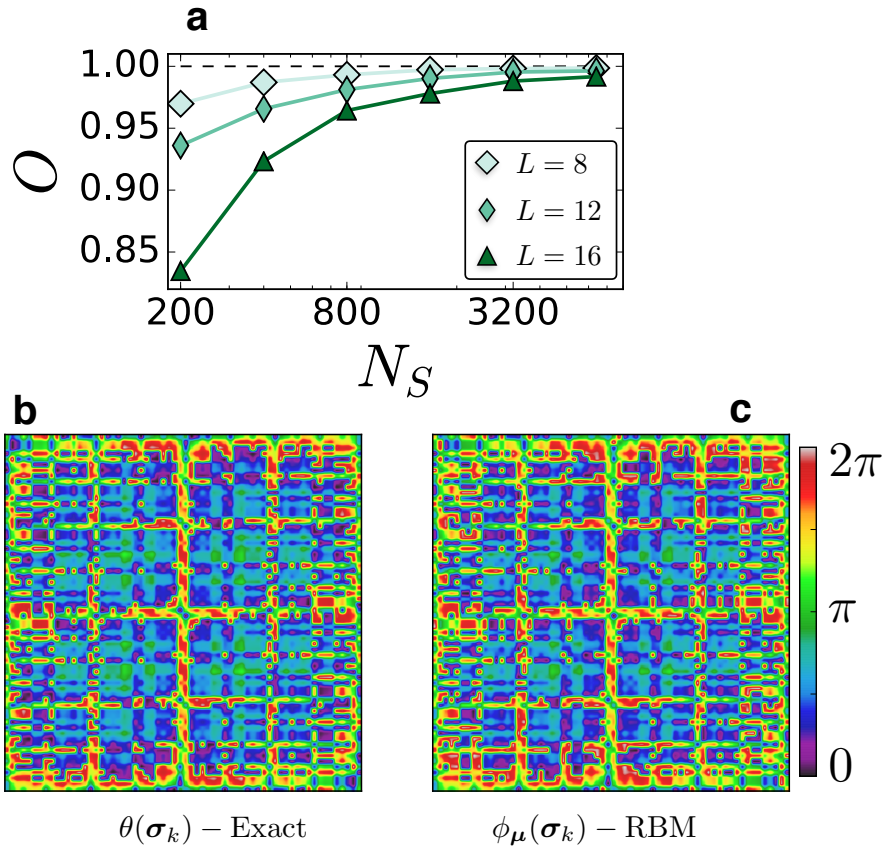

Figure 4.6: **Quench dynamics**. **a**) We show the overlap between the true and reconstructed state as a function of the number of training samples per basis N_s, for three system sizes N. The target state is obtained after time evolution with $t = 0.5$. **b**) Comparison of the full phase structure between ED and the neural wavefunction for $t = 0.5$, for $N = 12$. Each data point in the plot is one of the 2^{12} phase coefficients.

In Fig. 4.6a we show the overlap between the neural wavefunction $\psi_{\boldsymbol{\lambda},\boldsymbol{\mu}}(\boldsymbol{\sigma})$ and the time-evolved state $\Psi(\boldsymbol{\sigma}^z; t = 0.5)$ for different system sizes N, as a function of the number N_S of samples per basis and for $\gamma = 0.75$. Furthermore, in Fig. 4.6b-c, we compare the full phase structure for $N = 12$ (re-arraged as a two-dimensional matrix) for $t = 0.5$, showing an almost perfect agreement between the exact phases and the phase structure reconstructed by a neural wavefunction with $n_h = N$.

4.2.4 Entangled photonic states

We now turn to the reconstruction of the density operator of a mixed state, and we examine the simple case of two entangled photons. Tomographic techniques for this class of quantum states are widely used in a variety of tasks. These include the characterization of optical processes [169], optical detectors [170], and the tests of local realism of quantum mechanics [171, 172]. We first consider the ideal situation where the only fluctuations in the measurement outcomes are of statistical nature. Thus, we generate a synthetic dataset using the exact target quantum state $\hat{\varrho}$. In particular, given the pure Bell state

$$|\Phi^+\rangle = (|00\rangle + |11\rangle)/\sqrt{2}\,, \tag{4.60}$$

we consider the mixed state obtained by applying a global depolarizing channel

$$\hat{\varrho} = (1 - p_{\rm dep})|\Phi^+\rangle\langle\Phi^+| + \frac{p_{\rm dep}}{4}\,\hat{\mathcal{I}}\,. \tag{4.61}$$

The parameter $p_{\rm dep}$ represents the strength of the noise. For $p_{\rm dep} = 0$ we recover the pure state $\hat{\varrho} = |\Phi^+\rangle\langle\Phi^+|$, while for $p_{\rm dep} = 1$ the system is in the maximally-mixed state $\hat{\varrho} \propto \hat{\mathcal{I}}$. We build the datasets by measuring the system in the informationally-complete set of $N_b = 9$ bases $\{\sigma_0^\alpha, \sigma_1^\beta\}$ with $\alpha, \beta = \mathrm{x}, \mathrm{y}, \mathrm{z}$. Further, we generate multiple datasets with a different number N_S of measurements per basis (each containing 2 bits of information). We set the number of hidden units to $n_h = 1$, and initialize the weights with a uniform distribution centered around zero with width $w = 0.01$ (and biases set to zero). The network parameters are updated using AdaDelta optimization algorithm [173] over training batches containing 10 samples, and the best network is discovered by choosing $(\boldsymbol{\lambda}^*, \boldsymbol{\mu}^*)$ for which the average log-likelihood is maximum. We quantify the performance of the reconstruction by computing the fidelity between $\hat{\rho}_{\boldsymbol{\lambda}^*\boldsymbol{\mu}^*}$ and the target density operator $\hat{\varrho}$, which for mixed states is defined as

$$\mathcal{F} = \mathrm{Tr}\left(\sqrt{\sqrt{\hat{\rho}_{\boldsymbol{\lambda}^*\boldsymbol{\mu}^*}}\,\hat{\varrho}\,\sqrt{\hat{\rho}_{\boldsymbol{\lambda}^*\boldsymbol{\mu}^*}}}\right). \tag{4.62}$$

We report in Fig. 4.7 the fidelities obtained after training the NDO for three different

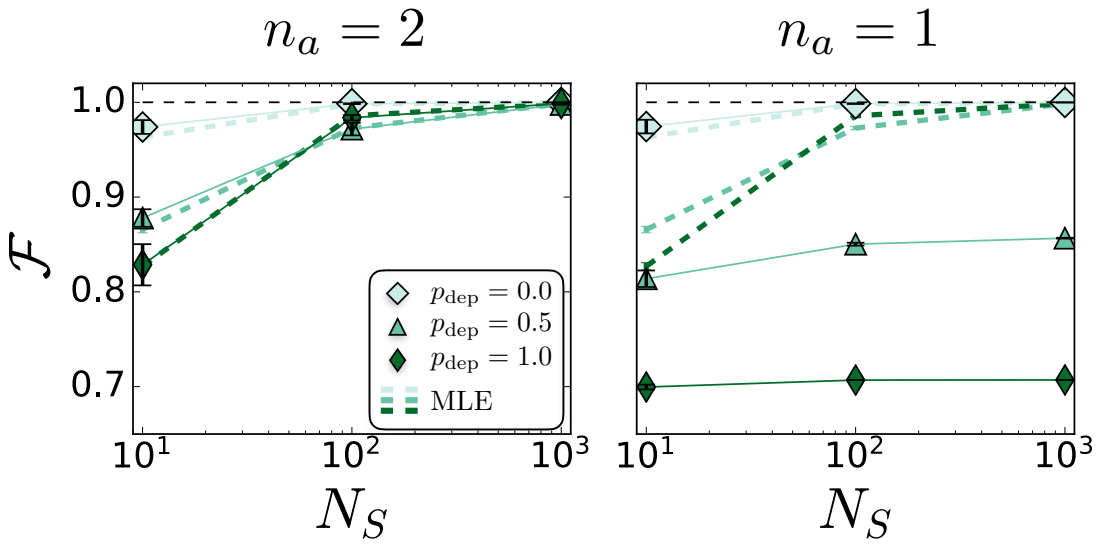

Figure 4.7: **Reconstruction of a depolarized Bell state**. Comparison of the reconstruction fidelities between NDO and MLE tomography for a Bell state $|\Phi^+\rangle$ undergoing a depolarizing channel with strength $p_{\rm dep}$. We show the scaling of the fidelity as a function of the number of measurements per basis N_S for two different choices of network structure (each point is plotted with standard deviation error from an average over 100 realizations of the dataset).

depolarizing strengths and an increasing number of measurements per basis. We compare our results with the fidelities obtained with standard MLE tomography [174, 146]. We observe slightly better fidelities when using two auxiliary units (Fig. 4.7a), while the NDO with $n_a = 1$ is not capable of purifying the state of the physical system (Fig. 4.7b). This is not surprising, since for a generic mixed state, the Hilbert space of the auxiliary system used in the purification should have at least the same dimension of the physical Hilbert space.

We now consider the case of real experimental data, where unknown sources of noise are present. Using the coincidence counts provided by Ref. [175], we perform NDO tomography on the experimental measurements for the state $\hat{\varrho} = |\Psi\rangle\langle|\Psi|$ with

$$|\Psi\rangle = \frac{1}{\sqrt{2}}(|00\rangle + i|11\rangle)\,, \tag{4.63}$$

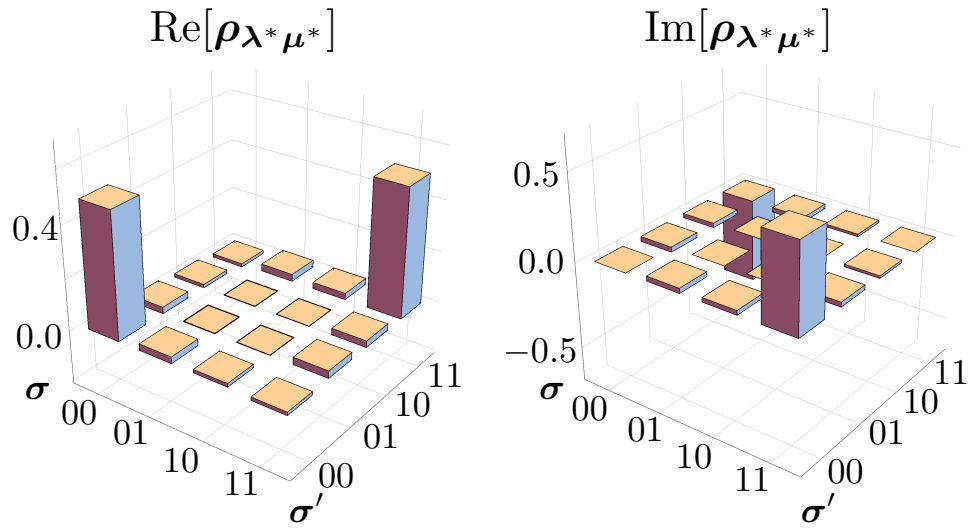

Figure 4.8: **Experimental reconstruction of a density matrix**. Real and imaginary part of the reconstructed NDO, trained on experimental coincidence counts for the quantum state $|\Psi\rangle = \frac{1}{\sqrt{2}}(|00\rangle + i|11\rangle)$ of two entangled photons.

where the degrees of freedom represent the polarizations of the entangled photons. In Fig. 4.8 we plot the real and imaginary part of the reconstructed NDO, selected with the same criterion of minimum negative log-likelihood. The fidelity between the NDO and the ideal state is found to be $\mathcal{F}_{\text{NDO}} = 0.9976$, with MLE tomography achieving similar fidelity $\mathcal{F}_{\text{MLE}} = 0.992$.

4.2.5 Experimental quantum simulator

In Chapter 1 we have briefly discussed the quantum simulation of a spin Hamiltonians $\hat{H}$ with a superconducting quantum hardware. The unitary time evolution was programmed as a sequence of quantum gates applied to some reference initial state of the hardware, with the output of the device being the time-evolved state $\psi(t) = e^{-i\hat{H}t/\hbar}\psi(0)$. This type of quantum simulation is called *digital*. A different paradigm is provided by *analog quantum simulators*. In this case, the simulation is done using another physical system with different Hamiltonian $\hat{H}_{\text{sim}}$, which can be tuned to reproduce the desired properties of the physical

system of interest. If there is a mapping between $\hat{H}$ and $\hat{H}_{\rm sim}$, then this approach allows to emulate the dynamics $\hat{U}(t) = e^{-i\hat{H}t}$ using the time-evolution propagator $\mathcal{U}_{\rm sim}(t) = e^{-i\hat{H}_{\rm sim}t}$ on the emulating physical system. Two examples of analog quantum simulators are ultra-cold quantum gases in optical lattice [176] and trapped ions [177, 178]. In the following, we will present the neural-network QSR of a quantum simulator based on trapped neutral Rydberg atoms [179].

Quantum simulation with Rydberg atoms

The (emulating) physical system consists of a $1d$ array of N cold ^{87}Rb atoms, where coherent couplings to excited Rydberg states realize effective repulsive van der Waals interactions $V_{ij} \propto R_{ij}^{-6}$, with R_{ij} the distance between atom i and j. The capability of preparing defect-free atomic arrays [180] and the highly controlled nature of the interactions make Rydberg atomic systems suitable for quantum simulation. Given the two-dimensional local Hilbert space spanned by the ground and excited Rydberg state $\{|g\rangle, |r\rangle\}$, the array of atoms under consideration interact with Hamiltonian

$$\hat{H} = \frac{\Omega}{2} \sum_{j=1}^{N} \hat{\sigma}_j^x - \Delta \sum_{j=1}^{N} \hat{n}_j + \sum_{i<j} V_{ij} \hat{n}_i \hat{n}_j \tag{4.64}$$

where $\hat{n}_j = |r_j\rangle\langle r_j|$ and $\hat{\sigma}_j^x = |g_j\rangle\langle r_j| + |r_j\rangle\langle g_j|$. The parameter Δ is the detuning (i.e. the difference in frequency between the driving lasers and the Rydberg excited state), while Ω is the Rabi frequency (i.e. the difference in frequency between the ground and the Rydberg excited state). The strengths of interactions V_{ij} can be set by changing the distance between the atoms. These interaction implement and effective blockade which prevents nearby atoms to be simultaneously excited in the Rydberg states [181].

This array of cold atoms can be readily used to perform quantum simulation of a Ising-like model. In fact, the Hamiltonian can be easily re-written in the following form

$$\hat{H} = \frac{\Omega}{2} \sum_{j=1}^{N} \hat{\sigma}_j^x + \frac{\Delta}{2} \sum_{j=1}^{N} \hat{\sigma}_j^z + \frac{1}{4} \sum_{i<j} V_{ij} \Big(\hat{\sigma}_i^z \hat{\sigma}_j^z - \hat{\sigma}_i^z - \hat{\sigma}_j^z \Big) + \text{ const. } , \tag{4.65}$$

which is an anti-ferromagnetic Ising model interacting with both a longitudinal and transverse magnetic field. Depending on the relative value of the Hamiltonian parameters, the ground state of the system can exhibit different spacial symmetries. For a large and negative detuning Δ, with $|\Delta/\Omega| \gg V_{ij}$, the ground state contains all atoms in the ground state

$$|\Psi_0\rangle = |g_0, g_1, \ldots, g_{N-1}\rangle \, . \tag{4.66}$$

In turn, when the detuning is increased, the number of atoms in the excited states also increase, activating the Rydberg blockade caused by the van der Waals interactions. As such, by carefully choosing the relative value between the detuning and the interaction strengths V_{ij}, it is possible to prepare ground states behaving as (Rydberg) crystals, with different broken spacial symmetries [182].

Among the various phases that can be engineered with this quantum simulators, we are particularly interested in the $\mathbb{Z}_2$ anti-ferromagnetic order. To achieve that, in the experimental setup, the Rabi frequency is set to $\Omega = 2$MHz and the nearest-neighbour interaction is set to $V_{j,j+1} = 30$MHz. When the detuning, which is kept as a free parameter for the experiment, is set to a value such that $V_{j,j+1} \gg \Delta \gg \Omega \gg V_{j,j+2}$, the Rydberg blockade acts on neighbouring sites and is negligible for next-to-nearest neighbours. As such, the ground state of the Hamiltonian breaks $\mathbb{Z}_2$ translational symmetry. Higher order symmetry, such as $\mathbb{Z}_3$ and $\mathbb{Z}_4$ have also been realized with the same experimental setup [183]. The realization of the ground state in the $\mathbb{Z}_2$ phase requires a slow adiabatic evolution with time-evolution propagator

$$\hat{U}(t) = e^{-i\hat{H}(\Delta(t))t/\hbar} \, , \tag{4.67}$$

where the detuning follows a ramp in the range $\Delta \in [-10, 10]$MHz. In practice, the atoms are trapped by tweezers using an acousto-optic deflector, and initialized to the ground state $|g_j\rangle$ for large negative Δ [183]. As the trap is switched off, the system follows the dynamics $\hat{U}(t)$ until the measurements, consisting of imaging the atoms in the ground state with atomic fluorescence.

Neural-network quantum reconstruction

The reference basis where we define the neural wavefunction is simply $|\boldsymbol{\sigma}\rangle = |\sigma_1, \ldots, \sigma_N\rangle$, where $\sigma_j = 0$ and $\sigma_j = 1$ corresponds to atoms j being in the ground and Rydberg state respectively. The main assumption is that the quantum state prepared with the detuning sweep is pure and positive. This means that the state is described by the state

$$|\Psi_\Delta\rangle = \sum_{\boldsymbol{\sigma}} \Psi_\Delta(\boldsymbol{\sigma})|\boldsymbol{\sigma}\rangle \, . \tag{4.68}$$

with real coefficients $\Psi_\Delta(\boldsymbol{\sigma}) > 0$. The positivity is enforced by the Hamiltonian, since the off-diagonal elements can be gauged to be negative with a canonical transformation. However, in general, complex phases can in principle build up during the adiabatic evolution. We will however perform the neural-network reconstruction using a positive neural wavefunction $\psi_{\boldsymbol{\lambda}}$. While in principle we could include arbitrary phases, the lack of measurements in additional bases prevent us to learn the additional set of parameters $\boldsymbol{\mu}$ capturing the phase structure. Thus, we perform the reconstruction using a set of projective measurements in the reference basis $|\boldsymbol{\sigma}\rangle$, distributed according to $P(\boldsymbol{\sigma}) = |\langle\boldsymbol{\sigma}|\Psi_\Delta\rangle|^2 = |\Psi_\Delta(\boldsymbol{\sigma})|^2$. In our investigation, we consider a chain of $N = 8$ atoms. An experiment was realized to collect the necessary data, consisting of a collection of datasets $\mathcal{D}_\Delta$ containing approximately 3000 measurements (as N-bit strings) for various detuning Δ across the quantum phase transition. We verified the convergence in dataset size by training the neural wavefunctions on different subsets of data with increasing number of measurements. The results that follows are obtained by training the neural wavefunction on the full datasets using a fixed number of hidden units $n_h = N = 9$.

We first look at the domain wall distribution in the system, which can be used to characterize the phase transition. A domain wall is defined as two neighbouring atoms being in the same state, or the atoms at the edge being in the ground state. We show in Fig. 4.9a the domain wall density as a function of the detuning. We plot the observable computed directly on the data (black line), the RBM reconstruction (markers) and a simulation of the model Hamiltonian $\hat{H}$ using ED (dashed line). In the disordered phase

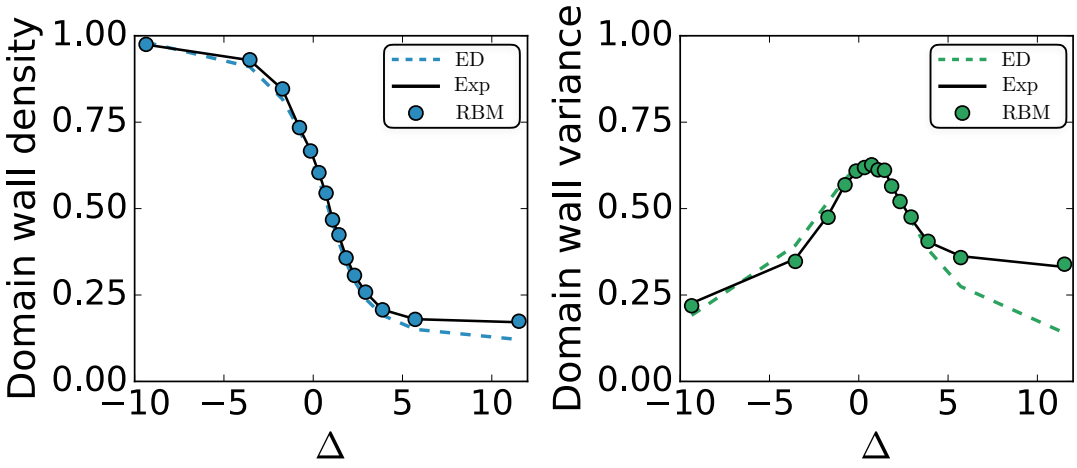

Figure 4.9: **Domain walls**. We show the domain wall density **(a)** and its variance **(b)**. We compare the RBM reconstruction with the experimental data and ED simulations of the model Hamiltonian.

(large and negative Δ), all the atoms are in the ground state and the number of domain wall is maximum. When the system is brought closer to the $\mathbb{Z}_2$ phase, the density decreases towards its minimum. The data and ED curves are perfectly matched by the RBM with a number of hidden units as low as $n_h = 1$ (not shown). We also look at the variance of the domain wall density, which has a peak at the phase transition. We expect the reconstruction to require more resources compared to the domain wall density, since we are now looking at the fluctuations of an observables. By performing a scaling with the size of the latent space of the network (i.e. the number of hidden units n_h) we observe that the size required for convergence is slightly larger than the one for the density ($n_h = 1$). In Fig. 4.9b we show the same comparison, and note deviations as the system is closer to the $\mathbb{Z}_2$ phase.

We now turn to magnetic observables. In Fig. 4.10 we plot the average longitudinal $\langle \sigma^z \rangle$ and transverse $\langle \sigma^x \rangle$ magnetizations per site. We repeat the same analysis performed for the domain wall density. For the longitudinal magnetization we find a very good agreement with both the data and the ED simulations. For the transverse magnetization, we cannot compare with the experimental data, since the expectation value depends on the coherences

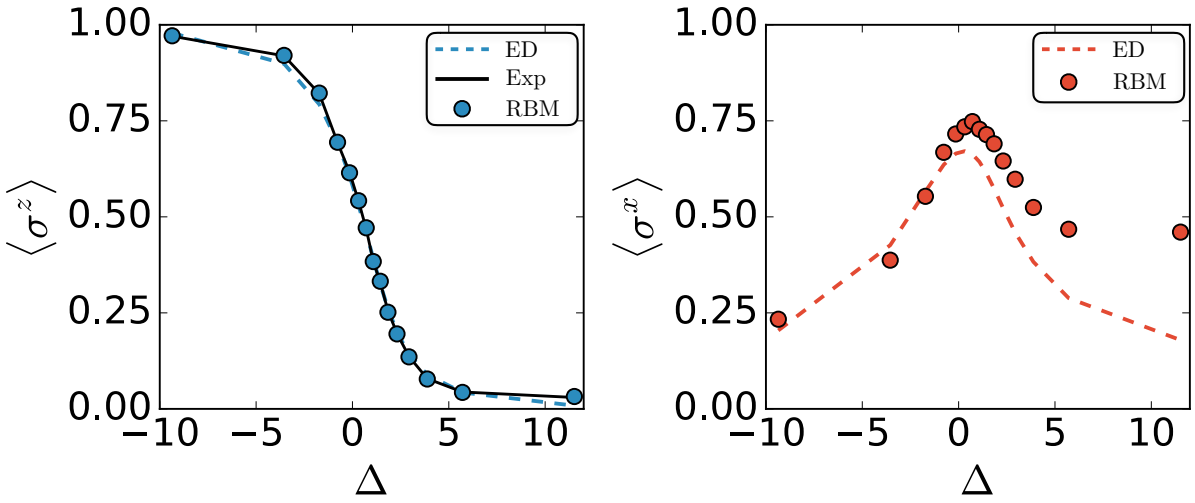

Figure 4.10: **Magnetizations**. Average of the longitudinal **(a)** and transverse **(b)** magnetizations per site. For the latter, comparison with the data is not possible, being an off-diagonal observable.

of the quantum state and it is not explicitly contained in the dataset. Similar to the variance of the domain wall density, we perform a scaling with the number of hidden units, and find convergence for $n_h < N$. The result also shows a residual transverse magnetization in the $\mathbb{Z}_2$ phase. This plateau, as well as the deviation of the domain wall variance, is caused by two main factors. On the one hand, there are errors during the measurement process, which for the current setup consists of a non-symmetric channel with probabilities $p_g \simeq 1\%$ and $p_r \simeq 7\%$ of detecting a false ground and excited state respectively. On the other hand, there is some inevitable degree of open system quantum dynamics during the adiabatic sweep. This means that quantum correlations build up between the atomic chain and the surrounding environment, leading to decoherence. A simple example of decoherence in this setup is the spontaneous decay of an atom from the Rydberg excited states to the ground state.

We finally consider the measurement of entanglement, which is out of the experimental reach. As seen before, we take a bipartition of the chain into two sub-regions with size n and $N-n$. Since we are dealing with a small system size, instead of using the replica trick

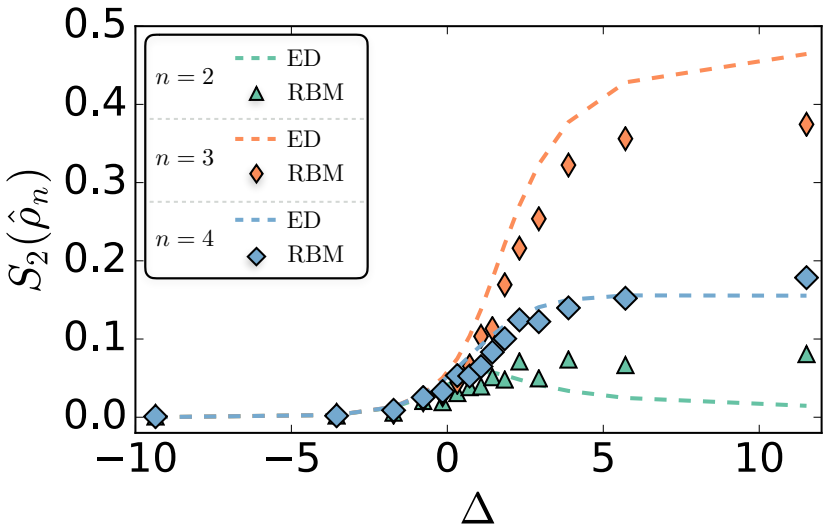

Figure 4.11: **Entanglement entropy**. We compute the exact entanglement entropy from the trained neural wavefunction. We plot the comparison of the reconstructed entanglement entropy as a function of detuning, with the exact results from ED. We show the entropy for three different sizes of the sub-region.

we compute the full reduced density matrix from the trained neural wavefunction $|\psi_{\boldsymbol{\lambda}}\rangle$,

$$\hat{\rho}_{\boldsymbol{\lambda}}^{[n]} = \mathrm{Tr}_{\sigma_{n+1},\ldots,\sigma_N}(|\psi_{\boldsymbol{\lambda}}\rangle\langle\psi_{\boldsymbol{\lambda}}|)\,. \tag{4.69}$$

Given the reduced density matrix $\hat{\rho}_{\boldsymbol{\lambda}}^{[n]}$, the second Renyi entropy is just:

$$S_2(\rho_n) = -\log \mathrm{Tr}_{\sigma_1,\ldots,\sigma_n}\left([\hat{\rho}_{\boldsymbol{\lambda}}^{[n]}]^2\right) \tag{4.70}$$

In Fig. 4.11 we show the entanglement entropy as a function of the detuning Δ, for three different sizes of the sub-region. We compare the results obtained from the neural-network reconstruction with the calculations from ED. While in the paramagnetic phase we observe an almost perfect agreement, deep in the $\mathbb{Z}_2$ phase we observe some deviations, but an overall correct behaviour. Once again, this is caused by the noise introduced by the environment.

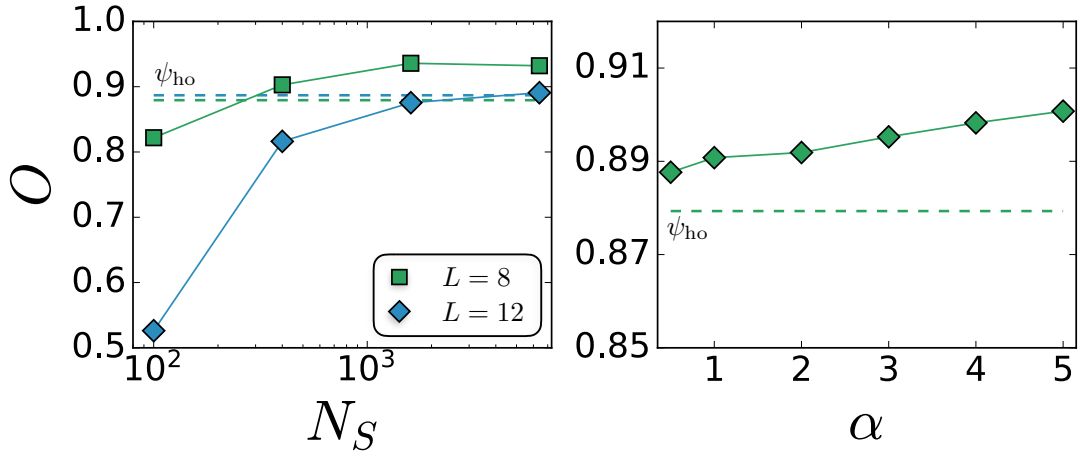

Figure 4.12: **Random states**. Overlap between the neural wavefunction and the random state, as a function of the number of training samples with $n_h = N$ (**a**) and the parameter $\alpha = n_h/N$ with $N_S = 6400$ (**b**). Dashed lines represent the overlap between the random state and an homogeneous state $\psi_{\rm ho}(\boldsymbol{\sigma}) = 1/\sqrt{2^N}$.

4.3 Remarks

Random states and overfitting

All the target quantum systems considered so far are characterized by a structured quantum state, which is the reason for the very high performances of neural-network QSR. In contrast, we now turn to a completely unstructured case and test the limitation of this technique on a random quantum state generated from the Haar measure over the unitary group. To this end we employ a built-in function of the Python library QuTiP [184] for the system sizes $N = 8, 12$. For simplicity, we restrict again to the case of a positive state in the reference basis $\{|\boldsymbol{\sigma}\rangle\}$, and attempt to learn the amplitudes $\Psi(\boldsymbol{\sigma})$. In Fig. 4.12 we show the scaling of the overlap between the neural wavefunction and the random state with the number of training samples N_S for $\alpha = n_h/N = 1$ (**a**), and the scaling with α for $N_S = 6400$ (**b**). We also compare our results with the overlap between the target random state and an homogeneous state $\psi_{\rm ho}(\boldsymbol{\sigma}) = 1/\sqrt{2^N}$ (dashed lines). For $N = 12$ the overlap saturates very quickly to the value obtained from the homogeneous wavefunction, and the

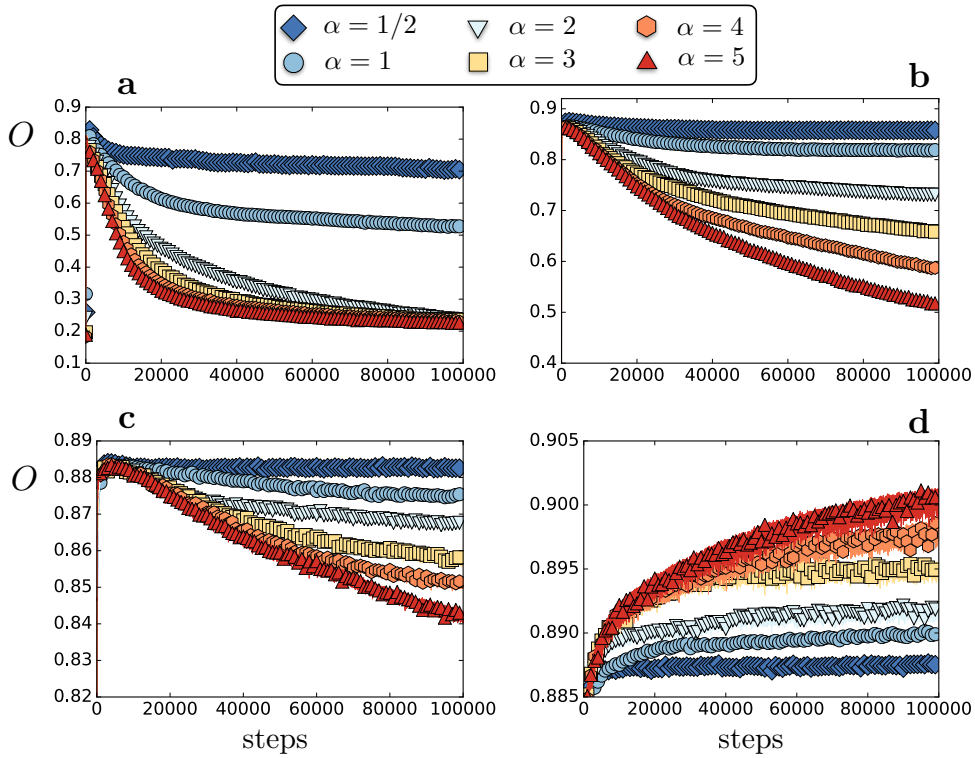

Figure 4.13: **Overfitting random states**. Overlap between the neural wavefunction and the random state as a function of the training steps for $N_S = 100$ (**a**), $N_S = 400$ (**b**), $N_S = 1600$ (**c**) and $N_S = 6400$ (**d**).

improvement upon increasing the number of parameters of the RBM is very small. Such degrading of accuracy is indeed expected for such a class of target quantum states. The success of ML techniques stems from the ability to learn and extract some hidden structure in the data. ML based techniques for QSR will in general fail when there is little or no structure in the quantum state. Therefore, random or very chaotic states remain outside the capability of this method.

As per all ML applications, the training process should be carefully designed to avoid overfitting. This issue occurs when a complex model does not generalize well to unseen data, even though the model fits well the training data. In our experiments with RBMs, overfitting may manifest itself when the model is excessively powerful, i.e. when $n_h \gg N$,

and/or when the datasets used during the training stage are statistically small. We consider the effect of overfitting in the RBM training for this class of random quantum states. In Fig. 4.13 we plot the value of the overlap between the neural wavefunction and the random state as a function of the training step, for four different number N_S of training samples. When N_S is too small (**a**-**c**), the overlap decreases as the parameter α grows larger. Furthermore we observe that the overlap keeps decreasing with further training of the RBM, a common signature of overfitting encountered in many ML experiments and usually solved by an early stopping mechanism. By comparing the first three panels we can see how this effect becomes smaller as the number of training samples increases, and disappears for $N_S = 6400$ (**d**). In the latter case, a network with larger α achieves higher performances, thus showing that there are enough samples in the dataset to eliminate the overfitting issue from the training.

Reconstruction errors for NDO

In this section we consider the different sources of error affecting the representation and reconstruction of a quantum state, considering NDOs. We provide a detailed analysis for the case of the Bell state $|\Psi^+\rangle$ under global depolarizing noise, with density matrix

$$\hat{\varrho} = (1 - p_{\text{dep}})|\Psi^+\rangle\langle\Psi^+| + p_{\text{dep}}\frac{\hat{\mathcal{I}}}{4}\,. \tag{4.71}$$

We start by considering the *representational ability*, which is quantified by the amount of classical resources required to parametrize the target state $\hat{\varrho}$. A natural choice for a convergence parameter is given by the size of the NDO, expressed in terms of $\alpha_h = n_h/N$ and $\alpha_a = n_a/N$, where $N = 2$ is the number of qubits and n_h and n_a the number of hidden and auxiliary units respectively. Larger values of α_h and α_a allows the NDO to capture increasingly complex correlations within the physical system and between the system and the environment respectively. Thus, we expect the minimum number of auxiliary units n_a for a faithful representation to be directly related to the purity of the target state. Inefficiency in the NDO representation can be detected by observing higher performances upon increasing α_h and α_a, which can be quantified by the fidelity $\mathcal{F}$ with the true target

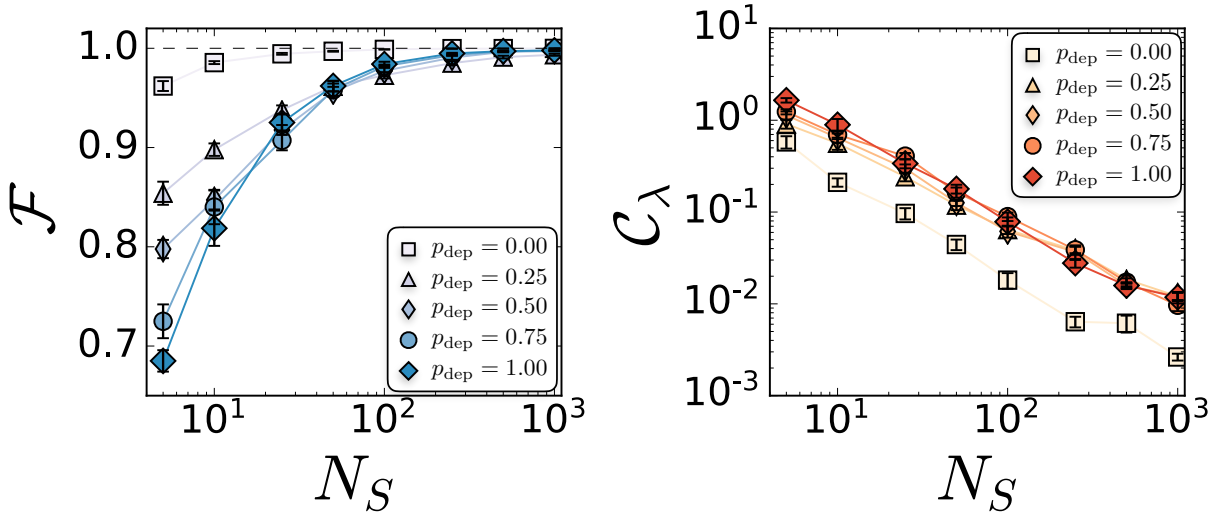

Figure 4.14: **Measurements insufficiency**. Error due to number of measurements. We plot the scaling of the fidelity and total KL divergence $\mathcal{C}^{\mathcal{D}}_{\boldsymbol{\lambda\mu}}$ with the number of measurements per basis N_S for different depolarizing strengths p_{dep}.

state, for example. This is shown in Fig. 4.7, through the scaling of the fidelities with the number of measurements per basis N_S at different strengths of the noise channel p_{dep}. We show that, by training two NDOs using $n_a = 1, 2$ (and $n_h = 1$), the NDO with one auxiliary unit does not have enough representational ability to describe the state when mixing with the environment is present. This effect becomes more severe as the mixing rate increases (indicated by lower fidelities). For a quantum state with arbitrary mixing, an upper bound on the number of auxiliary units is given by the number of qubits $n_a \leq N$, since any mixed quantum state can be purified by using an auxiliary system whose Hilbert space has dimension equal to the physical Hilbert space. The same type of scaling argument also applies to the hidden units n_h, and such investigation determines when the representation of the quantum state is minimal.

A second source of error resides in *insufficient training data*. In this case, the error can be attributed to two factors: the number of measurements, and the measurement settings (i.e. the choice of bases). For the type of reconstruction algorithm we have proposed, the number of measurements required to reconstruct a given quantum state depends mostly

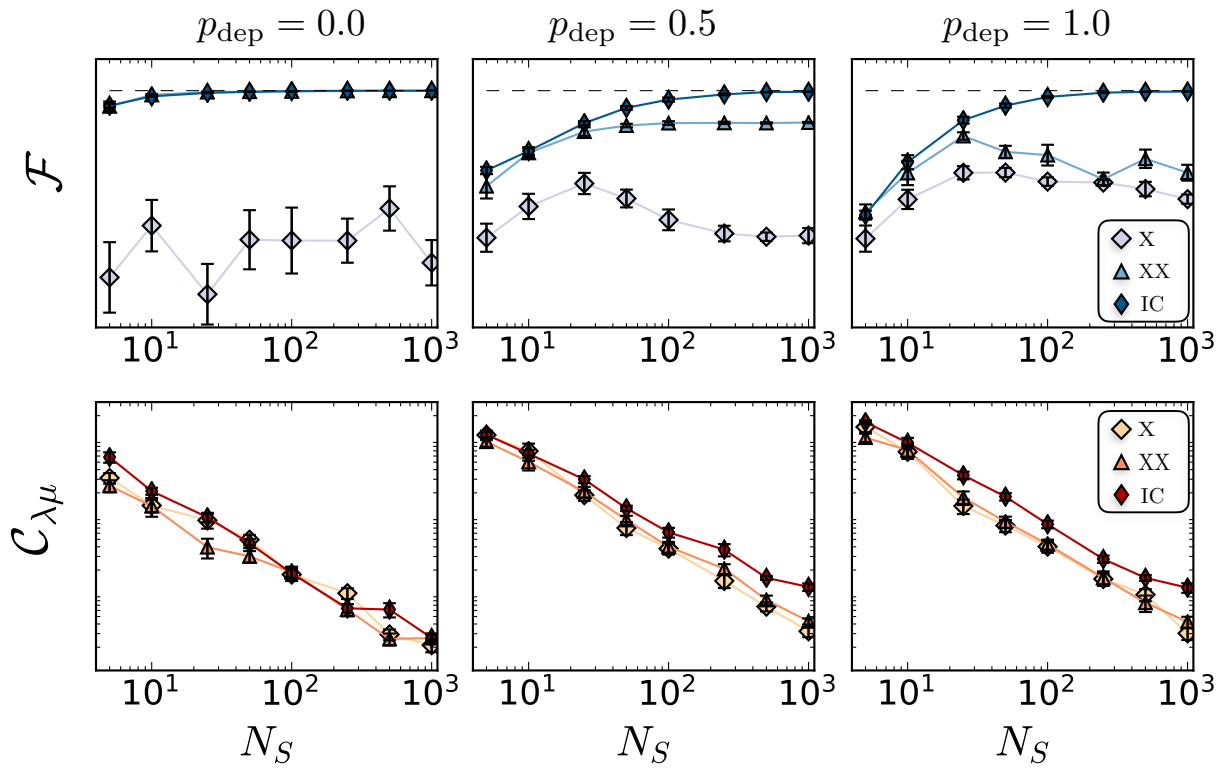

Figure 4.15: **Bases insufficiency**. Errors due to measurement settings: scaling of the fidelity and KL divergence with the number of measurements per basis N_S for different depolarizing strengths $p_{\rm dep}$. We report three different choices of the measurements bases.

on the presence of structure in the state. To assess if the available measurement data is sufficient, we train the NDO using a subset of the training dataset, and systematically increase its size. We establish the number of measurement to be sufficient if convergence in the total KL divergence $\mathcal{C}^{\mathcal{D}}_{\boldsymbol{\lambda\mu}}$ (or equivalently in the negative log-likelihood) computed on the data is achieved. We show such convergence in Fig. 4.14, where we plot the scaling of both the fidelity and the total KL divergence with the number of measurement per basis, for different values of the depolarizing probability $p_{\rm dep}$ (each data point is an averaged over 10 realizations of the training datasets). We observe that the convergence values of N_S increases with the mixing of the target state, and the total KL divergence scales with power law against the N_S.

The other source of error in the training data is the choice of measurements, i.e. number of bases. We can imagine the case where the training algorithm is effective (i.e. the optimization of the KL divergence is successful), but the dataset if not informationally complete, which means that some features of the state are not accessible from the available measurements. In general, the number measurements settings are lower for a pure state than for a mixed state, since the coherences in the density matrix of the former are constrained. The inefficiencies due to lack of measurements settings in the apparatus can be detected within our framework by observing a nearly zero KL divergence, but a sub-optimal fidelity with the true target state. We show this in Fig. 4.15, where we plot the fidelity and KL divergence for different choices of measurement bases (again for different mixing rates). We consider the informationally complete set of Pauli measurements (IC), in addition to the X and the XX measurements sets given by $\left\{(\sigma_0^x, \sigma_1^z), (\sigma_0^z, \sigma_1^x), (\sigma_0^z, \sigma_1^z)\right\}$ and $\left\{(\sigma_0^x, \sigma_1^x), (\sigma_0^x, \sigma_1^z), (\sigma_0^z, \sigma_1^x), (\sigma_0^z, \sigma_1^z)\right\}$ respectively. We can see that the KL divergences identically convergence to zero upon increasing N_S, with no substantial differences between the three bases choices. This means that the optimization procedures always succeed. However, the reconstruction fidelities are different depending on the bases choice, and increase as more measurements settings are included in the dataset, as expected.

A last source of error in the reconstruction is given by *ineffective training*, defined as when the training procedure fails to find the global minimum in the optimization landscape (or equivalently one of the local minima with nearly zero KL divergence). This can be detected simply by observing the KL divergence equilibrating to some non-zero value, and can be dealt with by improving the optimization routines, either with better algorithms or with a better hyper-parameter search.

4.4 Conclusions

We have designed a new framework for the reconstruction of an unknown quantum state from a set of measurements, based on a neural-network representation of the quantum state. For positive pure state, the reconstruction translates into the traditional unsupervised learning procedure, widely use in ML. For quantum states with a non-trivial sign/phase structure and density operators, we require additional measurement to reconstruct the state. The amount and the type of measurements depends on the specific structure of the quantum state [3]. First, we have performed a series of numerical experiments on synthetic data. We have considered paradigmatic states in quantum magnetism and quantum optics, demonstrating that a RBM-based representation of a quantum state, together with standard unsupervised learning, is capable of discovering a faithful representation from a set of measurements for system sizes out of the reach of any other tomographic technique. In contrast, a degrade in performance is observed for structureless, random quantum states.

We have also carried out the neural-network reconstruction on a set of experimental measurements. In the first case, we have implemented a NDO to reconstruct the density matrix of two entangled photons, using the set of coincidence counts reported in the original paper [175]. Further, in collaboration with the California Institute of Technology and Harvard University, a new set of experimental measurements were taken on a Rydberg-atom quantum simulator. We performed the reconstruction of the quantum state of the simulator using these measurements of atomic occupation number. We found perfect agreement between observables calculated from the measurement data and the ones reconstructed by the neural wavefunction. The neural-network representation of the quantum state can now be used to calculate expectation values of observables not accessible in the experimental setup. For instance, we computed the transverse magnetization, and found good agreement with simulation with ED on the model Hamiltonian. This holds when the system is in the paramagnetic phase, with some deviations as the model is brought deeper into the $\mathbb{Z}_2$ phase. Such deviations are attributed to the mixing of the quantum state due to the noise and decoherence during the adiabatic sweep of the detuning, as well as measurements

[3] In the worst case scenario, the full set of 3^N Pauli bases might be required

infidelities. We also calculated the second Renyi entanglement entropy, impossible to extract experimentally. We compared our reconstruction with the exact calculation from ED and found also a good agreement, up to the further part of the phase diagram in the ordered phase. This in turn show that the quantum state prepared with the adiabatic sweep in the experiment is in fact very close to the theoretical model.

The possibility to obtain approximate estimates of important quantities out of the reach of experimental measurements from only a moderate number of measurements, makes neural network an important addition to the toolbox of quantum tomography. On the one hand, the reconstruction allows calculation of otherwise unknown physical observables. This is the case for entanglement entropy, which we showed it can be obtained using only simple measurements of the density, currently accessible with cold atoms [185] setups and adiabatic quantum simulators [186]. Furthermore, the reconstruction can help us to extract informations about the interaction between the quantum system and the environment. This is particularly important in the process of increasing the coherence time of noisy intermediate-scale quantum hardware. Unsupervised learning of quantum data with neural networks will guide us in discovering fingerprints of the noise and decoherence in the hardware, providing invaluable feedback for experiments.

Chapter 5

Neural-network protection of a topological qubit

We have demonstrated that RBMs, equipped with unsupervised learning algorithms, have a great potential for characterizing quantum matter and probing/verifying near-term quantum devices. Most notably, techniques such as QSR will assist the development of the current generation of quantum hardware, by extracting important information about the underlying noise, for example. The mitigation of the noise, for instance through quality increase in qubits manufacturing, will allow more reliable quantum operations to run for a longer time. However, for practical applications of quantum algorithms and for the quantum simulation of large scale materials, the quantum hardware must also be supplied with quantum error correction (QEC), a protocol to recover from the errors and to maintain the quantum coherence in the hardware. The general strategy to achieve such error-correcting regime consist of encoding the logical state of a qubit redundantly, so that errors can be corrected before they corrupt it [187]. A leading candidate for the implementation of such fault-tolerant quantum hardware is the *surface code*, where a logical qubit is stored as a topological state of an array of many *physical* qubits [188]. As the noise corrupts the state of the physical qubits, QEC recovers from the errors before they proliferate and destroy the logical state. The protocols performing this correction are termed "decoders", and must

be implemented by classical algorithms running on conventional computers [189, 190]. In this Chapter, we demonstrate how the RBM can be used to construct a flexible decoder for generic quantum codes within the stabilizer framework [81]. We will not present a general or rigorous formulation of QEC, but instead consider a specific realization of a topological code, the Kitaev toric code.

5.1 The Kitaev toric code

In topological QEC codes, the logical quantum information is stored into global degrees of freedom in such a way that the hardware becomes robust against local perturbations. Most topological codes can be described in terms of the stabilizers formalism [191]. A stabilizer code is a particular class of QEC code characterized by a protected subspace $\mathscr{C}$ defined by an abelian group $\mathscr{S}$, called the *stabilizer group*. More precisely, the set of quantum states $|\psi_j\rangle \in \mathscr{C}$ (called *codeword* states) living in the protected subspace are left invariant by the elements $\hat{S}_k \in \mathscr{S}$ of the stabilizer group, i.e. $\hat{S}_k|\psi_j\rangle = |\psi_j\rangle$. For a stabilizer code with N qubits and m independent stabilizers, the number k of encoded logical qubits is equal to $k = N - m$. Moreover, a stabilizer code is said to be local if all the stabilizers act only on nearby qubits. The simplest example of a topological stabilizer code is the $2d$ toric code, first introduced by Kitaev [192]. For this local QEC code, the quantum information is encoded into the homological degrees of freedom, with topological invariance given by the first homology group [193]. The code features N qubits placed on the links of a $L \times L$ square lattice embedded on a torus. The stabilizers group is $\mathscr{S} = \{\hat{Z}_p, \hat{X}_v\}$, where the plaquette and vertex stabilizers are defined respectively as

$$\hat{Z}_p = \bigotimes_{\ell \in p} \hat{\sigma}^z_\ell \qquad\qquad \hat{X}_v = \bigotimes_{\ell \in v} \hat{\sigma}^x_\ell \,, \tag{5.1}$$

with $\hat{\sigma}^z_\ell$ and $\hat{\sigma}^x_\ell$ acting respectively on the links contained in the plaquette p and the links connected to the vertex v. (Fig. 5.1). All the stabilizers commutes with each other, since they either share no links or an even number thereof. Because of the periodic boundaries

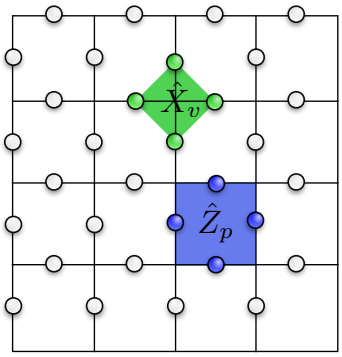

Figure 5.1: A toric code with $L = 4$, for a total of $N = 2L^2 = 32$ qubits. We show in green and blue a vertex and a plaquette stabilizer respectively.

of the lattice, the stabilizers satisfy the condition

$$\bigotimes_p \hat{Z}_p = \bigotimes_v \hat{X}_v = \hat{\mathcal{I}} \,, \tag{5.2}$$

and thus only $2L^2-1$ out of the $2L^2$ stabilizers are independent from each other. Therefore, the total number of encoded logical qubit for the $2d$ the toric code is $k = 2$. The code protected subspace is the Hilbert space spanned by the codeword basis states $|\psi_j\rangle$ that are invariant under stabilizers operations:

$$\mathscr{C} = \left\{ |\psi_j\rangle \;:\; \hat{Z}_p|\psi_j\rangle = \hat{X}_v|\psi_j\rangle = |\psi_j\rangle \;\; \forall v, p \right\} . \tag{5.3}$$

Note that $\mathscr{C}$ corresponds to the ground state manifold of the Hamiltonian

$$\hat{H} = -\sum_p \hat{Z}_p - \sum_v \hat{X}_v \,, \tag{5.4}$$

which contains many highly degenerate ground states, separated by an energy gap from the remaining states in the energy spectrum. The model itself possesses a rich physics, with a loop quantum gas ground state and a simple realization of topological order.

5.1.1 Homology classes

Let us now consider a generic Pauli operator $\hat{P}$, defined as the tensor product of $\hat{\sigma}^z$'s and $\hat{\mathcal{I}}$'s acting on the physical qubits (disregarding $\hat{\sigma}^x$ without loss of generality). The action of $\hat{P}$ on a given code state $|\psi_0\rangle$ can be better understood in term of homology. Recall that a $2d$ lattice is a collection of 0-cells (vertices), 1-cells (links) and 2-cells (plaquette). We can define a mapping that assigns an element from the $\mathbb{Z}_2$ group to each element of a n-cell, defining a n-chain. For instance, a 1-chain $\boldsymbol{c} \in C_1$ can be written as $\boldsymbol{c} = \sum_k z_k l_k$ with $z_k \in \mathbb{Z}_2$ and l_k identifying the links in the lattice. The group of 1-chains C_1 inherits the addition rule from $\mathbb{Z}_2$ ($\boldsymbol{c} \oplus \boldsymbol{c} = 0$) and thus it is itself an abelian group. We can now efficiently write any Pauli operator $\hat{P}$ as a 1-chain $\boldsymbol{c}$

$$\hat{P} = \bigotimes_{k \in \text{Links}} \left(\hat{\sigma}_k^z\right)^{z_k} , \tag{5.5}$$

where $\hat{P}$ acts with $\hat{\sigma}_k^z$ on the qubit on link k if $z_k = 1$ and with the identity $\mathcal{I}_k$ if $z_k = 0$.

Imagine now to initialize the code in the quantum state $|\psi_0\rangle$, chosen between the codewords basis states on the protected subspace, $|\psi_0\rangle \in \mathscr{C}$. The effect of the application of a generic Pauli operator $\hat{P}$ to the state $|\psi_0\rangle$ depends on the boundaries of the 1-chain $\boldsymbol{c}$ corresponding to $\hat{P}$. In general, the boundary operator ∂_n maps n-chains into $(n-1)$-chains:

$$\partial_n : C_n \to C_{n-1} . \tag{5.6}$$

For the case of the 1-chain Pauli operator $\boldsymbol{c}$, we have two possible outcomes. In the first case, the boundary operator is different from zero, $\partial_1 \boldsymbol{c} \neq 0$. The resulting 0-chain is different than zero for a set of vertices, which are the endpoints of the 1-chain $\boldsymbol{c}$ (Fig. 5.2a). This implies that the Pauli operator $\hat{P}$ does not commute with all the stabilizers. From the point of view of the physical Hamiltonian, this is equivalent to creating excitations in the vertices where the stabilizers do not commute with $\hat{P}$. The system being now into an excited states is equivalent for the state $|\psi_0\rangle$ to have left the protected subspace of the code. In the second case, the application of the boundary operator generates a chain with no boundaries (Fig. 5.2b-c). The chain $\boldsymbol{c}$, also called a *cycle*, belongs to the sub-group

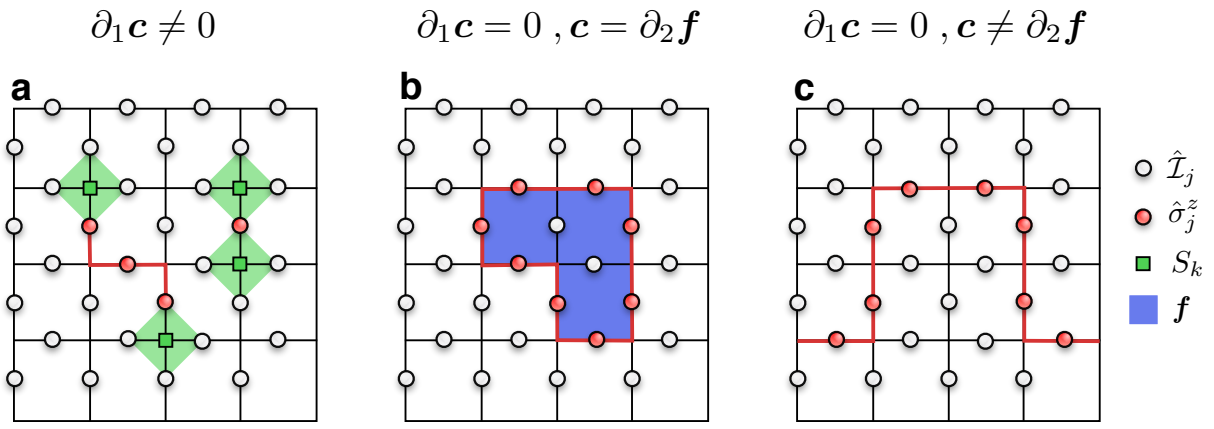

Figure 5.2: **Pauli operators and boundaries**. Different 1-chains corresponding to Pauli operators acting on the state of the code. The qubits are acted upon by the identity (white) or the $\hat{\sigma}^z$ (red) operator. The non-trivial stabilizer measurements are green squares. **a**) A chain with a non-zero boundary, creating an even number of defects at the endpoints of the error strings. **b**) A chain with no boundary (cycle), with a trivial homology class (i.e. enclosing a finite number of plaquette). **c**) A cycle with non-trivial homology class (i.e. a non-contractible loop in the torus).

$\boldsymbol{c} \in \mathcal{Z}_1 \subset C_1$. In this case, since the cycle has no endpoints ($\partial_1 \boldsymbol{c} = 0$), the corresponding operator $\hat{P}$ always shares either zero or an even number of links with all the stabilizers in $\mathscr{S}$, which implies

$$[\hat{P}, \hat{Z}_p] = [\hat{P}, \hat{X}_v] = 0\ . \tag{5.7}$$

Therefore, the quantum state resulting from the application of the Pauli operator is contained in the code protected subspace, $|\psi'\rangle = \hat{P}|\psi_0\rangle \in \mathscr{C}$.

We assume again a Pauli operator $\hat{P}$ applied to the code state $|\psi_0\rangle \in \mathscr{C}$, and described by a cycle $\boldsymbol{c}$. The set of cycles in the lattice can be further broken down into sub-classes by considering their homology. We define a sub-group $\mathcal{B}_1 \subset \mathcal{Z}_1$ of cycles which are boundaries of 2-chains, that is:

$$\mathcal{B}_1 = \left\{ \boldsymbol{c} \in C_1\ :\ \partial_1 \boldsymbol{c} = 0 \text{ and } \exists \boldsymbol{f} \in C_2\ \text{ s.t. }\ \boldsymbol{c} = \partial_2 \boldsymbol{f} \right\}. \tag{5.8}$$

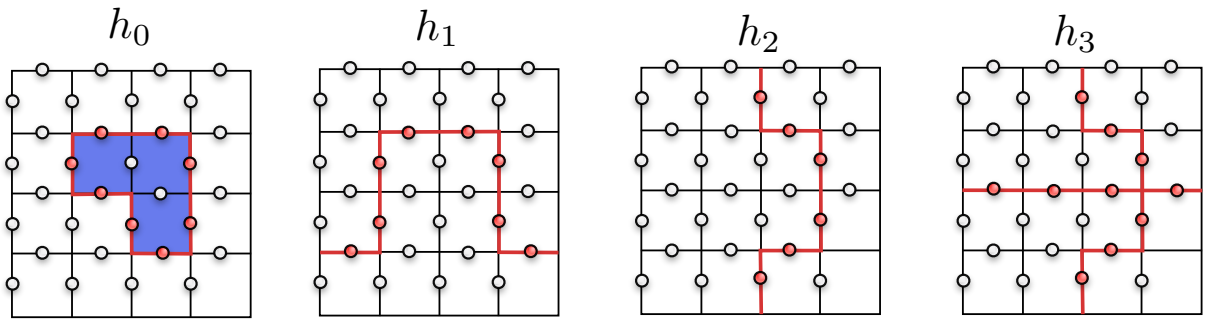

Figure 5.3: **Homology classes**. The four homology classes for Pauli operators containing only identities and $\hat{\sigma}^z$'s. Aside the trivial class, the other classes corresponds to either one of the two different non-contractible loops in the torus (h_1 and h_2), or both loops (h_3).

The boundary sub-group $\mathcal{B}_1$ generates a partition of the cycles group $\mathcal{Z}_1$ into equivalence classes. This defines the first homology group as the quotient group $\mathcal{H}_1 = \mathcal{Z}_1/\mathcal{B}_1$ whose elements are called *homology classes*. Further, the definition of $\mathcal{H}_1$ introduce the notion of homological equivalence: two 1-chains $\boldsymbol{c}$ and $\boldsymbol{c}'$ are homologically equivalent if $\boldsymbol{c}' = \boldsymbol{c} \oplus \boldsymbol{d}$ where $\boldsymbol{d} \in \mathcal{B}_1$. In other words, the sub-group $\mathcal{B}_1$ contains the set of cycles that encloses a well defined number of plaquette in the lattice. In this case, we say that the cycle $\boldsymbol{c}$ has *trivial homology*, that is, the corresponding Pauli operator can be re-written as a product of plaquette stabilizers $\hat{Z}_p$, and thus it is contained in the stabilizer group $\hat{P} \in \mathscr{S}$. The code state $|\psi_0\rangle$ is then left invariant from the application of the Pauli operator, $\hat{P}|\psi_0\rangle = |\psi_0\rangle$ (Fig. 5.2b). On the other hand, if no 2-chain $\boldsymbol{f}$ exists such that the cycle $\boldsymbol{c}$ is the boundary of $\boldsymbol{f}$ ($\boldsymbol{c} = \partial_2 \boldsymbol{f}$), then $\boldsymbol{c}$ belongs to a non-trivial homology class and the corresponding operator $\hat{P}$, even though it commutes with all the stabilizers, acts non-trivially on the code subspace (Fig. 5.2c). If we only consider Pauli operators built with identities and $\hat{\sigma}^z$ Pauli matrices, there exists four different homology classes, corresponding to combination of the two non-contractible loops in the real lattice on the torus (Fig. 5.3).

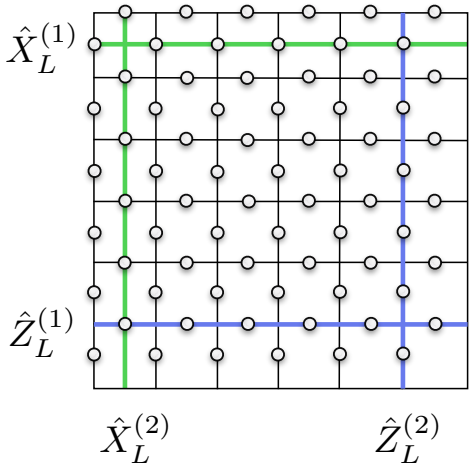

Figure 5.4: **Logical operators**. The four logical operators for the two logical qubits.

The topological qubit

The topological invariance given by the first homology group provides a way to encode the logical quantum information into the protected subspace of the code. In fact, given two cycles from two different homology classes, the only operation that can transform one cycle into the other are global operations on the code. As such, if we store the quantum information into the homology class, local perturbations will leave the logical quantum information untouched. As such, we can defined the logical zero state $|0_L\rangle$ using the trivial homology class, for example. In practice, there are many quantum states of the physical qubits that corresponds to $|0_L\rangle$, and they are all homologically equivalent. We now need a mean to manipulate this logical quantum information, i.e. logical operators. Given the commutation rules of the stabilizers, and given that the logical-Z and logical-X must anti-commute with each other, the natural definition of the logical operators are the non-contractible loops in the torus (Fig. 5.4). More precisely, $\hat{Z}_L^{(i)}$ are the two non-trivial cycles in the real lattice and $\hat{X}_L^{(i)}$ are the two non-trivial cycles in the dual lattice. The index $i = 1, 2$ identifies here logical qubit. It is clear now that such logical operations, once applied to the quantum state of the code, they change the homology class and thus they manipulate the logical quantum information in a non-trivial way.

5.1.2 Error correction

We consider now the resilience of the topological qubit against noise, and examine how QEC can be used to maintain the quantum coherence in the code. The effect of noise is described by a trace preserving completely positive map $\mathscr{E}$ which is applied to the quantum state of the code $\hat{\rho} = |\psi_0\rangle\langle\psi_0|$, which belongs to the protected subspace, (e.g. trivial homology). Here, we consider the simple case of a dephasing channel, where each qubits in the code independently undergoes the channel

$$\hat{\rho}_k \longrightarrow (1 - p_{err})\hat{\rho}_k + p_{err}\, \hat{\sigma}_k^z \hat{\rho}\, \hat{\sigma}_k^z \,. \tag{5.9}$$

In practice, we can regard this channel as a Pauli (error) operator $\hat{E}$, described by a 1-chain $\boldsymbol{e}$, where $e_k = 1$ with probability p_{err} ($\hat{\sigma}^z$ error) and $e_k = 0$ with probability $1 - p_{err}$ (no error). Since the error operator $\hat{E}$ does not commute with a subset of vertex stabilizers $\hat{X}_v$, the application of the noise channel maps the quantum state of the toric code out of the protected subspace. We wish now to design a procedure to recover from the error operator $\hat{E}$ and restore the quantum state in such a way that the homology class is left unchanged. Clearly we cannot measure each single qubit separately without destroying the quantum coherence, but we can instead measure all the stabilizer operators without disturbing the quantum state of the code. Because $[\hat{E}, \hat{Z}_p] = 0\ \forall p$, we can extract information on $\hat{E}$ by measuring the vertex stabilizers only, whose eigenvalues will be equal to -1 on a vertex v if $\hat{X}_v$ shares an odd number of links with the error operator. Unless $\boldsymbol{c}$ is a cycle (case in which the eigenvalues of $\hat{X}_v$ are all equal to unity and the error is then undetectable), the stabilizers measurement provides the locations of the endpoints of the error strings present in $\boldsymbol{c}$, also called the *syndrome* $\boldsymbol{S}(\boldsymbol{e}) = \partial_1 \boldsymbol{e}$ (green squares in Fig. 5.5a).

Given an error operator $\hat{E}$ and its syndrome $\boldsymbol{S}(\boldsymbol{e})$, the first step of the recovery is to map the code state back into the protected subspace. This translates into applying a recovery operator $\hat{R}$ (with 1-chain $\boldsymbol{r}$), such that $\boldsymbol{S}(\boldsymbol{e}) = \boldsymbol{S}(\boldsymbol{r})$. This effectively change all the stabilizer measurements from non-trivial (-1) to trivial (+1). From a different perspective, where non-trivial measurements are quasi-particle excitations, the recovery consists of annihilating pairwise all the quasi-particles. While this brings back the state

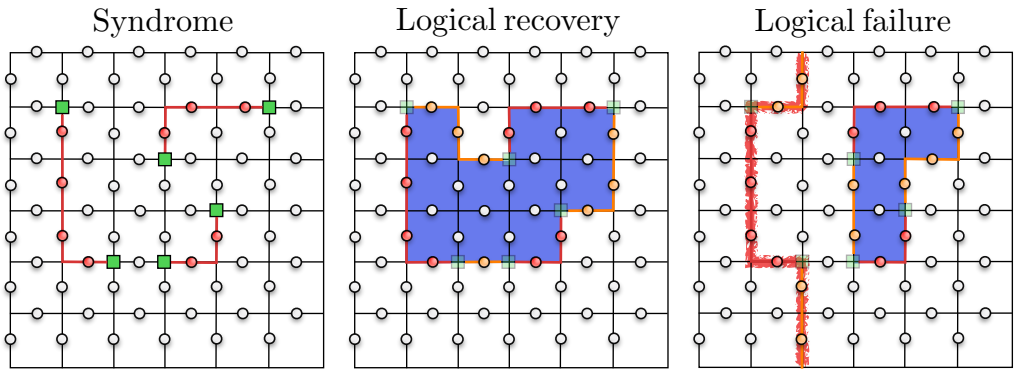

Figure 5.5: **Error correction**. **a**) An error chain $\boldsymbol{e}$ (red circles) and its syndrome $\boldsymbol{S}(\boldsymbol{e})$ (green squares). **b**) A successful recover chain (orange circles), creating a trivial cycle. **c**) A recovery chain generating a non-trivial cycle, and thus leading to a logical failure. This corresponds to apply the logical operators $\hat{Z}_L^{(2)}$.

into the code subspace, it does not imply that the logical quantum information has been restored. In fact, the success of the error correction depends on the homological equivalence between $\boldsymbol{e}$ and $\boldsymbol{r}$. If the two chains are homologically equivalent, the cycle given by the combined error and recovery operation is trivial, $\boldsymbol{e} \oplus \boldsymbol{r} \in \mathcal{B}_1$ and $\hat{R}\hat{E} \in \mathscr{S}$, leading to a successful recovery (Fig. 5.5b). If however the cycle $\boldsymbol{e} \oplus \boldsymbol{r}$ has non-trivial homology, the combined operator can be transformed into one of the logical operators and thus it directly manipulates the encoded logical information, leading to a logical failure (Fig. 5.5c).

Given some physical error chain $\boldsymbol{e}$, the non-trivial task of choosing the recovery chain $\boldsymbol{r}$ is call *decoding*. Several decoders have been proposed for the $2d$ toric code, based on different strategies [194, 195, 196, 197, 198]. Maximum likelihood decoding consists of finding a recovery chain $\boldsymbol{r}$ with the most likely homology class [199, 200]. A different recovery strategy, designed to reduce computational complexity, consists of generating the recovery chain $\boldsymbol{r}$ compatible with the syndrome simply by using the minimum number of errors. Such a procedure, called Minimum Weight Perfect Matching (MWPM) [201], has the advantage that can be performed without the knowledge of the error probability p_{err}. This algorithm is however sub-optimal (with lower threshold probability [193]) since it does not take into account the high degeneracy of the error chains given a fixed syndrome.

5.2 Learning error correction via pattern completion

Let us now reformulate once more the problem at hand. Given a particular (unknown) error chain $\boldsymbol{e}$, the stabilizer measurements provide us with the syndrome $\boldsymbol{S}(\boldsymbol{e})$, that is the collection of the endpoints of the error strings in the lattice. Given this set of vertices, we wish to find a recovery chain $\boldsymbol{r}$ sharing the same syndrome, and restoring the code state in the correct homology class. Therefore, we are interested in discovering the mapping from syndromes to the recovery chains that are homologically equivalent to the unknown error chain. Such mapping will depends on many factors, such as the geometry of the code, the type of noise channel, and possibly the unknown correlations between the physical qubits, as well as the correlations in the noise. In general, one should incorporate all the details when designing the decoder for a particular code realization. Once again, this problem reminds us the hand-crafting of the features required in the image classification example discussed in Chapter 1. Along the same lines, we propose to approach this problem from a data-driven perspective, and employ a RBM to first learn the relationship between syndromes and errors, and then to run as a decoder and carry out the error correction.

In this setup, one data sample for the training is made up by an error chain $\boldsymbol{e}$ and its syndrome $\boldsymbol{S}$. The data can be easily obtained in a real-world implementation of the code, simply by initializing the code to a reference state, measure the syndrome and then each physical qubits. Clearly this destroys the quantum coherence but, if repeated many times, it generates a dataset $\mathcal{D} = \{\boldsymbol{e}, \boldsymbol{S}\}$, implicitly containing the probabilistic relation between the errors and the syndromes. We aim to train the RBM to learn the joint distribution $q(\boldsymbol{e}, \boldsymbol{S})$ underlying the dataset. Once trained, given the partial information $\boldsymbol{S}$, we can perform the correction by generating a recovery chain $\boldsymbol{r}$ from the conditional distribution of errors given the syndrome. In other words, the decoding reduces to a pattern completion problem. Given the partial pattern (syndrome), the trained RBM is sampled to complete the missing part (the error). Given the fact that RBMs were invented to solve pattern completion problems (amongs other things), it feels particularly natural to employ a RBM to perform the error correction.

5.2.1 The neural decoder

We construct the neural decoder using a regular RBM, which is re-arranged in Fig. 5.6 for clarity. There is still one hidden layer $\boldsymbol{h} \in \{0,1\}^{n_h}$, and one visible layer, which is however split into a syndrome layer $\boldsymbol{S} \in \{0,1\}^{N/2}$ and an error layer $\boldsymbol{e} \in \{0,1\}^{N}$. The full RBM distribution is given by

$$p_{\boldsymbol{\lambda}}(\boldsymbol{e},\boldsymbol{S},\boldsymbol{h}) = Z_{\boldsymbol{\lambda}}^{-1} \mathrm{e}^{-E_{\boldsymbol{\lambda}}(\boldsymbol{e},\boldsymbol{S},\boldsymbol{h})} \,, \tag{5.10}$$

where $\boldsymbol{\lambda} = \{\boldsymbol{U},\boldsymbol{W},\boldsymbol{b},\boldsymbol{c},\boldsymbol{d}\}$ are the network parameters,

$$E_{\boldsymbol{\lambda}}(\boldsymbol{e},\boldsymbol{S},\boldsymbol{h}) = -\sum_{ik} U_{ik} h_i S_k - \sum_{ij} W_{ij} h_i e_j - \sum_j b_j e_j - \sum_i c_i h_i - \sum_k d_k S_k \tag{5.11}$$

is the energy of the model and

$$Z_{\boldsymbol{\lambda}} = \mathrm{Tr}_{\{\boldsymbol{h},\boldsymbol{S},\boldsymbol{e}\}}\ \mathrm{e}^{-E_{\boldsymbol{\lambda}}(\boldsymbol{e},\boldsymbol{S},\boldsymbol{h})} \tag{5.12}$$

is the partition function. The joint probability distribution over the visible layer $(\boldsymbol{e},\boldsymbol{S})$ is obtained analogously to the derivation in Chapter 2, by integrating out the hidden variables from the full distribution

$$p_{\boldsymbol{\lambda}}(\boldsymbol{e},\boldsymbol{S}) = \sum_{\boldsymbol{h}} p_{\boldsymbol{\lambda}}(\boldsymbol{e},\boldsymbol{S},\boldsymbol{h}) = \frac{1}{Z_{\boldsymbol{\lambda}}} \mathrm{e}^{-\mathcal{E}_{\boldsymbol{\lambda}}(\boldsymbol{e},\boldsymbol{S})} \,, \tag{5.13}$$

with the effective visible energy

$$\mathcal{E}_{\boldsymbol{\lambda}}(\boldsymbol{e},\boldsymbol{S}) = -\sum_j b_j e_j - \sum_k d_k S_k - \sum_i \log\left(1 + \mathrm{e}^{c_i + \sum_k U_{ik} S_k + \sum_j W_{ij} e_j}\right) \,. \tag{5.14}$$

We train the RBM using generative modelling, by minimizing the average KL divergence on the dataset:

$$\mathcal{C}_{\boldsymbol{\lambda}} = -\|\mathcal{D}\|^{-1} \sum_{\{\boldsymbol{e},\boldsymbol{S}\}} \log p_{\boldsymbol{\lambda}}(\boldsymbol{e},\boldsymbol{S}) \,, \tag{5.15}$$

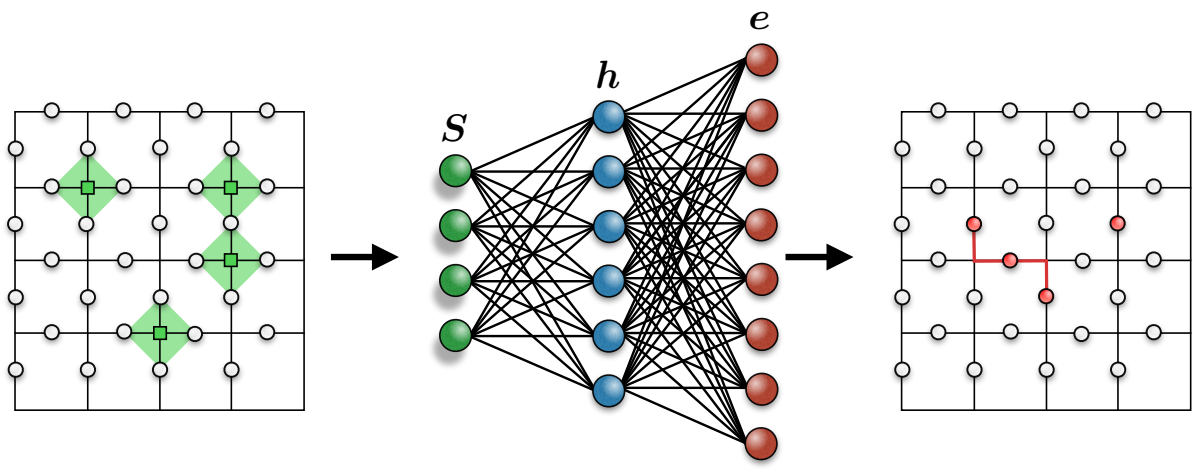

Figure 5.6: **The neural decoder**. The hidden layer $\boldsymbol{h}$ is fully-connected to the syndrome and error layers $\boldsymbol{S}$ and $\boldsymbol{e}$ After training, the syndrome is fed to the neural decoder, which generates a compatible error recovery chain.

where again we omit the constant data entropy term. As before, the gradient of the cost function reduces to the gradient of the log probability

$$\nabla_{\boldsymbol{\lambda}_j} \log p_{\boldsymbol{\lambda}}(\boldsymbol{e}, \boldsymbol{S}) = -\nabla_{\boldsymbol{\lambda}_j} \mathcal{E}_{\boldsymbol{\lambda}}(\boldsymbol{e}, \boldsymbol{S}) + \sum_{\boldsymbol{e},\boldsymbol{S}} \log p_{\boldsymbol{\lambda}}(\boldsymbol{e}, \boldsymbol{S}) \nabla_{\boldsymbol{\lambda}_j} \mathcal{E}_{\boldsymbol{\lambda}}(\boldsymbol{e}, \boldsymbol{S}) \,, \tag{5.16}$$

which contains the positive (easy) phase and the negative phase, to be calculated using block Gibbs sampling. The training algorithm is exactly equivalent to the one seen in Chapter 2 for a regular RBM, where the only difference is the distinction between errors and syndrome units within the visible layer.

Decoding

We now discuss the decoding algorithm, which proceeds assuming that we successfully learned the distribution $p_{\boldsymbol{\lambda}}(\boldsymbol{e}, \boldsymbol{S})$. Given an error chain $\boldsymbol{e}_0$ with syndrome $\boldsymbol{S}_0$ we wish to use the RBM to generate an error chain compatible with $\boldsymbol{S}_0$ to use for the recovery. To achieve this goal we separately train networks on different datasets obtained from

different error regimes p_{err}. Assuming we know the error regime that generated $\boldsymbol{e}_0$, the QEC procedure consists of sampling a recovery chain from the distribution $p_{\boldsymbol{\lambda}}(\boldsymbol{e} \,|\, \boldsymbol{S}_0)$ given by the network trained at the same probability p_{err} of $\boldsymbol{e}_0$. Although the RBM does not learn this distribution directly, by sampling the error and hidden layers while keeping the syndrome layer fixed to $\boldsymbol{S}_0$, since $p_{\boldsymbol{\lambda}}(\boldsymbol{e}, \boldsymbol{S}_0) = p_{\boldsymbol{\lambda}}(\boldsymbol{e} \,|\, \boldsymbol{S}_0)p(\boldsymbol{S}_0)$, we are enforcing sampling from the desired conditional distribution. An advantage of this procedure over decoders that employ conventional MC [196, 197] on specific stabilizer codes is that specialized sampling algorithms tied to the stabilizer structure, or multi-canonical methods such as parallel tempering, are not required. Finally, note that the assumption of perfect learning is not critical, since the above sampling routine can be modified with an extra rejection step as discussed in Ref. [202] to ensure sampling occurs from the proper physical distribution.

An error correction procedure can be defined as follows: we first initialize the machine into a random state of the error and hidden layers and to $\boldsymbol{S}_0$ for the syndrome layer. We then let the machine equilibrate by repeatedly performing block Gibbs sampling. After some amount of equilibration steps, we begin checking the syndrome of the error layer $\boldsymbol{e}$ in the machine and, as soon as $\boldsymbol{S}(\boldsymbol{e}) = \boldsymbol{S}_0$ we select the error as recovery chain $\boldsymbol{r}$. If such a condition is not met before a fixed amount of sampling steps, the recovery attempt is stopped and considered failed. This condition makes the precise computational requirements of the algorithm ill-defined, since the cut-off time can always be increased resulting in better performance for a higher computational cost. Once we have found a compatible recovery chain $\boldsymbol{r}$, we can check the results of the correction by computing the Wilson loops

$$w(\gamma) = -\prod_{k \in \gamma^*} \big(2(\boldsymbol{e}_0 \oplus \boldsymbol{r})_k - 1\big) \tag{5.17}$$

where γ^* are non-contractible loops in the dual lattice. Since any cycle with trivial homology crosses $w(\gamma)$ an even number of times, we find that, if L is even, $w(\gamma) = -1$ (for any γ) if the cycle $\boldsymbol{e}_0 \oplus \boldsymbol{r}$ has non-trivial homology, signature of a logical failure.

5.2.2 Numerical simulations

We trained the RBMs in different error regimes by building several datasets $\mathcal{D}_p = \{\boldsymbol{e}_k, \boldsymbol{S}_k\}$ at elementary error probabilities $p = \{0.5, 0.6, \ldots, 0.15\}$ of the dephasing channel. For a given error probability, the network hyper-parameters are individually optimized via a grid search. We also note that, in contrast with previous simulations, we have observed a more severe overfitting, i.e. the network reproducing very well the distribution contained in the training dataset but being unable to properly generalize the learned features. To avoid this common issue in neural networks, we employ weight-decay regularization [122], by adding an extra penalty term to the KL divergence, proportional to the square weights times a coefficient $l_2 \simeq 10^{-5}$. Once training is complete, to test the performance we generate a test set $\mathcal{T}_p = \{\boldsymbol{e}_k\}$ and for each error chain $\boldsymbol{e}_k \in \mathcal{T}_p$, after a suitable equilibration time (usually $N_{eq} \propto 10^2$ sampling steps), we collect the first error chain $\boldsymbol{e}$ compatible with the original syndrome, $\boldsymbol{S}(\boldsymbol{e}) = \boldsymbol{S}(\boldsymbol{e}_k)$. We use this error chain for the recovery, $\boldsymbol{r}^{(k)} = \boldsymbol{e}$. Importantly, error recovery with $\boldsymbol{r}^{(k)}$ chosen from the first compatible chain means that the cycle $\boldsymbol{e}_k + \boldsymbol{r}^{(k)}$ is sampled from a distribution that includes all homology classes. By computing the Wilson loops on the cycles we can measure their homology class. This allows us to gauge the accuracy of the decoder in term of the logical failure probability, defined as $P_{fail} = \frac{n_{\text{fail}}}{\|\mathcal{T}_p\|}$, where n_{fail} is the number of cycles with non-trivial homology. Because of the fully-connected architecture of the network, and the large complexity of the probability distribution arising from the high degeneracy of error chains given a syndrome, we found that the dataset size required to accurately capture the underlying statistics must be relatively large ($\|\mathcal{D}_p\| \propto 10^5$). In Fig. 5.7 we plot the logical failure probability P_{fail} as a function of the elementary error probability for the neural decoding scheme. We note that at low p_{err}, our logical failure probabilities follow the expected [203] scaling form $p_{err}^{L/2}$ (not shown in the figure).

To compare our numerical results we also perform error correction using the recovery scheme given by MWPM [204]. This algorithm creates a graph whose vertices corresponds to the syndrome and the edges connect each vertex with a weight equal to the Manhattan distance (the number of links connecting the vertices in the original square lattice). MWPM

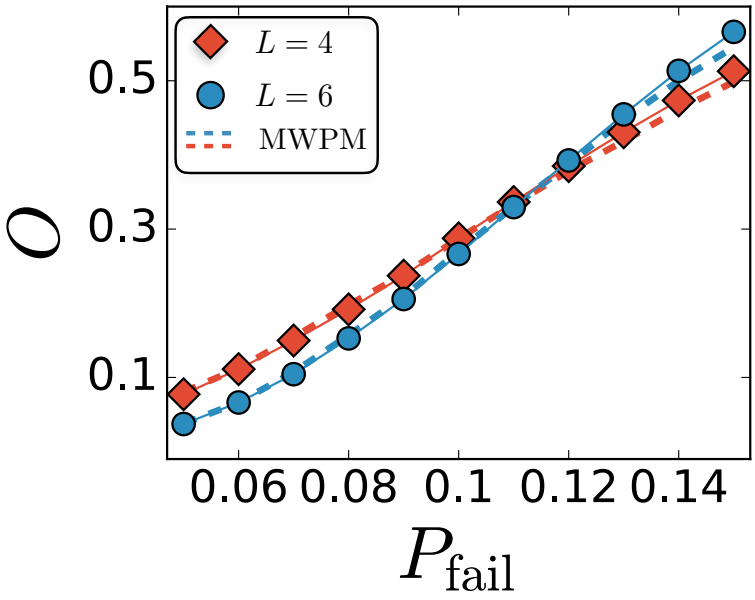

Figure 5.7: **Probabilistic decoding**. Logical failure probability as a function of elementary error probability for MWPM (lines) and the neural decoder (markers).

then finds an optimal matching of all the vertices pairwise using the minimum weight, which corresponds to the minimum number of edges in the lattice [205]. Fig. 5.7 displays the comparison between a MWPM decoder (line) and our neural decoder (markers). As is evident, the neural decoder has an almost identical logical failure rate for error probabilities below the threshold ($p_{err} \approx 10.9$ [193]), yet a significant higher probability above. Note that by training the RBM on different datasets we have enforced in the neural decoder a dependence on the error probability. This is in contrast to MWPM which is performed without such knowledge. Another key difference is that the distributions learned by the RBM contain the entropic contribution from the high degeneracy of error chains, which is directly encoded into the datasets. It will be instructive to explore this further, to determine whether the differences in Fig. 5.7 come from inefficiencies in the training, the different decoding model of the neural network, or both. Finite-size scaling on larger L will allow calculation of the threshold defined by the neural decoder.

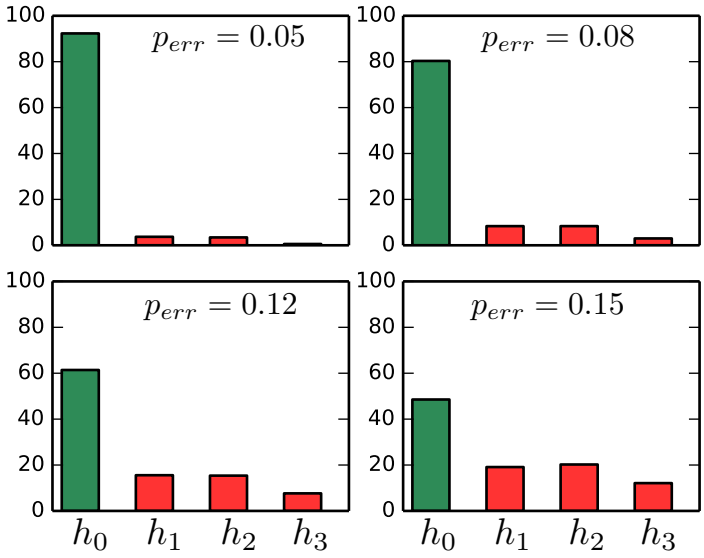

Figure 5.8: **Histogram of the homology classes**. The green bars represent the trivial homology class h_0 corresponding to contractible loops on the torus. The other three classes correspond respectively to the logical operations $\hat{Z}_L^{(1)}$, $\hat{Z}_L^{(2)}$ and $\hat{Z}_L^{(1)}\hat{Z}_L^{(2)}$.

In the above algorithm, which amounts to a simple and practical implementation of the neural decoder, our choice to use the first compatible chain for error correction means that the resulting logical operation is sampled from a distribution that includes all homology classes. This is illustrated in Fig. 5.8, where we plot the histogram of the homology classes for several different elementary error probabilities. Accordingly, our neural decoder can easily be modified to perform maximum-likelihood optimal decoding. For a given syndrome, instead of obtaining only one error chain to use in decoding, one could sample many error chains and build up the histogram of homology classes with respect to any reference error state. Then, choosing the recovery chain from the largest histogram bin will implement, by definition, maximum-likelihood decoding. Although the computational cost of this procedure will clearly be expensive using the current fully-connected restricted RBM, it would be interesting to explore specializations of the neural network architecture in the future to see how its performance may compare to other maximum-likelihood decoding algorithms [199].

5.3 Conclusions

We have presented a decoder for topological codes using a simple algorithm implemented with a RBM, relying on the efficient sampling of error chains distributed over all homology classes. Numerical results show that our decoder has a logical failure probability that is close to MWPM, but not identical, a consequence of our neural network being trained separately at different elementary error probabilities. This leads to the natural question of the relationship between the neural decoder and optimal decoding, which could be explored further by a variation of our algorithm that implements maximum likelihood decoding.

In its current implementation, the RBM is restricted within a given layer of neurons, but fully-connected between layers. This means that the decoder does not depend on the specific geometry used to implement the code, nor on the structure of the stabilizer group; it is trained simply using a raw data input vector, with no information on locality or dimension. Such a high degree of generalizability, which is one of the core advantages of this decoder, also represents a challenge for investigating bigger systems. For example, a bottleneck in our scheme to decode larger sizes is finding an error chain compatible with the syndrome within a reasonable cut-off time.

In order to scale up the system size on the $2d$ toric code, one could relax some of the general fully-connected structure of the network, and specialize it to accommodate the specific details of the code. Geometric specialization such as this has been explicitly demonstrated to improve the representational efficiency of neural networks in the case of the toric code [206, 207]. Note that, even with moderate specialization, the neural decoder as we have presented above can immediately be extended to other choices of error models [208], such as the more realistic case of imperfect syndrome measurement [209], or transferred to other topological stabilizer codes, such as color codes [210, 211]. We also point out that the training of the networks are performed off-line and have to be carried out only once. As such, the high computational cost of the training need not be considered when evaluating the decoder computational efficiency for any of these examples. After our initial proposal, new decoders based on ML have been proposed, with performance improvements and a broader range of application [82, 83, 84, 85].

Chapter 6

Conclusions

The field of artificial intelligence has seen a tremendous advancement in the years following 2010. In particular, ML algorithms are now widely used to solve the most disparate problems in modern society. The approach generally relies on neural networks trained on available data[1]. While the theoretical framework underlying neural networks has been around for many decades, the high performance required to allow deployment in real world applications has been achieved only recently. This fact is thanks to a new breed of algorithms and the increase in computational power, which allows for the practical training of large deep neural networks [213]. The solution to various long-standing problems in data science [66, 67, 68] further increased the boost in popularity of ML, which is now being researched and applied in many different scenarios. At the core of this powerful techniques is the ability to build a hierarchy of levels of abstractions of complex objects [64], which relies on discovering low-dimensional representations within high-dimensional datasets [214].

Among the multitude of network configurations and learning procedures, we have focussed our attention to a stochastic neural network called a restricted Boltzmann machine. This generative model, invented in the early 80s, became a fundamental tool for pre-training layers of deep neural networks [215, 216]. More recently however, this type of unsupervised pre-training was surpassed in terms of performance by competing methods (such as

[1] There exist also ML algorithms for autonomous learning without data, e.g. *reinforcement learning* [212].

different network connectivity and activation functions), also due to the difficult evaluation of RBMs caused by the presence of the unknown partition function. Consequently, RBMs were generally dismissed in practical applications of deep learning, in favour of more "easy-to-train" networks, capable of reaching higher performances with less computational burden. Nevertheless, it is likely that ML community might undertake a more deep investigation of this powerful generative model. This was highlighted by Yoshua Bengio, a major figure in the ML community, during the 2018 ICML workshop on Theoretical Foundations and Applications of Deep Generative Models: "*GANs* [generative adversarial networks] *are great, but we should go back to exploring more widely the landscape of generative models, such as ... Boltzmann Machines.*" [217]. In turn, to us physicists, the very nature of the RBM, i.e. a classical spin system at thermal equilibrium, suggests that this neural network can be appropriately implemented to capture the features of physical states of matter.

In this Thesis, we revived the RBM and applied it to different problems in quantum many-body physics. Embracing a data-driven approach, we defined a class of neural-network quantum states based on the relationship between the generative properties of the RBM and the probabilistic nature of the measurement process in quantum mechanics. For pure and positive quantum states, we have shown that a regular RBM is capable of parametrizing the quantum state. In the presence of a complex quantum state, an additional RBM is necessary to capture the phase structure. Finally, for the case of mixed states, a neural-network density operator can be defined by purifying two RBMs with a set of auxiliary units embedded in the latent space of the machines.

After having defined pure and mixed neural-network states, we introduced a new framework, based on unsupervised learning, for QSR of unknown quantum states given a set of measurement data [143]. We argued that this problem is the natural application of generative models, and gave a demonstration by reconstructing a variety of quantum states from synthetic data generated with classical simulations, as well as experimental data generated by measurements on quantum hardware in laboratories [144, 145]. Once properly trained, the RBM is capable of generating observables which might be hard to extract both from classical simulations and experimental measurements on quantum system engineered in the labs. One important example is entanglement entropy which plays a very important

role in the study of condensed matter systems. We have shown that replicated RBMs can be used to sample the "Swap" operator and obtain an approximate estimate of the second Renyi entropy, with profound consequences for both computational and experimental studies of strongly-correlated materials. From the experimental perspective, the second Renyi entropy has been measured in experiment [167], though is restricted to very small system. Instead, the entanglement can be reconstructed by first training a RBM on simple density measurements (currently available in various experimental setups), and then sampling the Swap operator in the RBM framework. We have demonstrated this approach for experimental data measured on a Rydberg-atom quantum simulator [145], and we predict its use for quantum computers, adiabatic quantum simulators and bosonic ultra-cold atom experiments. Regarding the classical simulation of quantum materials, the QMC realizations of the replica trick, regardless of the specific realization (e.g. path integrals, stochastic series expansion, etc) can often be difficult to implement in practice. Instead, we can imagine running the QMC simulation for generating samples (without measuring) and train a RBM to learn the wavefunction and use it to calculate the entanglement entropy.

The reconstruction of quantum states from measurements goes beyond extracting physical properties. RBMs can be also implemented to probe quantum devices and shed light on underlying noise processes, providing a powerful tool for increasing the quality of the current generation of quantum hardware. This form of analysis is just the tip of the spear. Soon, we will witness an increasing usage of ML in experimental setups, which will create a feedback loop to refine and improve current experiments. A recent example was the ML reduction of measurements errors on an ion-trap quantum computer [218]. Past that, we envision a full integration between neural networks and experiments, for instance in the context of quantum control to mitigate decoherence [219, 220]. Finally, integration of neural networks and quantum hardware seems the obvious choice for the real-world implementation of decoding schemes on future topologically fault-tolerant hardware. Along this line, we have proposed the first neural-network decoder for stabilizer codes [81], which does not depend on the specific model of the QEC code and it is trained using only raw data. This high degree of generalizability will be crucial to accommodate the different implementations of fault-tolerant hardware, such as its geometry, dimensionality and connectivity.

To conclude, as ML techniques continue to become integrated into the field of condensed matter physics, quantum information science and technology, we anticipate their role in quantum state and process tomography, error correction, quantum control and other tasks in validation will rapidly increase. RBMs offer a powerful method for generative modelling, with training algorithms that are well studied by the ML community. Their demonstrated ability to provide practical tradeoffs between representation, computation, and statistics, offers a rich field of study of quantum states, which will be important in the integration of classical and quantum algorithms inevitable in near-term devices and computers.

References

[1] P. W. Anderson. More is different. *Science*, 177(4047):393–396, 1972.

[2] Immanuel Bloch. Ultracold quantum gases in optical lattices. *Nature Physics*, 1:23 EP –, 10 2005.

[3] Bevin Huang, Genevieve Clark, Efrén Navarro-Moratalla, Dahlia R. Klein, Ran Cheng, Kyle L. Seyler, Ding Zhong, Emma Schmidgall, Michael A. McGuire, David H. Cobden, Wang Yao, Di Xiao, Pablo Jarillo-Herrero, and Xiaodong Xu. Layer-dependent ferromagnetism in a van der waals crystal down to the monolayer limit. *Nature*, 546:270 EP –, 06 2017.

[4] Cheng Gong, Lin Li, Zhenglu Li, Huiwen Ji, Alex Stern, Yang Xia, Ting Cao, Wei Bao, Chenzhe Wang, Yuan Wang, Z. Q. Qiu, R. J. Cava, Steven G. Louie, Jing Xia, and Xiang Zhang. Discovery of intrinsic ferromagnetism in two-dimensional van der waals crystals. *Nature*, 546:265 EP –, 04 2017.

[5] Markus Greiner, Olaf Mandel, Tilman Esslinger, Theodor W. Hänsch, and Immanuel Bloch. Quantum phase transition from a superfluid to a mott insulator in a gas of ultracold atoms. *Nature*, 415:39 EP –, 01 2002.

[6] Horst L. Stormer, Daniel C. Tsui, and Arthur C. Gossard. The fractional quantum hall effect. *Rev. Mod. Phys.*, 71:S298–S305, Mar 1999.

[7] Elbio Dagotto. Correlated electrons in high-temperature superconductors. *Rev. Mod. Phys.*, 66:763–840, Jul 1994.

[8] Xiao-Gang Wen. Colloquium: Zoo of quantum-topological phases of matter. *Rev. Mod. Phys.*, 89:041004, Dec 2017.

[9] Rahul Nandkishore and David A. Huse. Many-body localization and thermalization in quantum statistical mechanics. *Annual Review of Condensed Matter Physics*, 6(1):15–38, 2015.

[10] Alexander Weiße and Holger Fehske. *Exact Diagonalization Techniques*, pages 529–544. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008.

[11] Phillip Weinberg and Marin Bukov. QuSpin: a Python Package for Dynamics and Exact Diagonalisation of Quantum Many Body Systems part I: spin chains. *SciPost Phys.*, 2:003, 2017.

[12] Cornelius Lanczos. An iteration method for the solution of the eigenvalue problem of linear differential and integral operators. *J. Res. Natl. Bur. Stand. B*, 45:255–282, 1950.

[13] A. Wietek and A. M. Läuchli. Sublattice Coding Algorithm and Distributed Memory Parallelization for Large-Scale Exact Diagonalizations of Quantum Many-Body Systems. *ArXiv e-prints*, April 2018.

[14] Andreas M. Läuchli, Julien Sudan, and Erik S. Sørensen. Ground-state energy and spin gap of spin-$\frac{1}{2}$ kagomé-heisenberg antiferromagnetic clusters: Large-scale exact diagonalization results. *Phys. Rev. B*, 83:212401, Jun 2011.

[15] David J. Luitz, Nicolas Laflorencie, and Fabien Alet. Many-body localization edge in the random-field heisenberg chain. *Phys. Rev. B*, 91:081103, Feb 2015.

[16] Vedika Khemani, D. N. Sheng, and David A. Huse. Two universality classes for the many-body localization transition. *Phys. Rev. Lett.*, 119:075702, Aug 2017.

[17] Luigi Amico, Rosario Fazio, Andreas Osterloh, and Vlatko Vedral. Entanglement in many-body systems. *Rev. Mod. Phys.*, 80:517–576, May 2008.

[18] Giuliano Benenti, Giulio Casati, and Giuliano Strini. *Principles of Quantum Computation and Information*. WORLD SCIENTIFIC, 2004.

[19] Alfrd Rnyi. On measures of entropy and information. In *Proceedings of the Fourth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Contributions to the Theory of Statistics*, pages 547–561, Berkeley, Calif., 1961. University of California Press.

[20] Siddhartha Sen. Average entropy of a quantum subsystem. *Phys. Rev. Lett.*, 77:1–3, Jul 1996.

[21] Mark Srednicki. Entropy and area. *Phys. Rev. Lett.*, 71:666–669, Aug 1993.

[22] Michael M. Wolf, Frank Verstraete, Matthew B. Hastings, and J. Ignacio Cirac. Area laws in quantum systems: Mutual information and correlations. *Phys. Rev. Lett.*, 100:070502, Feb 2008.

[23] J. Eisert, M. Cramer, and M. B. Plenio. Colloquium: Area laws for the entanglement entropy. *Rev. Mod. Phys.*, 82:277–306, Feb 2010.

[24] M B Hastings. An area law for one-dimensional quantum systems. *Journal of Statistical Mechanics: Theory and Experiment*, 2007(08):P08024, 2007.

[25] E.H. Lieb and D.W. Robinson. The finite group velocity of quantum spin systems.

[26] G. Vidal, J. I. Latorre, E. Rico, and A. Kitaev. Entanglement in quantum critical phenomena. *Phys. Rev. Lett.*, 90:227902, Jun 2003.

[27] J. I. Latorre, E. Rico, and G. Vidal. Ground state entanglement in quantum spin chains. *Quantum Info. Comput.*, 4(1):48–92, January 2004.

[28] M. Fannes, B. Haegeman, and M. Mosonyi. Entropy growth of shift-invariant states on a quantum spin chain. *Journal of Mathematical Physics*, 44(12):6005–6019, 2003.

[29] Pasquale Calabrese and John Cardy. Entanglement entropy and quantum field theory. *Journal of Statistical Mechanics: Theory and Experiment*, 2004(06):P06002, 2004.

[30] Pasquale Calabrese and John Cardy. Entanglement entropy and conformal field theory. *Journal of Physics A: Mathematical and Theoretical*, 42(50):504005, 2009.

[31] Christoph Holzhey, Finn Larsen, and Frank Wilczek. Geometric and renormalized entropy in conformal field theory. *Nuclear Physics B*, 424(3):443 – 467, 1994.

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

[33] Itensor v2.1.1.

[34] F. Verstraete, V. Murg, and J.I. Cirac. Matrix product states, projected entangled pair states, and variational renormalization group methods for quantum spin systems. *Advances in Physics*, 57(2):143–224, 2008.

[35] Romn Ors. A practical introduction to tensor networks: Matrix product states and projected entangled pair states. *Annals of Physics*, 349:117 – 158, 2014.

[36] F.F. Assaad and H.G. Evertz. World line and determinantal quantum monte carlo methods for spins, phonons, and electrons. *Lect. Notes Phys.*, 739:277–356, 2008.

[37] N.V Prokof'ev, B.V Svistunov, and I.S Tupitsyn. worm algorithm in quantum monte carlo simulations. *Physics Letters A*, 238(4):253 – 257, 1998.

[38] Olav F. Syljuåsen and Anders W. Sandvik. Quantum monte carlo with directed loops. *Phys. Rev. E*, 66:046701, Oct 2002.

[39] Anders W. Sandvik. Stochastic series expansion method with operator-loop update. *Phys. Rev. B*, 59:R14157–R14160, Jun 1999.

[40] W. M. C. Foulkes, L. Mitas, R. J. Needs, and G. Rajagopal. Quantum monte carlo simulations of solids. *Rev. Mod. Phys.*, 73:33–83, Jan 2001.

[41] Federico Becca and Sandro Sorella. *Quantum Monte Carlo Approaches for Correlated Systems*. Cambridge University Press, 2017.

[42] Steven R. White and A. L. Chernyshev. Neél order in square and triangular lattice heisenberg models. *Phys. Rev. Lett.*, 99:127004, Sep 2007.

[43] Luca Capriotti, Adolfo E. Trumper, and Sandro Sorella. Long-range néel order in the triangular heisenberg model. *Phys. Rev. Lett.*, 82:3899–3902, May 1999.

[44] Zheng Weihong, Ross H. McKenzie, and Rajiv R. P. Singh. Phase diagram for a class of spin-$\frac{1}{2}$ heisenberg models interpolating between the square-lattice, the triangular-lattice, and the linear-chain limits. *Phys. Rev. B*, 59:14367–14375, Jun 1999.

[45] Matthew P. A. Fisher, Peter B. Weichman, G. Grinstein, and Daniel S. Fisher. Boson localization and the superfluid-insulator transition. *Phys. Rev. B*, 40:546–570, Jul 1989.

[46] Matthias Troyer and Uwe-Jens Wiese. Computational complexity and fundamental limitations to fermionic quantum monte carlo simulations. *Phys. Rev. Lett.*, 94:170201, May 2005.

[47] Julia Wildeboer and Alexander Seidel. Correlation functions in su(2)-invariant resonating-valence-bond spin liquids on nonbipartite lattices. *Phys. Rev. Lett.*, 109:147208, Oct 2012.

[48] O Perron. Zur theorie der matrices. *Math. Ann.*, 64:248, 1907.

[49] G Frobenius. Ueber matrizen aus nicht negativen elementen. *Sitzungsberichte der Kniglich Preussischen Akademie der Wissenschaften*, pages 456–477, 1912.

[50] Sergey Bravyi, David P. Divincenzo, Roberto Oliveira, and Barbara M. Terhal. The complexity of stoquastic local hamiltonian problems. *Quantum Info. Comput.*, 8(5):361–385, May 2008.

[51] R.P. Feynman. Simulating physics with computers. *International Journal of Theoretical Physics*, 21(6-7):467, 1982.

[52] Seth Lloyd. Universal quantum simulators. *Science*, 273(5278):1073–1078, 1996.

[53] M. Born and V. Fock. Beweis des Adiabatensatzes. *Zeitschrift fur Physik*, 51:165–180, March 1928.

[54] Pasquale Calabrese and John Cardy. Evolution of entanglement entropy in one-dimensional systems. *Journal of Statistical Mechanics: Theory and Experiment*, 2005(04):P04010, 2005.

[55] J. M. Martinis and K. Osborne. Superconducting Qubits and the Physics of Josephson Junctions. *eprint arXiv:cond-mat/0402415*, February 2004.

[56] Andrew A. Houck, Hakan E. Türeci, and Jens Koch. On-chip quantum simulation with superconducting circuits. *Nature Physics*, 8:292 EP –, 04 2012.

[57] Jens Koch, Terri M. Yu, Jay Gambetta, A. A. Houck, D. I. Schuster, J. Majer, Alexandre Blais, M. H. Devoret, S. M. Girvin, and R. J. Schoelkopf. Charge-insensitive qubit design derived from the cooper pair box. *Phys. Rev. A*, 76:042319, Oct 2007.

[58] Abhinav Kandala, Antonio Mezzacapo, Kristan Temme, Maika Takita, Markus Brink, Jerry M. Chow, and Jay M. Gambetta. Hardware-efficient variational quantum eigensolver for small molecules and quantum magnets. *Nature*, 549:242 EP –, 09 2017.

[59] Nikolaj Moll, Panagiotis Barkoutsos, Lev S Bishop, Jerry M Chow, Andrew Cross, Daniel J Egger, Stefan Filipp, Andreas Fuhrer, Jay M Gambetta, Marc Ganzhorn, Abhinav Kandala, Antonio Mezzacapo, Peter Mller, Walter Riess, Gian Salis, John Smolin, Ivano Tavernelli, and Kristan Temme. Quantum optimization using variational algorithms on near-term quantum devices. *Quantum Science and Technology*, 3(3):030503, 2018.

[60] A. Kandala, K. Temme, A. D. Corcoles, A. Mezzacapo, J. M. Chow, and J. M. Gambetta. Extending the computational reach of a noisy superconducting quantum processor. *ArXiv e-prints*, May 2018.

[61] John Preskill. Quantum Computing in the NISQ era and beyond. *Quantum*, 2:79, August 2018.

[62] John Preskill and Spiros Michalakis. Quantum computers animated. `https://www.youtube.com/watch?v=T2DXrs0OpHU`, 2013. [Online; accessed 27-August-2018].

[63] Kevin P. Murphy. *Machine Learning: A Probabilistic Perspective*. The MIT Press, 2012.

[64] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521:436–444, 2008.

[65] G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006.

[66] A. Krizhevsky, I. Sutskever, and G. Hinton. Imagenet classification with deep convolutional neural networks. *Proc. Advances in Neural Information Processing Systems*, 25:1090–1098, 2012.

[67] G. Hinton. A practical guido to training restricted boltzmann machines. *Neural Networks: Tricks of the Trade*, pages 599–619, 2012.

[68] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12:2493–2537, 2011.

[69] P. Baldi, P. Sadowski, and D. Whiteson. Searching for exotic particles in high-energy physics with deep learning. *Nature Communications*, 5:4308, 2014.

[70] L. F. Arsenault, O. A. von Lilienfeld, and A. J. Millis. Machine learning for many-body physics: efficient solution of dynamical mean-field theory. *arXiv:1506.08858*, 2015.

[71] Juan Carrasquilla and Roger G. Melko. Machine learning phases of matter. *Nature Physics*, 13:431–434, February 2017.

[72] Evert P. L. van Nieuwenburg, Ye-Hua Liu, and Sebastian D. Huber. Learning phase transitions by confusion. *Nature Physics*, 13:435–439, February 2017.

[73] Giacomo Torlai and Roger G. Melko. Learning thermodynamics with Boltzmann machines. *Physical Review B*, 94(16):165134, October 2016.

[74] Giuseppe Carleo and Matthias Troyer. Solving the quantum many-body problem with artificial neural networks. *Science*, 355(6325):602–606, February 2017.

[75] Li Huang and Lei Wang. Accelerated monte carlo simulations with restricted boltzmann machines. *Phys. Rev. B*, 95:035105, Jan 2017.

[76] Junwei Liu, Yang Qi, Zi Yang Meng, and Liang Fu. Self-learning monte carlo method. *Phys. Rev. B*, 95:041101, Jan 2017.

[77] Dong-Ling Deng, Xiaopeng Li, and S. Das Sarma. Quantum entanglement in neural network states. *Phys. Rev. X*, 7:021021, May 2017.

[78] Yusuke Nomura, Andrew S. Darmawan, Youhei Yamaji, and Masatoshi Imada. Restricted boltzmann machine learning for solving strongly correlated quantum systems. *Phys. Rev. B*, 96:205152, Nov 2017.

[79] Dong-Ling Deng, Xiaopeng Li, and S. Das Sarma. Machine learning topological states. *Phys. Rev. B*, 96:195145, Nov 2017.

[80] Xun Gao and Lu-Ming Duan. Efficient representation of quantum many-body states with deep neural networks. *Nature Communications*, 8(1):662, 2017.

[81] Giacomo Torlai and Roger G. Melko. Neural decoder for topological codes. *Phys. Rev. Lett.*, 119:030501, Jul 2017.

[82] Savvas Varsamopoulos, Ben Criger, and Koen Bertels. Decoding small surface codes with feedforward neural networks. *Quantum Science and Technology*, 3(1):015004, 2018.

[83] Paul Baireuther, Thomas E. O'Brien, Brian Tarasinski, and Carlo W. J. Beenakker. Machine-learning-assisted correction of correlated qubit errors in a topological code. *Quantum*, 2:48, January 2018.

[84] Christopher Chamberland and Pooya Ronagh. Deep neural decoders for near term fault-tolerant experiments. *Quantum Science and Technology*, 3(4):044002, 2018.

[85] N. Maskara, A. Kubica, and T. Jochym-O'Connor. Advantages of versatile neural-network decoding for topological codes. *ArXiv e-prints*, February 2018.

[86] Jing Chen, Song Cheng, Haidong Xie, Lei Wang, and Tao Xiang. Equivalence of restricted boltzmann machines and tensor network states. *Phys. Rev. B*, 97:085104, Feb 2018.

[87] Y. Huang and J. E. Moore. Neural network representation of tensor network and chiral states. *ArXiv e-prints*, January 2017.

[88] Ivan Glasser, Nicola Pancotti, Moritz August, Ivan D. Rodriguez, and J. Ignacio Cirac. Neural-network quantum states, string-bond states, and chiral topological states. *Phys. Rev. X*, 8:011006, Jan 2018.

[89] Y. Levine, O. Sharir, N. Cohen, and A. Shashua. Bridging Many-Body Quantum Physics and Deep Learning via Tensor Networks. *ArXiv e-prints*, March 2018.

[90] P. Mehta and D. J. Schwab. An exact mapping between the Variational Renormalization Group and Deep Learning. *ArXiv e-prints*, October 2014.

[91] Maciej Koch-Janusz and Zohar Ringel. Mutual information, neural networks and the renormalization group. *Nature Physics*, 14(6):578–582, 2018.

[92] S.-H. Li and L. Wang. Neural Network Renormalization Group. *ArXiv e-prints*, February 2018.

[93] Yi-Zhuang You, Zhao Yang, and Xiao-Liang Qi. Machine learning spatial geometry from entanglement features. *Phys. Rev. B*, 97:045153, Jan 2018.

[94] Edwin Stoudenmire and David J Schwab. Supervised learning with tensor networks. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 4799–4807. Curran Associates, Inc., 2016.

[95] I. Glasser, N. Pancotti, and J. I. Cirac. Supervised learning with generalized tensor networks. *ArXiv e-prints*, June 2018.

[96] Zhao-Yu Han, Jun Wang, Heng Fan, Lei Wang, and Pan Zhang. Unsupervised generative modeling using matrix product states. *Phys. Rev. X*, 8:031012, Jul 2018.

[97] E Miles Stoudenmire. Learning relevant features of data with multi-scale tensor networks. *Quantum Science and Technology*, 3(3):034003, 2018.

[98] Wikipedia contributors. Cyc — Wikipedia, the free encyclopedia, 2018. [Online; accessed 1-September-2018].

[99] David E. Rumelhart, James L. McClelland, and CORPORATE PDP Research Group, editors. *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Vol. 1: Foundations*. MIT Press, Cambridge, MA, USA, 1986.

[100] Warren S. McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4):115–133, 1943.

[101] Jeremy Bernstein. A.I. `https://www.newyorker.com/magazine/1981/12/14/a-i`, 1981. [Online; accessed 5-September-2018].

[102] Frank Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65(6):386–408, 1958.

[103] Mikel Olazaran. A sociological study of the official history of the perceptrons controversy. *Social Studies of Science*, 26(3):611–659, 1996.

[104] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors. *Nature*, 323:533 EP –, 10 1986.

[105] J J Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences*, 79(8):2554–2558, 1982.

[106] J. J. Hopfield, D. I. Feinstein, and R. G. Palmer. 'unlearning'has a stabilizing effect in collective memories. *Nature*, 304:158 EP –, 07 1983.

[107] P. J. M. Laarhoven and E. H. L. Aarts, editors. *Simulated Annealing: Theory and Applications*. Kluwer Academic Publishers, Norwell, MA, USA, 1987.

[108] P. Smolensky. Parallel distributed processing: Explorations in the microstructure of cognition, vol. 1. chapter Information Processing in Dynamical Systems: Foundations of Harmony Theory, pages 194–281. MIT Press, Cambridge, MA, USA, 1986.

[109] W. K. Hastings. Monte carlo sampling methods using markov chains and their applications. *Biometrika*, 57(1):97–109, 1970.

[110] Geoffrey E. Hinton. Training products of experts by minimizing contrastive divergence. *Neural Computation*, 14(8):1771–1800, 2002.

[111] Y Bengio and O Delalleau. Justifying and generalizing contrastive divergence. *Neural Computation*, 21:1601–21, 2009.

[112] A Fischer and C. Igel. Bounding the bias of contrastive divergence learning. *Neural Computation*, 23:664–73, 2011.

[113] Miguel Á. Carreira-Perpiñán and Geoffrey E. Hinton. On contrastive divergence learning. In *AISTATS*, 2005.

[114] Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. On the importance of initialization and momentum in deep learning. In Sanjoy Dasgupta and David McAllester, editors, *Proceedings of the 30th International Conference on Machine Learning*, volume 28 of *Proceedings of Machine Learning Research*, pages 1139–1147, Atlanta, Georgia, USA, 17–19 Jun 2013. PMLR.

[115] D. P. Kingma and J. Ba. Adam: A Method for Stochastic Optimization. *ArXiv e-prints*, December 2014.

[116] John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *J. Mach. Learn. Res.*, 12:2121–2159, July 2011.

[117] M. D. Zeiler. ADADELTA: An Adaptive Learning Rate Method. *ArXiv e-prints*, December 2012.

[118] Shun ichi Amari. Neural learning in structured parameter spaces - natural riemannian gradient. In *In Advances in Neural Information Processing Systems*, pages 127–133. MIT Press, 1997.

[119] Razvan Pascanu and Yoshua Bengio. Revisiting natural gradient for deep networks. In *International Conference on Learning Representations 2014 (Conference Track)*, April 2014.

[120] Shun-ichi Amari. Natural Gradient Works Efficiently in Learning. *Neural Computation*, 10(2):251–276, February 1998.

[121] Sandro Sorella. Green function monte carlo with stochastic reconfiguration. *Physical review letters*, 80(20):4558, 1998.

[122] Anders Krogh and John A. Hertz. A simple weight decay can improve generalization. In *ADVANCES IN NEURAL INFORMATION PROCESSING SYSTEMS 4*, pages 950–957. Morgan Kaufmann, 1992.

[123] Robert H. Swendsen and Jian-Sheng Wang. Replica monte carlo simulation of spin-glasses. *Phys. Rev. Lett.*, 57:2607–2609, Nov 1986.

[124] Radford M. Neal. Annealed importance sampling. *Statistics and Computing*, 11(2):125–139, April 2001.

[125] Ruslan Salakhutdinov and Ruslan Salakhutdinov. Learning and evaluating boltzmann machines. Technical report, 2008.

[126] R. Peierls. On ising's model of ferromagnetism. *Mathematical Proceedings of the Cambridge Philosophical Society*, 32(3):477?481, 1936.

[127] H. A. Kramers and G. H. Wannier. Statistics of the two-dimensional ferromagnet. part i. *Phys. Rev.*, 60:252–262, Aug 1941.

[128] Lars Onsager. Crystal statistics. i. a two-dimensional model with an order-disorder transition. *Phys. Rev.*, 65:117–149, Feb 1944.

[129] Ulli Wolff. Collective monte carlo updating for spin systems. *Phys. Rev. Lett.*, 62:361–364, Jan 1989.

[130] Anders W. Sandvik. Computational studies of quantum spin systems. *AIP Conference Proceedings*, 1297(1):135–338, 2010.

[131] Nicolas Le Roux and Yoshua Bengio. Representational power of restricted boltzmann machines and deep belief networks. *Neural Comput.*, 20(6):1631–1649, June 2008.

[132] Matthew B. Hastings, Iván González, Ann B. Kallin, and Roger G. Melko. Measuring Renyi Entanglement Entropy in Quantum Monte Carlo Simulations. *Physical Review Letters*, 104(15):157201, April 2010.

[133] Kallin, Ann Berlinsky. *Computational Methods for the Measurement of Entanglement in Condensed Matter Systems*. PhD thesis, 2014.

[134] Ruslan Salakhutdinov, Andriy Mnih, and Geoffrey Hinton. Restricted boltzmann machines for collaborative filtering. In *Proceedings of the 24th International Conference on Machine Learning*, ICML '07, pages 791–798, New York, NY, USA, 2007. ACM.

[135] Dong-Ling Deng, Xiaopeng Li, and S. Das Sarma. Machine learning topological states. *Phys. Rev. B*, 96:195145, Nov 2017.

[136] G. Carleo, Y. Nomura, and M. Imada. Constructing exact representations of quantum many-body systems with deep neural networks. *ArXiv e-prints*, February 2018.

[137] Dong-Ling Deng, Xiaopeng Li, and S. Das Sarma. Quantum entanglement in neural network states. *Phys. Rev. X*, 7:021021, May 2017.

[138] Yusuke Nomura, Andrew S. Darmawan, Youhei Yamaji, and Masatoshi Imada. Restricted boltzmann machine learning for solving strongly correlated quantum systems. *Phys. Rev. B*, 96:205152, Nov 2017.

[139] Zi Cai and Jinguo Liu. Approximating quantum many-body wave functions using artificial neural networks. *Phys. Rev. B*, 97:035116, Jan 2018.

[140] K Vogel and H Risken. Determination of quasiprobability distributions in terms of probability distributions for the rotated quadrature phase. *Physical Review A*, 40(5):2847, 1989.

[141] Ulf Leonhardt. Quantum-State Tomography and Discrete Wigner Function. *Physical Review Letters*, 74(21):4101–4105, May 1995.

[142] Andrew G. White, Daniel F. V. James, Philippe H. Eberhard, and Paul G. Kwiat. Nonmaximally Entangled States: Production, Characterization, and Utilization. *Physical Review Letters*, 83(16):3103–3107, October 1999.

[143] Giacomo Torlai, Guglielmo Mazzola, Juan Carrasquilla, Matthias Troyer, Roger Melko, and Giuseppe Carleo. Neural-network quantum state tomography. *Nature Physics*, 14(5):447–450, 2018.

[144] Giacomo Torlai and Roger G. Melko. Latent space purification via neural density operators. *Phys. Rev. Lett.*, 120:240503, Jun 2018.

[145] Giacomo Torlai *et al.* Quantum data compression of a Rydberg-atom quantum simulator. *In preparation*, 2018.

[146] Daniel F. V. James, Paul G. Kwiat, William J. Munro, and Andrew G. White. Measurement of qubits. *Phys. Rev. A*, 64:052312, Oct 2001.

[147] H. Häffner, W. Hänsel, C. F. Roos, J. Benhelm, D. Chek-al kar, M. Chwalla, T. Körber, U. D. Rapol, M. Riebe, P. O. Schmidt, C. Becher, O. Gühne, W. Dür, and R. Blatt. Scalable multiparticle entanglement of trapped ions. *Nature*, 438(7068):643–646, December 2005.

[148] Chao-Yang Lu, Xiao-Qi Zhou, Otfried Gühne, Wei-Bo Gao, Jin Zhang, Zhen-Sheng Yuan, Alexander Goebel, Tao Yang, and Jian-Wei Pan. Experimental entanglement of six photons in graph states. *Nature Physics*, 3(2):91–95, February 2007.

[149] David Gross, Yi-Kai Liu, Steven T. Flammia, Stephen Becker, and Jens Eisert. Quantum state tomography via compressed sensing. *Phys. Rev. Lett.*, 105:150401, Oct 2010.

[150] G. Tóth, W. Wieczorek, D. Gross, R. Krischek, C. Schwemmer, and H. Weinfurter. Permutationally invariant quantum tomography. *Phys. Rev. Lett.*, 105:250403, Dec 2010.

[151] M Cramer, MB Plenio, ST Flammia, R Somma, D Gross, SD Bartlett, O Landon-Cardinal, D Poulin, and YK Liu. Efficient quantum state tomography. *Nature communications*, 1:149, 2009.

[152] BP Lanyon, C Maier, M Holzäpfel, T Baumgratz, C Hempel, P Jurcevic, I Dhand, AS Buyskikh, AJ Daley, M Cramer, et al. Efficient tomography of a quantum many-body system. *arXiv preprint arXiv:1612.08000*, 2016.

[153] Xi-Lin Wang, Luo-Kan Chen, W. Li, H.-L. Huang, C. Liu, C. Chen, Y.-H. Luo, Z.-E. Su, D. Wu, Z.-D. Li, H. Lu, Y. Hu, X. Jiang, C.-Z. Peng, L. Li, N.-L. Liu, Yu-Ao Chen, Chao-Yang Lu, and Jian-Wei Pan. Experimental Ten-Photon Entanglement. *Physical Review Letters*, 117(21):210502, November 2016.

[154] G. Tóth, W. Wieczorek, D. Gross, R. Krischek, C. Schwemmer, and H. Weinfurter. Permutationally invariant quantum tomography. *Phys. Rev. Lett.*, 105:250403, Dec 2010.

[155] M Cramer, MB Plenio, ST Flammia, R Somma, D Gross, SD Bartlett, O Landon-Cardinal, D Poulin, and YK Liu. Efficient quantum state tomography. *Nature communications*, 1:149, 2009.

[156] H. Häffner, W. Hänsel, C. F. Roos, J. Benhelm, D. Chek-al kar, M. Chwalla, T. Körber, U. D. Rapol, M. Riebe, P. O. Schmidt, C. Becher, O. Gühne, W. Dür, and R. Blatt. Scalable multiparticle entanglement of trapped ions. *Nature*, 438(7068):643–646, December 2005.

[157] Steven T Flammia, David Gross, Yi-Kai Liu, and Jens Eisert. Quantum tomography via compressed sensing: error bounds, sample complexity and efficient estimators. *New Journal of Physics*, 14(9):095022, 2012.

[158] Tobias Moroder, Philipp Hyllus, Gza Tth, Christian Schwemmer, Alexander Niggebaum, Stefanie Gaile, Otfried Ghne, and Harald Weinfurter. Permutationally invariant state reconstruction. *New Journal of Physics*, 14(10):105001, 2012.

[159] Bernd Lücke, Jan Peise, Giuseppe Vitagliano, Jan Arlt, Luis Santos, Géza Tóth, and Carsten Klempt. Detecting multiparticle entanglement of dicke states. *Phys. Rev. Lett.*, 112:155304, Apr 2014.

[160] Christian Schwemmer, Géza Tóth, Alexander Niggebaum, Tobias Moroder, David Gross, Otfried Gühne, and Harald Weinfurter. Experimental comparison of efficient tomography schemes for a six-qubit state. *Phys. Rev. Lett.*, 113:040503, Jul 2014.

[161] P G de Gennes. Collective motions of hydrogen bonds. *Solid State Communications*, 1(6):132–137, 1963.

[162] Masuo 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(5):1454, 1976.

[163] H. Rieger and N. Kawashima. Application of a continuous time cluster algorithm to the two-dimensional random quantum ising ferromagnet. *The European Physical Journal B - Condensed Matter and Complex Systems*, 9(2):233–236, 1999.

[164] Guglielmo Mazzola and Matthias Troyer. Quantum monte carlo annealing with multi-spin dynamics. *Journal of Statistical Mechanics: Theory and Experiment*, 2017(5):053105, 2017.

[165] H. G. Evertz. The loop algorithm. *Advances in Physics*, 52(1):1–66, January 2003.

[166] Florent Krzakala, Alberto Rosso, Guilhem Semerjian, and Francesco Zamponi. Path-integral representation for quantum spin models: Application to the quantum cavity method and monte carlo simulations. *Phys. Rev. B*, 78:134428, Oct 2008.

[167] Rajibul Islam, Ruichao Ma, Philipp M. Preiss, M. Eric Tai, Alexander Lukin, Matthew Rispoli, and Markus Greiner. Measuring entanglement entropy in a quantum many-body system. *Nature*, 528(7580):77–83, December 2015.

[168] Philip Richerme, Zhe-Xuan Gong, Aaron Lee, Crystal Senko, Jacob Smith, Michael Foss-Feig, Spyridon Michalakis, Alexey V. Gorshkov, and Christopher Monroe. Non-local propagation of correlations in quantum systems with long-range interactions. *Nature*, 511(7508):198–201, 07 2014.

[169] Deny R. Hamel, Lynden K. Shalm, Hannes Hübel, Aaron J. Miller, Francesco Marsili, Varun B. Verma, Richard P. Mirin, Sae Woo Nam, Kevin J. Resch, and Thomas Jennewein. Direct generation of three-photon polarization entanglement. *Nature Photonics*, 8:801 EP –, 09 2014.

[170] M. Bellini S. Grandi, A. Zavatta and M. G. A. Paris. Experimental quantum tomography of a homodyne detector. *New Journal of Physics*, 19:053015, 2017.

[171] J. Lavoie, R. Kaltenbaek, and K. J. Resch. Experimental violation of svetlichny's inequality. *New Journal of Physics*, page 073051, 2009.

[172] Milena D'Angelo, Alessandro Zavatta, Valentina Parigi, and Marco Bellini. Tomographic test of bell's inequality for a time-delocalized single photon. *Phys. Rev. A*, 74:052114, Nov 2006.

[173] Matthew D. Zeiler. ADADELTA: an adaptive learning rate method. *arxiv1212:5701*, 2012.

[174] K. Banaszek, G. M. D'Ariano, M. G. A. Paris, and M. F. Sacchi. Maximum-likelihood estimation of the density matrix. *Phys. Rev. A*, 61:010304, Dec 1999.

[175] J.B. Altepeter, E.R. Jeffrey, and P.G. Kwiat. Photonic state tomography. *Advances in Atomic, Molecular, and Optical Physics*, 52, 2006.

[176] Immanuel Bloch, Jean Dalibard, and Sylvain Nascimbène. Quantum simulations with ultracold quantum gases. *Nature Physics*, 8:267 EP –, 04 2012.

[177] R. Blatt and C. F. Roos. Quantum simulations with trapped ions. *Nature Physics*, 8:277 EP –, 04 2012.

[178] C. Monroe and J. Kim. Scaling the ion trap quantum processor. *Science*, 339(6124):1164–1169, 2013.

[179] Hendrik Weimer, Markus Müller, Igor Lesanovsky, Peter Zoller, and Hans Peter Büchler. A rydberg quantum simulator. *Nature Physics*, 6:382 EP –, 03 2010.

[180] Manuel Endres, Hannes Bernien, Alexander Keesling, Harry Levine, Eric R. Anschuetz, Alexandre Krajenbrink, Crystal Senko, Vladan Vuletic, Markus Greiner, and Mikhail D. Lukin. Atom-by-atom assembly of defect-free one-dimensional cold atom arrays. *Science*, 2016.

[181] D. Jaksch, J. I. Cirac, P. Zoller, S. L. Rolston, R. Côté, and M. D. Lukin. Fast quantum gates for neutral atoms. *Phys. Rev. Lett.*, 85:2208–2211, Sep 2000.

[182] Paul Fendley, K. Sengupta, and Subir Sachdev. Competing density-wave orders in a one-dimensional hard-boson model. *Phys. Rev. B*, 69:075106, Feb 2004.

[183] Hannes Bernien, Sylvain Schwartz, Alexander Keesling, Harry Levine, Ahmed Omran, Hannes Pichler, Soonwon Choi, Alexander S. Zibrov, Manuel Endres, Markus Greiner, Vladan Vuletić, and Mikhail D. Lukin. Probing many-body dynamics on a 51-atom quantum simulator. *Nature*, 551:579 EP –, 11 2017.

[184] J.R. Johansson, P.D. Nation, and Franco Nori. Qutip 2: A python framework for the dynamics of open quantum systems. *Computer Physics Communications*, 184(4):1234 – 1240, 2013.

[185] W. S. Bakr, A. Peng, M. E. Tai, R. Ma, J. Simon, J. I. Gillen, S. Fölling, L. Pollet, and M. Greiner. Probing the superfluid-to-mott Insulator Transition at the Single-Atom Level. *Science*, 329(5991):547–550, July 2010.

[186] M. W. Johnson, M. H. S. Amin, S. Gildert, T. Lanting, F. Hamze, N. Dickson, R. Harris, A. J. Berkley, J. Johansson, P. Bunyk, E. M. Chapple, C. Enderud, J. P. Hilton, K. Karimi, E. Ladizinsky, N. Ladizinsky, T. Oh, I. Perminov, C. Rich, M. C. Thom, E. Tolkacheva, C. J. S. Truncik, S. Uchaikin, J. Wang, B. Wilson, and G. Rose. Quantum annealing with manufactured spins. *Nature*, 473(7346):194–198, May 2011.

[187] D. Nigg, M. Mueller, E. A. Martinez, P. Schindler, M. Hennrich, T. Monz, M. A. Martin-Delgado, and R. Blatt. Quantum computations on a topologically encoded qubit. *Science*, 345(6194):302–305, 2014.

[188] Hector Bombin. Topological codes. In D. A. Lidar and T. A. Brun, editors, *Quantum Error Correction*, chapter 19. Cambridge, 2013.

[189] Austin G. Fowler, Matteo Mariantoni, John M. Martinis, and Andrew N. Cleland. Surface codes: Towards practical large-scale quantum computation. *Phys. Rev. A*, 86:032324, 2012.

[190] H. Bombin and M. A. Martin-Delgado. Quantum measurements and gates by code deformation. *J. Phys. A:Math. Theor.*, 42:095302, 2009.

[191] Daniel Gottesman. Stabilizer codes and quantum error correction. *arXiv:quant-ph/9705052*, 1997.

[192] A. Yu. Kitaev. Fault-tolerant quantum computation by anyons. *Annals of Physics*, 1:2–30, 2003.

[193] E. Dennis, A. Kitaev, A. Landahl, and J. Preskill. Topological quantum memory. *Journal of Mathematical Physics*, 43:4452, 2002.

[194] G. Duclos-Cianci and D. Poulin. Fast decoders for topological quantum codes. *Phys. Rev. Lett.*, 104:050504, 2010.

[195] G. Duclos-Cianci and D. Poulin. Fault-tolerant renormalization group decoder for abelian topological codes. *Quant. Inf. Comp.*, 14:0721–0740, 2014.

[196] J. R. Wootton and D. Loss. High threshold error correction for the surface code. *Phys. Rev. Lett.*, 109:160503, 2012.

[197] Adrian Hutter, James R. Wootton, and Daniel Loss. Efficient markov chain monte carlo algorithm for the surface code. *Phys. Rev. A*, 89:022326, Feb 2014.

[198] A. Fowler. Optimal complexity correction of correlated errors in the surface code. *arXiv:1310.0863*, 2013.

[199] S. Bravyi, M. Suchara, and A. Vargo. Efficient algorithms for maximum likelihood decoding in the surface code. *Phys. Rev. A*, 90:032326, 2014.

[200] B. Heim, K. M. Svore, and M. B. Hastings. Optimal circuit-level decoding for surface codes. *arXiv:1609.06373*, 2016.

[201] J. Edmonds. Paths, trees and flowers. *Canadian Journal of Mathematics*, 17:449, 1997.

[202] Y. Huang and J. E. Moore. Neural network representation of tensor network and chiral states. *ArXiv e-prints*, January 2017.

[203] Fern H E Watson and Sean D Barrett. Logical error rate scaling of the toric code. *New Journal of Physics*, 16(9):093045, 2014.

[204] V. Kolmogorov. Blossom v: a new implementation of a minimum cost perfect matching algorithm. *Math. Prog. Comp.*, 1:43, 2002.

[205] Austing G. Fowler, Adam C. Whiteside, and Lloyd C. L. Hollenberg. Towards practical classical processing for the surface code. *Phys. Rev. Lett.*, 108:180501, 2012.

[206] Juan Carrasquilla and Roger G. Melko. Machine learning phases of matter. *Nat Phys*, 13(5):431–434, 05 2017.

[207] Dong-Ling Deng, X. Li, and S. Das Sarma. Exact Machine Learning Topological States. *arXiv:1609.09060*, 2016.

[208] E. Novais and E. R. Mucciolo. Surface code threshold in the presence of correlated errors. *Phys. Rev. Lett.*, 110:010502, 2013.

[209] C. Wang, J. Harrington, and J. Preskill. Confinement-higgs transition in a disordered gauge theory and the accuracy threshold for quantum memory. *Annals of Physics*, 1:31–58, 2003.

[210] H. G. Katzgraber, H. Bombin, and M. A. Martin-Delgado. Error threshold for color codes and random three-body ising models. *Phys. Rev. Lett*, 103:090501, 2009.

[211] B. J. Brown, N. H. Nickerson, and D. E. Browne. Fault-tolerant error correction with the gauge color code. *Nature Communications*, 7, 2016.

[212] Richard S. Sutton and Andrew G. Barto. *Introduction to Reinforcement Learning*. MIT Press, Cambridge, MA, USA, 1st edition, 1998.

[213] Geoffrey E. Hinton, Simon Osindero, and Yee-Whye Teh. A fast learning algorithm for deep belief nets. *Neural Computation*, 18(7):1527–1554, 2006. PMID: 16764513.

[214] Geoffrey Hinton. Learning multiple layers of represeantation. *Trends in Cognitive Science*, 10:428–434, 2007.

[215] G. Hinton, S. Osindero, and Y. Teh. A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554, 2006.

[216] Ruslan Salakhutdinov and I. Murray. On the quantitative analysis of deep belief networks. *ICML'08 Proceedings of the 25th international conference on machine learning*, pages 872–879, 2008.

[217] Yoshua Bengio. Twitter. `https://twitter.com/NicolasChapados/status/1018146629931753477`, 2018.

[218] A. Seif, K. A. Landsman, N. M. Linke, C. Figgatt, C. Monroe, and M. Hafezi. Machine learning assisted readout of trapped-ion qubits. *Journal of Physics B Atomic Molecular Physics*, 51(17):174006, September 2018.

[219] Sandeep Mavadia, Virginia Frey, Jarrah Sastrawan, Stephen Dona, and Michael J. Biercuk. Prediction and real-time compensation of qubit decoherence via machine learning. *Nature Communications*, 8:14106 EP –, 01 2017.

[220] M. Bukov, A. G. R. Day, D. Sels, P. Weinberg, A. Polkovnikov, and P. Mehta. Reinforcement Learning in Different Phases of Quantum Control. *ArXiv e-prints*, May 2017.