Department of Physics and Astronomy

University of Heidelberg

Master thesis

in Physics

submitted by

Robert Klassert

born in Aschaffenburg

2021

Simulation and Emulation of Probabilistic Quantum Spins with Neural Networks

This Master thesis has been carried out by Robert Klassert

at the

Kirchhoff-Institut für Physik

under the supervision of

Priv.-Doz. Dr. Martin Gärttner

and Dr. Andreas Baumbach

Simulation and Emulation of Probabilistic Quantum Spins with Neural Networks:

Generative artificial neural network representations of quantum states have become a powerful variational ansatz for the simulation of quantum spin systems.
In particular, restricted Boltzmann machines rely on probabilistic inference with Markov chain Monte Carlo methods to draw samples based on which physical observables can be estimated. In this regard, modern neuromorphic hardware for emulating spiking neural networks promises advantages over general purpose von Neumann computers when executing spike-based neural sampling, most notably, that inference is independent of the system size.
This thesis studies the mixed-signal neuromorphic chip BrainScaleS-2 as a platform for probabilistic neural quantum states. The results can be grouped into three projects. Firstly, we extended previous efforts of learning neuromorphic quantum state representations to ground states and steady states. Secondly, we developed a variational learning algorithm specific to the requirements of BrainScaleS-2 for the search of ground states in stoquastic systems and studied its performance for the transverse field Ising model. Finally, we examined limitations of the presented approaches and developed links to guide future research in this area.

Simulation und Emulation probabilistischer Quantenspins mit neuronalen Netzen:

Generative künstliche neuronale Netze zur Darstellung von Quantenzuständen haben sich zu einem leistungsfähigen Variationsansatz für die Simulation von Quantenspinsystemen entwickelt.
Insbesondere beschränkte Boltzmann-Maschinen stützen sich auf probabilistische Inferenz mit Markow-Ketten Monte-Carlo Methoden, um Stichproben zu ziehen, auf deren Grundlage physikalische Observablen geschätzt werden können. In dieser Hinsicht verspricht moderne neuromorphe Hardware für die Emulation von spikenden neuronalen Netzen Vorteile gegenüber allgemeinen Von-Neumann-Computern durch spike-basiertes neuronales Sampling, vor allem, da die Inferenz unabhängig von der Systemgröße ist.
Diese Arbeit untersucht den neuromorphen Mixed-Signal-Chip BrainScaleS-2 als Plattform für probabilistische neuronale Quantenzustände. Die Ergebnisse lassen sich in drei Projekte gliedern. Erstens haben wir frühere Bemühungen zum Erlernen neuromorpher Quantenzustandsrepräsentationen auf Grundzustände und stationäre Zustände ausgedehnt. Zweitens haben wir ein auf die Anforderungen von BrainScaleS-2 zugeschnittenes variationelles Lernverfahren für die Suche nach Grundzuständen in stoquastischen Systemen entwickelt und dieses am Beispiel des Ising-Modells mit transversalem Feld untersucht. Schließlich haben wir die Limitationen und Perspektiven der vorgestellten Ansätze geprüft und Anknüpfungspunkte für zukünftige Forschung in diesem Bereich entwickelt.

Contents

1 Introduction **6**

2 Background: Probabilistic Quantum Spins **9**
2.1 Quantum mechanics 9
2.2 Variational approximation 17
2.3 Probabilistic formulation of quantum spins 18

3 Method: Sampling and Learning with Neural Networks **22**
3.1 Neuron models 22
3.2 Generative neural networks 25
3.3 Spike-based sampling 29
3.4 Learning algorithms 35

4 Towards Neuromorphic Quantum State Tomography **39**
4.1 Learning target states - the brute force method 39
4.2 Results for state representation 43
4.3 Discussion 55

5 Searching and Finding Ground States **56**
5.1 An energy minimization algorithm for BrainScaleS-2 56
5.2 Results for the TFIM 61
5.3 Discussion 67

6 Limitations and Prospects **69**
6.1 Limitations of the BrainScaleS-2 chip 69
6.2 Exploiting marginal POVM distributions 77

7 Conclusion & Outlook **82**

A Lists **84**

B Parameter settings **86**

C Acronyms **87**

D Acknowledgments **89**

E Bibliography **90**

1 Introduction

The Hilbert space of quantum many-body systems and consequently the computational resources required to describe them grow exponentially with the system sizes making their simulation extremely difficult. One example is the modeling of large spin systems which promises fundamental insights into strongly correlated condensed matter and is highly relevant for the characterization and validation of digital quantum computers [SAM10, AM12, Pre18]. Fortunately, physical systems also exhibit symmetries and structure that can shrink the exponential complexity and allow tractable approaches for their description. The process of automatically discovering these patterns despite the curse of dimensionality is a discipline of machine learning (ML) which in recent years has found its way into quantum many-body physics [Car20]. The prevalent approach is the use of generative models based on artificial neural networks (ANN) for representing wave functions or density matrices and their underlying physical properties as Neural Quantum States (NQS).

In order to represent quantum states, i.e. complex-valued probability amplitudes, with ANNs a multitude of approaches have been proposed. Among them the restricted Boltzmann machine (RBM) [Hin07], a stochastic recurrent neural network (RNN) inspired by statistical mechanics, has been extensively studied. Given access to complex-valued parameters, RBMs can be directly trained to represent wave function coefficients [CT17]. Alternatively, two separate networks with real parameters can be used for the phase and amplitude components respectively [TMC+18]. For the approximation of mixed states represented as density matrices an RBM ansatz based on purification has been proposed [TMC+18]. A conceptually different approach directly translates density matrices or wave functions into probability distributions, for example by the use of positive operator-valued measurements (POVM) [CTMA19], which enables the use of real-valued RBMs as well as other kinds of generative models.

NQS have shown to be efficient function approximators that rival competing techniques like tensor networks in flexibility and scalability by providing accurate and compressed state descriptions using only a small number of parameters [CT17, CCX+18]. Among other applications, NQS have been successfully employed as variational ansätze for variational groundstate search (VGS) [CT17, BSD20, VGLH21], quantum dynamics [CT17, CGG18, LCCC20, RSG21], and quantum state tomography (QST) [TMC+18, SGA+19, CTMA19, PKB+20, AMNK20].

Arguably, these advances in numerical techniques have ultimately been enabled by the ubiquitous availability of cheap digital computers due to Moore's law which for the

past five decades predicted correctly that the number of transistors in a microchip doubles every two years [Moo98]. Scientific computing, including quantum many-body simulation, takes place overwhelmingly on those digital computers which are designed according to the *von Neumann architecture* where a central processing unit (CPU) communicates with a global memory unit. When simulating physics on a von Neumann computer the physical system is digitally represented in memory such that modifications and in particular interactions are applied indirectly by the CPU.

Today Moore's law is drawing to an end as transistor sizes approach the nanometer scale and development costs become infeasible [KHF18]. As a result, the interest in alternative, domain specific computing devices has risen as those could finally outperform digital computers in the long-term. In the light of modern machine learning and neuroscience the computations performed by neural networks are one such domain which the field of neuromorphic engineering aims to capture in hardware [SPP+]. There are numerous approaches for building neuromorphic computers both in terms of the level of abstraction at which they mimic neural circuitry and the specific hardware substrate that is employed for that purpose. Unlike general purpose computers, the core aspect of neuromorphic designs is that neurons are physically placed on the hardware and their functionality is inherently realized, whether in analog or digital form, using application specific integrated circuits (ASIC), memristors or photonics [MMQG20].

In view of the success of NQS on general purpose computers, in this work we investigate the utility that current neuromorphic hardware has to offer for solving quantum many-body problems. To that end we employ the BrainScaleS-2 (BSS2) chip [SBDW20] which is a neuromorphic chip developed by the Electronic Vision(s) group at Universität Heidelberg. The BrainScaleS architecture covers a unique point in the neuromorphic design space: it emulates biologically-inspired spiking neural networks (SNN) consisting of leaky integrate-and-fire (LIF) neurons with continuous-time analog circuits enabling an acceleration factor of 1000 compared to biological timescales [SBG+10].

SNNs mimic the event-based neural communication found in brains with dynamical integrate-and-fire neuron models such that time is explicitly harnessed as computational resource. Loosely speaking, each neuron has an internal potential variable whose value is the sum of postsynaptic potentials (PSP) which are caused by receiving spikes from afferent neurons. In turn, a neuron *fires* a spike when it has accumulated a certain potential, called the *activation threshold*. Importantly, all spike signals are of identical binary form in the sense that they only carry information about their spike times. As a result, SNNs operate on sparse spike trains such that information processed by them needs to be encoded to and decoded from this format. By integrating over the time dimension the static behavior of ANNs can be recovered in what is known as *rate coding*, i.e. where the frequency of spikes is interpreted as a real number. On the other hand, spiking neurons can be more expressive than artificial ones when they encode information temporally

in relative spike timings, which is hence called *temporal coding* [Maa97]. Subjecting a LIF SNN to external Poisson-distributed noise, it was shown that it is able to perform probabilistic inference by sampling from a discrete space akin to Boltzmann machines (BM) [PBB+]. One can understand this *neural sampling* scheme as a compromise between rate and temporal codes, where the collective firing rates of neuron combinations are interpreted as a probability distribution.

Due to the recent success of (R)BMs in quantum many-body physics, it is natural to establish a connection to neuromorphic implementations based on neural sampling. This line of research was pioneered recently by Czischek et al. [CBB+21] who demonstrated the viability to learn and represent small entangled quantum states in a POVM representation through neural sampling on BSS2. This thesis continues these efforts as follows. Chapter 2 introduces the background on how quantum spin systems can be represented probabilistically and chapter 3 details the simulation and emulation methods based on sampling and learning with BMs, including their implementation on neuromorphic hardware. Chapter 4 presents results on learning neuromorphic quantum state representations on BSS2, including for the first time for ground states and steady states. In chapter 5 a variational learning algorithm specific to the demands of BSS2 was developed for the search of ground states and it was applied to the Transverse Field Ising Model (TFIM) with system sizes of up to 10 spins. Limitations and prospects of the presented approaches are examined in chapter 6. We conclude in chapter 7 by summarizing and offering perspectives for future research.

2 Background: Probabilistic Quantum Spins

This chapter gives a brief overview of quantum mechanics and spin systems based on [BFK+15, NC10, PF10, Czi20]. In addition, two concrete problems in quantum many-body systems, namely VGS and QST, will be explained. Furthermore, the two probabilistic quantum state representations used throughout this thesis will be introduced.

2.1 Quantum mechanics

Quantum mechanics (QM) is the mathematical description of non-relativistic physical systems in terms of states, observables and how they evolve in time. The state space of QM is a complex Hilbert space $\mathcal{H}$, i.e. a complete vector space over $\mathbb{C}$ with a scalar product. States are vectors $|\psi\rangle \in \mathcal{H}$ in Hilbert space and observables are Hermitian operators $O : \mathcal{H} \to \mathcal{H}$ acting therein. In Dirac's notation $|\psi\rangle$ is called a "ket", which represents a column vector, while $\langle\psi| \in \mathcal{H}^\star$, is the dual row vector called a "bra".

The joint state space of two systems A and B is given by the tensor product $\mathcal{H}_{AB} = \mathcal{H}_A \otimes \mathcal{H}_B$.

This thesis deals exclusively with finite-dimensional quantum systems. Consequently, the states of these systems are spanned by bases with finitely many elements and operators are represented by matrices. One distinguishes between closed and open quantum systems depending on whether interactions with an environment are included.

Pure states and closed systems

A closed quantum system is completely described by the state vector $|\psi\rangle$ and is thus expressed as a superposition of $d_\mathcal{H} = \dim \mathcal{H}$ basis states

$$|\psi\rangle = \sum_{v=1}^{d_\mathcal{H}} c_v |v\rangle \tag{2.1}$$

where we assume an orthonormal basis $\{|v\rangle\}$. This representation is known as a *pure state* and the complex coefficients $c_v = \langle v|\psi\rangle \in \mathbb{C}$ are also called the *wave function* of the state.

The Born rule axiomatically connects the wave function coefficients to probabilities of finding the system in the corresponding basis state $p(v) = \langle\psi|v\rangle\langle v|\psi\rangle = |c_v|^2$. Due

to normalization of this probability distribution, a pure state must be a vector of unit length: $\langle\psi|\psi\rangle = 1$.

As a further consequence the time evolution of a pure state must preserve probability. Therefore, only *unitary* time evolution operators can be considered. The Schrödinger equation generates a continuous unitary time evolution through the Hamiltonian operator H which is the observable describing the system's total energy: $-i\hbar\partial_t|\psi\rangle = H|\psi\rangle$ where $\hbar$ is Planck's constant[1].

Due to the inherent probabilistic nature of QM, physical quantities are characterized in the form of expected values of observables. In the general case, ${d_\mathcal{H}}^2$ matrix elements of the observable operator have to be evaluated with respect to the chosen basis in order to calculate the expected value:

$$\langle O\rangle_\psi = \langle\psi|O|\psi\rangle = \sum_{v,v'} c_v^\star c_{v'} \langle v|O|v'\rangle \tag{2.2}$$

Mixed states and open systems

A problem with pure states is that they do not allow predictions expressing classical uncertainty about the quantum state. Suppose a system consists of an ensemble of distinct pure states $|\psi_i\rangle$ where each of the possible states is present with probability p_i. This system can be summarized concisely by the so-called *density matrix*:

$$\rho = \sum_i p_i |\psi_i\rangle\langle\psi_i| \tag{2.3}$$

Evidently, the density matrix is positive, Hermitian and has unit trace $\operatorname{Tr}\rho = 1$ since $\sum_i p_i = 1$. As it represents mixtures of pure states, the whole construct is called a *mixed state*. It of course includes pure states, in which case holds $\rho^2 = \rho = |\psi\rangle\langle\psi|$.

The expected value of O in state ρ also becomes a sum over mixture constituents which can be abbreviated using the trace:

$$\langle O\rangle_\rho = \sum_i p_i \langle\psi_i|O|\psi_i\rangle = \operatorname{Tr}\rho O \tag{2.4}$$

The Schrödinger equation can be translated into the density matrix formalism and is then called the von Neumann equation:

$$\frac{d\rho}{dt} = -i\,[H,\rho] \tag{2.5}$$

It describes the unitary time evolution of the system without the exchange of energy or particles with the environment.

[1] We will set $\hbar = 1$ in later chapters.

The description of open quantum systems (OQS), i.e. systems where an exchange with the environment takes place, is an important application for mixed states. The total system is composed of the Hilbert space of the system of physical interest and the Hilbert space of the environment: $\mathcal{H}_{\text{tot}} = \mathcal{H}_{\text{s}} \otimes \mathcal{H}_{\text{env}}$. A description of the system is then possible under assumptions about the nature of the interaction with the environment by applying a partial trace over the environment degrees of freedom of the density matrix of the whole system: $\dot{\rho}_s = \text{Tr}_{\text{env}}\, \dot{\rho}_{\text{tot}}$.

The result are so-called Master equations. The *Lindblad equation* is one such equation describing the time evolution of the OQS under a unitary, Hamiltonian part and including dissipation through a set of jump operators Γ_k with associated decay rates γ_k:

$$\frac{d\rho}{dt} = \mathcal{L}\rho = -i\,[H, \rho] + \sum_k \frac{\gamma_k}{2}(2\Gamma_k^\dagger \rho \Gamma_k - \{\rho, \Gamma_k^\dagger \Gamma_k\}) \tag{2.6}$$

The joint dynamics are summarized into the system's Liouvillian super-operator $\mathcal{L}$ that acts in the space of density matrices.

Interacting quantum spins

This thesis is concerned with spin-1/2 particles, commonly referred to as spins or qubits. Since their spin quantum number is $s = \frac{1}{2}$ they can occupy one of the two states specified by the z-axis projection: $s_z \in \{-\frac{1}{2}, \frac{1}{2}\}$. Thus, the Hilbert space of a single spin has two basis vectors and is spanned by a set of operators consisting of the identity matrix $\mathbb{1}_2$ and the Pauli matrices $\boldsymbol{\sigma} = (\sigma_x, \sigma_y, \sigma_z)$. The spin operators in the respective spatial directions are proportional to the Pauli matrices $S_i = \frac{\hbar}{2}\sigma_i$.

We define the two eigenstates of σ_z with maximal z-axis projection as the standard z-basis[2] of $\mathcal{H}$: $|\uparrow\rangle = |0\rangle = (1,0)^T$ and $|\downarrow\rangle = |1\rangle = (0,1)^T$. A pure spin state is then characterized by two complex numbers α, β: $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$. However, states that differ only by a global phase factor are physically identical, thus we can choose α as a positive real number. With the additional constraint of normalization we arrive at a two parameter description $\theta \in [0, \pi]\,, \phi \in [0, 2\pi)$:

$$|\psi\rangle = \cos\left(\frac{\theta}{2}\right)|0\rangle + e^{i\phi}\sin\left(\frac{\theta}{2}\right)|1\rangle \tag{2.7}$$

Pure spin states thus live on a two-dimensional manifold of $\mathcal{H}$. Visualization in 3D by interpreting (θ, ϕ) as the angles of spherical coordinates $\boldsymbol{a} = (\sin\theta\cos\phi, \sin\theta\sin\phi, \cos\theta)$ shows that pure states form the enclosing surface of the *Bloch sphere* (see fig. 2.1).

[2] also known as the "computational basis" in quantum computing

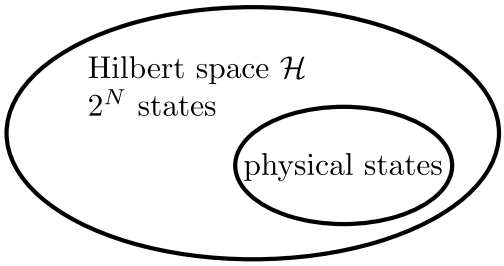


Figure 2.2: Schematic of Hilbert space and the subspace of relevant physical states.

Mixed qubit states are represented by a density matrix $\rho \in \mathbb{C}^{2\times 2}$ which can be expanded as $\rho = \frac{1}{2}(\mathbb{1} + \boldsymbol{a} \cdot \boldsymbol{\sigma})$ with coefficients $\boldsymbol{a} \in \mathbb{R}^3$ such that $|\boldsymbol{a}| \leq 1$. Furthermore, for $|\boldsymbol{a}| = 1$ the state is pure, which corresponds to the Bloch sphere representation. Therefore, all points with $|\boldsymbol{a}| < 1$ are mixed states which lie in the interior of the Bloch sphere.

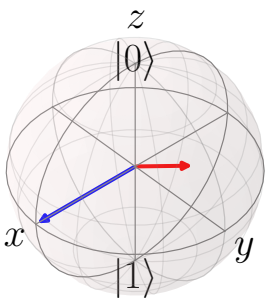


Figure 2.1: A pure (blue) and mixed (red) spin state on the Bloch sphere

The state space of a joint system of N qubits is exponentially large: dim $\mathcal{H} = 2^N$. In this case we can describe the joint basis with binary vectors $\boldsymbol{v} = (v_i)_{i=1}^N$ with $v_i \in \{0, 1\}$ and the corresponding basis states with the consecutive tensor products $|\boldsymbol{v}\rangle = \otimes_{i=1}^N |v_i\rangle$.

However, in physical systems, most of these states carry only a negligible probability due to certain structures e.g. low entanglement. Thus, the interesting region of the Hilbert space can form a rather small corner of the whole as depicted in fig. 2.2. Ideally, the physical subspace would scale polynomially with the system size. In this case there is hope of finding tractable approaches of simulating the quantum spin system.

The quantum Ising Model in one dimension, also known as the TFIM, will be used as a benchmark of the methods developed in this thesis. It describes a chain of N spins with nearest-neighbor interactions that are subject to an external field orthogonal to the interaction axis (see fig. 2.3). The Hamiltonian of the TFIM with interaction along the z-axis and transverse field in x-direction reads

$$H_{\text{TFIM}} = -J \sum_{\langle i,j \rangle} \sigma_z^i \sigma_z^j - h \sum_{i=1}^{N} \sigma_x^i \tag{2.8}$$

where J is the interaction strength, h is the field strength and $\langle i, j \rangle$ signifies nearest

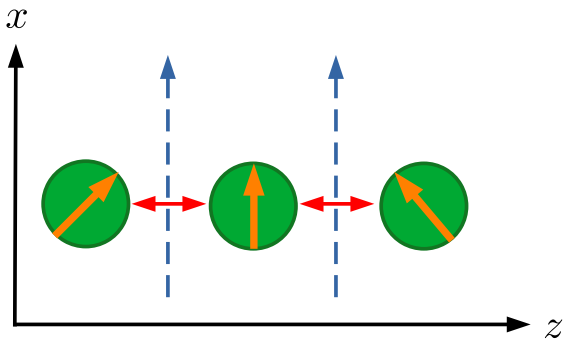


Figure 2.3: Visualization of the TFIM in one dimension. A chain of three spins (green) with open boundary conditions is shown along with their interactions in z-direction (red) and the applied external field in x-direction (blue). The orange arrows indicate expected values of the spin observables in the x-z-plane.

neighbor pairs. The σ^i are Pauli operators in the joint state space which only act on the single spin i: $\sigma^i = \otimes_k \sigma^{2-\delta_{k,i}}$. Typically, we will consider periodic boundary conditions such that there is an interaction between spin 1 and N. When $J > 0$ the model is called *ferromagnetic* since aligned spins lead to a lower energy, while $J < 0$ is the *antiferromagnetic* case where the opposite configuration is favored. We exclusively work with the ferromagnetic case.

Figure 2.4 shows the phase structure of the system. In the thermodynamic limit the TFIM has a quantum phase transition at the *critical point* $J = h$ which separates the ordered phase ($h < J$) where the energy is dominated by the spin-spin interactions $\sigma_z^i \sigma_z^{i+1}$ from the disordered phase ($h > J$) where spins increasingly align with the x-axis due to the influence of the external field σ_x^i.

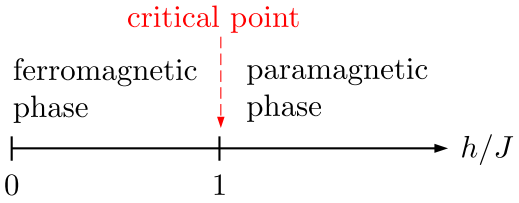


Figure 2.4: Phase diagram of the TFIM

Thus, the two relevant observables for the phase transition are the magnetization in x-direction,

$$\langle \sigma_x \rangle = \frac{\sum_i \langle \sigma_x^i \rangle}{N} \tag{2.9}$$

which is also the order parameter of the transition and the two-point zz-correlation function

$$C_{zz}(d) = \frac{\sum_i \langle \sigma_z^i \sigma_z^{i+d} \rangle}{N} \tag{2.10}$$

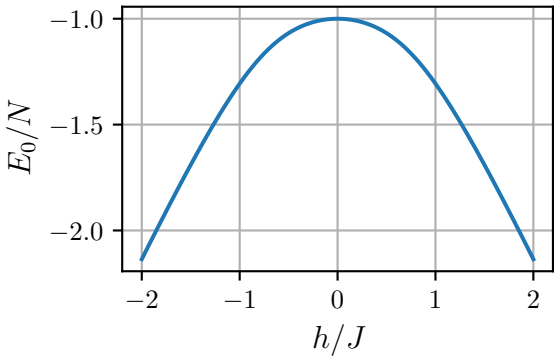


Figure 2.5: Ground state energy per site of the TFIM as a function of h/J for $N = 4$.

where d is the distance between spins.

In the vicinity of the critical point the spin-spin correlations are expected to drop off as a power law [KCH+17]: $C_{zz}(d) \simeq C_0(h) \exp(-d/\xi_{zz}(h))$. At zero-temperature the correlation length ξ_{zz} diverges at the critical point indicating the phase transition point. For small, finite system sizes the phase transition point will be shifted and the correlation lengths stay finite, but become maximal there.

The TFIM is integrable [Pfe70] such that analytical solutions for the energy spectrum and eigenstates are available. The ground state energy for finite system sizes is given by

$$E_0 = -\sum_{k \in K} \sqrt{1 + \frac{h}{J}^2 + \frac{2h}{J} \cos\left(\pi \frac{k}{N}\right)} \text{ with } K = \{-(N-1), ..., N-1\} \quad (2.11)$$

Figure 2.5 shows the ground state energy as a function of h/J for $N = 5$ spins.

Stoquastic Hamiltonians

A Hamiltonian where all off-diagonal elements are non-positive is called *stoquastic* [BDOT07]. From this property one can show that the ground state of the system is described by a wave function which only has non-negative real amplitudes.

Many systems occurring in nature have stoquastic Hamiltonians, including the TFIM, the ferromagnetic Heisenberg model and also continuous systems like the quantum harmonic oscillator.

A major advantage when simulating systems with stoquastic Hamiltonians is that they avoid the sign problem [Bra15]. It arises whenever a state has negative or complex probability amplitudes and causes Monte Carlo methods to require exponentially many samples to accurately estimate observables.

Measurement

We have already pointed out the close connection between quantum states and probabilities based on the Born rule. Now, we take a closer look at two types of measurements that can be performed, namely projective measurements and POVMs.

A projective measurement consists of a set of projection operators $P_i = |i\rangle\langle i|$ that are constructed from an orthonormal basis $\{|i\rangle\}$. Applying this measurement to the state ρ randomly obtains an outcome i according to the probability distribution

$$p(i) = \mathrm{Tr}\, P_i \rho \tag{2.12}$$

and projects the state into the respective subspace yielding $\rho' = P_i \rho P_i / \mathrm{Tr}(P_i \rho)$ [3].

POVMs drop the requirement of measurement operators being orthogonal projectors and are thus more general. A set of positive-semidefinite operators $\{M^a\}_a$ is called a POVM if they decompose the identity: $\sum_a M^a = \mathbb{1}$. This ensures that the respective probability distribution for a state ρ remains normalized:

$$p(a) = \mathrm{Tr}(\rho M^a) \tag{2.13}$$

Notice the equivalence to a projective measurement when replacing M^a with $|a\rangle\langle a|$.

Since POVM elements do not need to be orthogonal their so-called overlap matrix $T_{a,a'} = \mathrm{Tr}(M^a M^{a'})$ is not the identity.

A POVM is *informationally complete (IC)* if the measurement distribution includes all information about the state, i.e. it has at least $d_{\mathcal{H}}^2$ elements that span the space of Hermitian operators. This is the case if the overlap matrix is non-singular, since it allows for eq. (2.13) to be inverted, yielding the full density matrix:

$$\rho = \sum_{a,a'} T^{-1}_{a,a'} M^{a'} p(a) \tag{2.14}$$

Starting from a single-qubit IC-POVM with m outcomes $\{M^a\}_{a=1}^m$, one can construct an N-qubit version with m^N outcomes by multiplying the single-qubit elements $\{M^{\boldsymbol{a}}\}_{\boldsymbol{a}} = \{M^{a_1} \otimes ... \otimes M^{a_n}\}_{a_1,\cdots,a_N}$. The collective POVM outcomes can then be summarized into a vector $\boldsymbol{a} = (a_1, \cdots, a_N)$. Similarly, the overlap matrix and its inverse can be factorized into their single spin components.

In this formalism, expected values of observables turn into probabilistic expected values

$$\langle O \rangle = \mathrm{Tr}(\rho O) = \sum_{\boldsymbol{a}} Q_O^{\boldsymbol{a}} p(\boldsymbol{a}) = \langle Q_O^a \rangle_{p(\boldsymbol{a})} \tag{2.15}$$

with coefficients

$$Q_O^{\boldsymbol{a}} = \sum_{\boldsymbol{a}} \mathrm{Tr}(O M^{\boldsymbol{a}}) T^{-1} \tag{2.16}$$

[3] In the pure state case the expressions are $p(i) = \langle\psi|P_i|\psi\rangle = |\langle i|\psi\rangle|^2$ and $|\psi'\rangle = P_i|\psi\rangle / \sqrt{p(i)} = |i\rangle\langle i|$

Note that the computation of these coefficients still requires summing an exponential number of terms. However, for local operators one need only consider the marginal distribution of the involved spins and their respective POVM outcomes a_i.

Every IC-POVM $\{M^{\boldsymbol{a}}\}_{\boldsymbol{a}}$ has a dual $\{N^{\boldsymbol{b}}\}_{\boldsymbol{b}}$ with $N^{\boldsymbol{b}} = \sum_{\boldsymbol{a}} T^{-1}_{\boldsymbol{a},\boldsymbol{b}} M^{\boldsymbol{a}}$ such that

$$\operatorname{Tr} M^{\boldsymbol{a}} N^{\boldsymbol{b}} = \delta_{\boldsymbol{a},\boldsymbol{b}} \tag{2.17}$$

Distance measures for quantum states

The *quantum fidelity* quantifies the similarity of two quantum states. In the general case of two mixed states ρ and σ it is given as

$$F(\rho, \sigma) = \operatorname{Tr} \sqrt{\sqrt{\sigma}\rho\sqrt{\sigma}} \tag{2.18}$$

In the case of a mixed state ρ and a pure state $|\psi\rangle$ this expression becomes $F(\rho, |\psi\rangle) = |\langle\psi|\rho|\psi\rangle|$ and for two pure states $|\psi\rangle$ and $|\phi\rangle$ simplifies to the wave function overlap $F(|\psi\rangle, |\phi\rangle) = |\langle\psi|\phi\rangle|$.

The quantum fidelity puts an upper bound on all observable differences which is why it can be used as an universal indicator for how well observables agree between the given states.

In turn, for an arbitrary POVM $\{M^a\}_a$, the quantum fidelity between ρ and σ is bounded from above by the *classical fidelity* between the corresponding probability distributions:

$$F_c(p_\rho, p_\sigma) = \sum_a \sqrt{p_\rho(a) p_\sigma(a)} \geq F(\rho, \sigma) \tag{2.19}$$

When expressing two quantum states in terms of probability distributions p and q their similarity can also be measured using the Kullback-Leibler divergence (DKL).

The DKL can be viewed as a measure of distance, since it is zero when $p = q$ and it also has no upper bound. It is defined as the difference between the cross-entropy $X(p, q) = -\sum_i p_i \log q_i$ and the Shannon entropy $S(p) = -\sum_i p_i \log p_i$:

$$D_{KL}(p\|q) = X(p, q) - S(p) = \sum_i p_i \log \frac{p_i}{q_i} \in [0, \infty) \tag{2.20}$$

Thus, intuitively the DKL describes the entropy of distribution q relative to the reference p. Since one of the distributions serves as the reference the DKL is not a metric. If one needs a symmetric comparison, the classical fidelity can serve as a measure of similarity (see above).

2.2 Variational approximation

Due to the exponential size of the Hilbert space we need numerical approximations to describe and simulate large quantum systems. A general feature of these methods is the use of variation over some parameterized function space which represents the wave function or density matrix, respectively. The variational search must be guided by a suitable application-specific objective function. In this section we look at two domains where variational approximations can be utilized: ground state search and quantum state tomography.

2.2.1 Variational ground state search

The ground state is of particular interest because it describes the behavior of the quantum spin system at zero temperature where quantum effects are most pronounced.

In QM VGS aims to find the ground state of a given Hamiltonian by varying the parameters θ of a trial wave function $|\psi_\theta\rangle = \sum_{\boldsymbol{v}} c_{\theta,\boldsymbol{v}}|\boldsymbol{v}\rangle$ (which need not be normalized).

Since the spectrum of the Hamiltonian is bounded from below by the ground state energy E_0 any state that is not the ground state must yield a higher expected value:

$$E_\theta = \frac{\langle\psi_\theta|H|\psi_\theta\rangle}{\langle\psi_\theta|\psi_\theta\rangle} \geq E_0 \tag{2.21}$$

For large system sizes the computation of E_θ and derived quantities becomes infeasible since an exponential number of terms has to be summed. Variational Monte Carlo (VMC) addresses this issue by rewriting the variational energy as an average over a set of sample configurations S:

$$E_\theta = \sum_{\boldsymbol{v},\boldsymbol{w}} |c_{\theta,\boldsymbol{v}}|^2 \frac{c_{\theta,\boldsymbol{w}}}{c_{\theta,\boldsymbol{v}}} \langle\boldsymbol{v}|H|\boldsymbol{w}\rangle = \sum_{\boldsymbol{v}} p_\theta(\boldsymbol{v}) E^{\text{loc}}_{\boldsymbol{v}} \approx \frac{1}{|S|} \sum_{\boldsymbol{v_s}\in S} E^{\text{loc}}_{\boldsymbol{v_s}} \tag{2.22}$$

where $p_\theta(\boldsymbol{v})$ is shorthand for the measurement distribution and $E^{\text{loc}}_{\boldsymbol{v}} = \sum_{\boldsymbol{w}} c_{\theta,\boldsymbol{w}} H_{\boldsymbol{v},\boldsymbol{w}} / c_{\theta,\boldsymbol{v}}$ is the *local energy*. Note that one must be able to efficiently compute $c_{\theta,\boldsymbol{v}}$ for a given configuration $\boldsymbol{v}$ to succeed with this method.

In order to tune the variational parameters any kind of optimization algorithm can in principle be used. In section 3.4 we will have a closer look at the ones used in this thesis.

The simplest type of variational ansatz is the mean field approximation, where a separable wave function $|\psi_\theta\rangle = \otimes_i |\psi_i\rangle$ is assumed. While this reduces the number of required parameters to linear complexity in the system size, it of course fails once interaction terms start to dominate. More sophisticated ansätze need to be able to capture

higher-order correlations, ideally with a number of parameters that scale polynomially in the system size.

Current work on VMC in spin systems focuses on two promising families of trial wave functions: tensor networks [BC17, Orú19], such as Matrix Product States (MPS) [PVWC07] and ANNs such as RBMs [CT17] and RNNs [HIW+21].

2.2.2 Quantum state tomography

QST is the process of reconstructing a quantum state from measurement statistics on identically prepared systems. That is, it allows to reverse the act of measurement given sufficient data about the outcomes.

To extract the full information content of a density matrix ρ a measurement with $d_{\mathcal{H}}^2$ outcomes a would have to be carried out producing a distribution $p(a)$ according to eq. (2.13). In case of a system of N spins, the number of measurements required scales exponentially with 4^N which is infeasible to carry out in practice.

One proposal to alleviate this issue is to leverage a variational ansatz for the quantum state ρ_θ, or for an associated informationally complete probability distribution p_θ, whose number of parameters required to faithfully capture observables only grows polynomially with the system size. The parameters have to be optimized such that the ansatz fits the limited observed data while encoding a close approximation to the true state. With a probabilistic description p_θ this can be done by minimizing a distance measure such as the DKL between the measured distribution and the variational one. After the optimization more synthetic data can be generated by sampling from p_θ in order to estimate observable expected values.

Modern generative methods from ML are particularly well suited for this task since they are extremely expressive while allowing for efficient sample generation. For example, recent work has investigated the suitability of RBMs [TMC+18, TM18, NFJ+20], RNNs [CTMA19] and GANs [AMNK20] in the context of QST.

2.3 Probabilistic formulation of quantum spins

In order to be able to use the generative methods from ML already mentioned, we have to bring the quantum systems we are interested in into a compatible representation. This representation is a probabilistic one.

The tetrahedral POVM

A complete description of a quantum state in terms of a probability distribution can be obtained by measuring it in a sufficient amount of bases. For example, to do QST and reconstruct a density matrix ρ with full rank, we can perform an IC-POVM

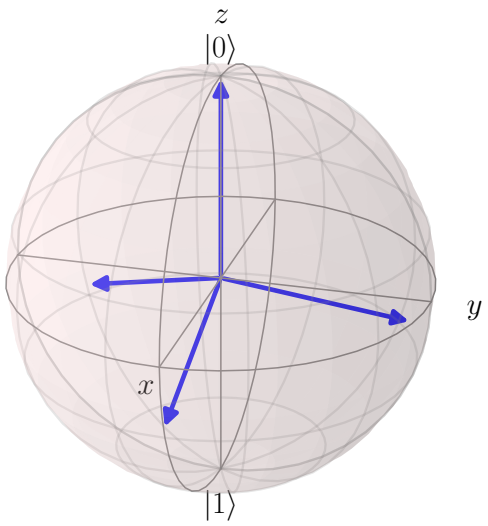


Figure 2.6: The elements of the tetrahedral POVM visualized on the Bloch sphere.

measurement. Conversely, we can propose a model probability distribution $p_\theta(a)$ with parameters θ, and declare it to describe a quantum state through the IC-POVM.

In this work, we have limited ourselves to the tetrahedral POVM. Its single spin elements are four projectors spanning a tetrahedron on the Bloch sphere:

$$M^a_{\text{tetra}} = \frac{1}{4}(\mathbb{I} + \boldsymbol{s}_a \cdot \boldsymbol{\sigma}) \tag{2.23}$$

where $a \in \{0,1,2,3\}$ and the directions $\boldsymbol{s}_0 = (0,0,1)$, $\boldsymbol{s}_1 = (2\sqrt{2}/3, 0, -1/3)$, $\boldsymbol{s}_2 = (-\sqrt{2}/3, \sqrt{2/3}, -1/3))$, $\boldsymbol{s}_3 = (-\sqrt{2}/3, -\sqrt{2/3}, -1/3)$ are visualized in fig. 2.6. The resulting overlap matrix and its inverse are

$$T_{\text{tetra}} = \frac{1}{4}\begin{bmatrix} 1 & 1/3 & 1/3 & 1/3 \\ 1/3 & 1 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1 & 1/3 \\ 1/3 & 1/3 & 1/3 & 1 \end{bmatrix} \rightarrow T^{-1}_{\text{tetra}} = \begin{bmatrix} 5 & -1 & -1 & -1 \\ -1 & 5 & -1 & -1 \\ -1 & -1 & 5 & -1 \\ -1 & -1 & -1 & 5 \end{bmatrix} \tag{2.24}$$

Since all off-diagonal entries of the overlap matrix are equal the tetrahedral POVM is also called *symmetric* and thus belongs to the class of *SIC-POVMs*.

Figure 2.7 shows the POVM distribution that is obtained for the Bell state $|\psi\rangle = (|00\rangle + |11\rangle)/\sqrt{2}$ according to eq. (2.13) with the 2-spin tetrahedral POVM.

A disadvantage of parameterizing the distribution $p(a) \equiv p_\theta(a)$ is that there is no positivity constraint on the corresponding density matrix. The reason is that while IC-POVMs preserve the unit trace property this can be accomplished with negative eigenvalues and therefore not every probability distribution represents a valid density matrix.

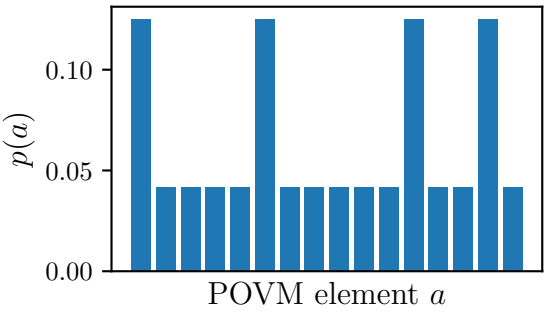


Figure 2.7: Standard tetrahedral POVM distribution of the Bell state $|\psi\rangle = (|00\rangle + |11\rangle)/\sqrt{2}$ where the x-axis enumerates the $4^2 = 16$ elements $\boldsymbol{a} = (a_1, a_2)$.

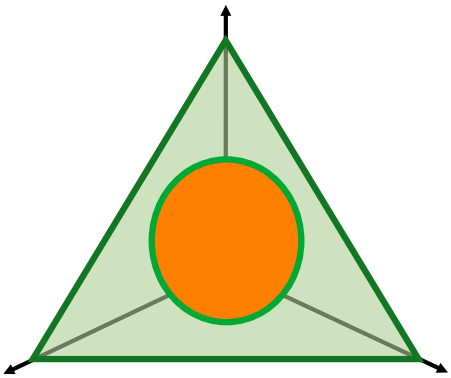


Figure 2.8: Schematic of the Qplex (round object) within the probability simplex (triangular object). The green boundary of the Qplex corresponds to the set of all pure states that form the surface of the orange interior set of mixed states.

This is visualized in fig. 2.8 where the probability space of physical states, termed *Qplex* [FMS14, AFSZ17] is a convex subset inside the general probability simplex. One can show that the Qplex is encapsulated by the density matrices corresponding to pure states while its interior volume is filled by mixed states.

z-basis representation

The POVM-based state representation is not suitable for ground state search through energy minimization due to the possibility of finding non-physical probability distributions with increasingly large negative eigenvalues.

We instead use the z-basis in conjunction with the assumption of a stoquastic Hamiltonian. In this case we know that the ground state we are looking for will be expressible using non-negative real wave function coefficients only. Therefore a z-basis measurement constitutes an informationally complete description since the full wave function is

computable as the square root of the probability distribution:

$$c_{\theta,v} = \sqrt{p_\theta(v)} \tag{2.25}$$

The measurement outcomes v simply correspond to the $d_{\mathcal{H}} = 2^N$ basis spin configurations $\{|v\rangle\}_{v=1}^{d_{\mathcal{H}}}$. Note that it is the prior knowledge about the ground state positivity that shrinks the number of degrees of freedom by a factor of two compared to a wave function with complex coefficients.

3 Method: Sampling and Learning with Neural Networks

This chapter describes our specific method based on neural networks for implementing and optimizing the probabilistic quantum state descriptions introduced above. We first discuss various neuron models before describing the Boltzmann machine, the generative neural network used throughout the thesis, and the Gibbs sampling algorithm. Spike-based sampling is introduced as an alternative framework for performing probabilistic inference with spiking neurons and the technical implementation details of LIF sampling for BSS2 are listed. Finally, the optimization - or learning - algorithms are described.

3.1 Neuron models

This section describes neurons, the computational units of neural networks, at various levels of abstraction from biology, to perceptrons, to spiking neurons and the LIF model (based on [GWKM02]).

Biological neurons

Let us briefly look at the features of biological neural tissue that are crucial for neural computation. Neurons are nerve cells and as such they are encapsulated by a thin cell membrane. As shown in fig. 3.1, the neuron's cell body (soma) connects the dendrites which aggregate inputs and the axon which propagates signals via its axon terminals. A synapse describes a gap of around $20\,\mathrm{nm}$ by which spikes from the *presynaptic* axon terminal are transmitted to the *postsynaptic* dendrite.

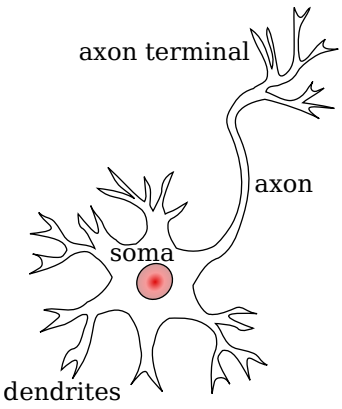


Figure 3.1: Sketch of a biological neuron. Image adapted from [Haa12]

There is a voltage across the cell membrane which is caused by the difference in ion concentrations inside and outside of the neuron. This membrane voltage remembers the influence of external stimuli and is the key variable that determines the spiking dynamics of the neuron. When the neuron receives only few stimuli, it will quickly decay to an equilibrium or resting potential which in humans is around

$-72\,\mathrm{mV}$. On the other hand, when enough input has been received the membrane voltage suddenly increases (depolarization) triggering the *action potential* also known as a *spike*. After a neuron has spiked its membrane voltage drops quickly and stays lower than the resting potential (hyperpolarization) for a short period of time during which it is *refractory* to further input.

The result of spike transmission to a receiving neuron is the induction of a *PSP*. This can be either of *excitatory* or *inhibitory* nature depending on whether its spiking probability is increased or decreased.

Artificial neurons

Artificial neurons - also called perceptrons - resemble biology in the way that they also integrate stimulus and subject this signal to a nonlinear transformation. Figure 3.2a depicts this process which can be mathematically described as accumulation of the constant input vector $\boldsymbol{x}$ by a linear weighting (including an offset term) $u = \sum_{k=1}^{n_{\text{in}}} w_k x_k + b$ and subsequent application of a non-linear function: $a = f(u)$. The final result of the computation is called the *activation* of the neuron which encodes information as a fixed number or rate.

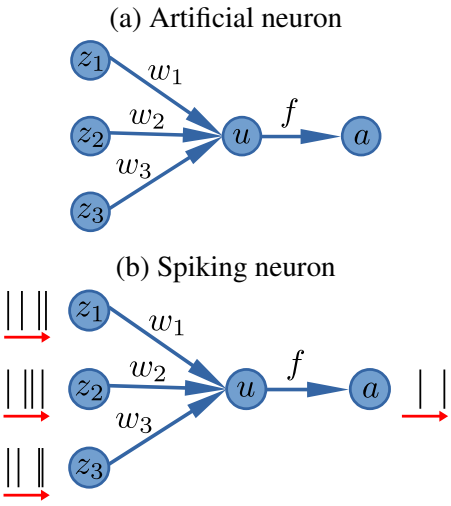


Figure 3.2: Comparing the artificial and spiking neuron

ANNs are highly non-linear functions built out of these primitive computational units. The connectivity of the ANN is a choice dependent on the application, however, the typical structure involves at least a dedicated set of input and output neurons which serve as the interface with the environment and sets of hidden neurons which encode latent representations. The simplest kind of ANN has only feedforward connections such that the neural computation is unidirectional and structured in layers that successively transform the input signal into the output activation via the hidden layers.

The number of hidden units of an ANN is a measure of the model complexity and governs its expressiveness. ANNs with at least 1 hidden layer are universal function approximators which means that they can compute any real function between input and output to arbitrary precision given a number of hidden units that is sufficiently large [Hor91].

Spiking neurons

A major step from artificial neurons towards a biologically realistic neuron model

is the introduction of *spiking dynamics*. Spiking neuron models not only integrate their input in space, namely from their presynaptic partners, but also in time. At the time t of arrival inbound spikes are imprinted with the corresponding synaptic weight onto the dynamical variable u of the receiving neuron. A firing criterion $f(u)$, which can be deterministic or stochastic, evaluates whether the neuron generates a spike of its own. This behavior is illustrated schematically in fig. 3.2b where the red arrows represent the spike arrival and generation over time.

Crucially, the variable u serves as a memory of past spike inputs which enables the relative timing of spikes to encode information. These temporal correlations of spikes enhance the expressiveness of SNNs compared to equally sized ANNs [Maa97]. From this perspective ANNs can be viewed as a special case when fixed input values are encoded as the rates of spike trains.

Leaky-integrate-and-fire neuron

The LIF neuron is a concrete case of a continuous spiking neuron model [GWKM02].

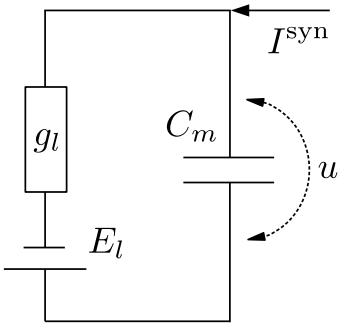


Figure 3.3: Equivalent circuit of the LIF neuron

In the LIF model the neuron's membrane is modeled as a capacitor with capacitance C_m. It can be charged by synaptic current stimulus $I^{\text{syn}}(t)$ while it is constantly discharged across a leak conductance g_l. These properties can be equivalently represented by the RC-circuit shown in fig. 3.3.

According to Kirchhoff's laws the voltage u across the capacitance is described by the differential equation:

$$C_m \frac{\mathrm{d}u(t)}{\mathrm{d}t} = g_l(E_l - u(t)) + I^{\text{syn}}(t) \tag{3.1}$$

The potential E_l plays the role of the resting state which is, in the absence of external input, approached on the time scale of the circuit $\tau_m = C_m/g_l$. For convenience one often rewrites this equation as $\tau_{\text{eff}}\partial_t u = u_{\text{eff}} - u$ with an effective membrane potential $u_{\text{eff}}(t) = E_l + I^{\text{syn}}(t)$.

The spike mechanism is triggered deterministically when the membrane potential crosses a threshold u_T from below:

$$u(t_{\text{spike}}) = u_T \wedge u'(t_{\text{spike}}) > 0 \tag{3.2}$$

After the spike has been fired, the membrane potential is clamped to a reset value during the absolute refractory period τ_{ref}:

$$u(t_{\text{spike}} \leq t \leq t_{\text{spike}} + \tau_{\text{ref}}) = u_R \tag{3.3}$$

In this thesis we only dealt with current-based synapses in which case synaptic weights carry the unit of current. Thus, spike input can be written as a convolution of the synaptic kernel $\kappa(t)$ and the spike trains of presynaptic neurons $S^i(t) = \sum_{t_s} \delta(t - t_s^i)$:

$$I^{\text{syn}}(t) = \sum_i w_i (S^i \star \kappa)(t) = \sum_i w_i \sum_{t_s^i} \kappa(t - t_s) \tag{3.4}$$

We use the exponential kernel $\kappa(t) = \Theta(t) \exp(-t/\tau_{\text{syn}})$ where τ_{syn} describes the time scale during which the synaptic impact is remembered. The PSP of a spike at time t_s on the membrane potential is analytically determined as a difference-of-exponentials [Pet16]:

$$PSP(t) = \frac{\tau_{\text{syn}}\tau_m}{C_m(\tau_{\text{syn}} - \tau_m)} \Theta(t - t_s) \left[\exp\left(-\frac{t - t_s}{\tau_{\text{syn}}} \right) - \exp\left(-\frac{t - t_s}{\tau_m} \right) \right] \tag{3.5}$$

3.2 Generative neural networks

Let us briefly introduce the concept of generative models before discussing a specific instance based on neural networks.

The basis of ML problems is the data set $X = \{\boldsymbol{x}_i\}_{i=1}^N$ where $\boldsymbol{x}_i$ are d-dimensional feature vectors assumed to be independent and identically distributed (i.i.d.) according to an unknown probability distribution $p(\boldsymbol{x})$. The goal of ML algorithms is to recognize patterns in these data to "train" a predictive model which compresses the high-dimensional structure of the training data and ideally generalizes, i.e. offers accurate predictions when applied to unseen data.

One type of prediction task is the creation of new data samples that are representative of those in the training set X. Models that are capable of this task are called *generative*. In other words, a generative model aims to capture the underlying probability distribution $p_{\text{model}}(\boldsymbol{x}) \approx p(\boldsymbol{x})$ *and* provide an efficient mechanism to draw samples from it $\boldsymbol{x}_{\text{new}} \sim p_{\text{model}}(\boldsymbol{x})$. Thus, there are two key ingredients that need to be met:

1. expressive and efficient latent representations: the model's internal degrees of freedom should provide sufficient capacity to capture high-dimensional feature correlations with a manageable amount of computational resources

2. high-quality and efficient generation: extracting new data should be fast and new samples ought to be uncorrelated among them and unbiased with respect to the ground truth (after training)

We will now discuss these considerations in the context of BMs.

3.2.1 The Boltzmann machine

The BM was introduced in [AHS85] where it is described as "parallel constraint satisfaction network [...] that is capable of learning the underlying constraints that characterize a domain simply by being shown examples from the domain".

It is inspired by the statistical mechanics of Ising's spin model [Isi25] which is why it also described as an *energy-based* model. However, the interaction parameters of the BM are considered variable with the goal of adjusting them such that the "equilibrium" state of the system exhibits patterns that closely match a given data set of configurations.

Thus, the BM is able to solve a probabilistic inference problem and it is in fact equivalent to undirected graphical models known from Bayesian statistics [HTF09].

A third interpretation of BMs as stochastic network models for neuronal activity connects them to the fields of cognitive science and ML. This connection will become especially clear when considering its implementation with a SNN.

A BM with N units occupies the binary configurations in $\boldsymbol{z} = \{z_i\}_{i=1}^N \in \{0,1\}^N$ [1] by defining the classical Hamiltonian

$$E_\theta(\boldsymbol{z}) = -\frac{1}{2}\sum_{ij} W_{ij} z_i z_j - \sum_i b_i z_i \tag{3.6}$$

with real interaction matrix $W \in \mathbb{R}^{N\times N}$ and bias vector $b \in \mathbb{R}^N$ summarized as the model parameters $\theta = (W, b)$ [2].

As in the Ising model connections are pairwise bidirectional such that $W_{ij} = W_{ji}$ and $W_{ii} = 0$ and therefore information can flow in *recurrent* fashion through the network.

In the statistical physics picture the equilibrium state of the BM is analogous to the canonical ensemble of N particles at temperature $T = 1$, where each particle can occupy two energy states. The probability of state $\boldsymbol{z}$ thus follows the well-known Boltzmann distribution with

$$p_\theta(\boldsymbol{z}) = \frac{1}{\mathcal{Z}} \exp\left(-E_\theta(\boldsymbol{z})\right) \tag{3.7}$$

where $\mathcal{Z} = \sum_{\boldsymbol{z}} \exp\left(-E(\boldsymbol{z})\right)$ is the partition sum. The *energy-based* nature is now apparent since low energy samples have the highest probability mass whereas high energy ones are exponentially less probable.

Thus far, the BM is an elaborate way of expressing the pairwise covariance between units. In order to create an expressive model that can also represent higher-order correlations the units are separated into two sets like in fig. 3.4 (left). The *visible* units

[1] An alternative state space uses $z \in \{-1, 1\}$ which is more natural when thinking in terms of spins. Both binary bases are linear transformations of each other and thus do not change the overall properties of the BM.

[2] Note that the bias can be eliminated by introducing an additional neuron with constant unit activity.

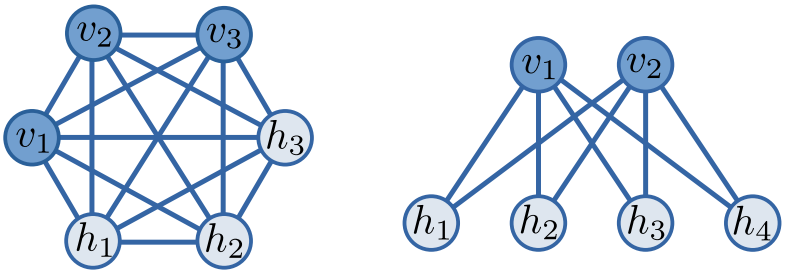


Figure 3.4: Left: fully-connected BM, Right: restricted BM

$\boldsymbol{v} = \{v_i\}_{i=1}^{N_v}$ are considered input and output of the network and thus, the amount of visibles must match the dimensionality of the data. Meanwhile the *hidden* units $\boldsymbol{h} = \{h_i\}_{i=1}^{N_h}$ are latent variables enhancing the complexity of the model. The number of parameters of the BM depends critically on N_h. We denote the joint state space $\boldsymbol{z} = [\boldsymbol{v}, \boldsymbol{h}]$ with $N = N_v + N_h$.

With this subdivision, the physically interesting model distribution over the visible units is obtained by marginalization:

$$p_\theta(\boldsymbol{v}) = \frac{1}{\mathcal{Z}} \sum_{\boldsymbol{h}} \exp(-E_\theta(\boldsymbol{z})) \tag{3.8}$$

3.2.2 Gibbs sampling

Calculating the partition sum or integrating out the latent variables $\boldsymbol{h}$ of the generative model are instances of probabilistic inference problems. For large N this is an intractable computation since it comprises the summation over exponentially many terms. Sampling techniques, in particular Markov chain Monte Carlo (MCMC), are a common remedy enabling efficient probabilistic inference.

A Markov chain is a sequence of states from a finite set S where the transition from one state s to the the next s' is governed by a stochastic operator $T(s'|s)$. Starting in a state s_0 the next entry is sampled according to $s_1 \sim T(s|s_0)$ and so forth. If this stochastic process has no unreachable states (irreducibility) and does not enter deterministic cycles (aperiodicity) the entries of the Markov chain approach a fixed point distribution $p(s) = \sum_{s'} T(s|s')p(s')$ which is independent from the initial condition s_0.

The goal of MCMC methods is to choose T such that the desired distribution $p(s)$ is obtained. A sufficient, but not necessary design choice for this is to create a *reversible* chain by demanding what is known as *detailed balance*: $T(s|s')p(s') = T(s'|s)p(s) \forall s, s'$. This condition ensures that probability flux is conserved locally and implies that the global fixed point exists. Given a suitable transition operator T one can perform probabilistic inference by generating sufficiently long Markov chains and recording samples of the relevant variables.

Algorithm 1: Gibbs sampling for BM with N units, drawing N_{sample} samples per N_{chain} Markov chains.

sample list $S = \{\}$;
for $c \in \{1, \cdots, N_{\text{chain}}\}$ **do**
    initial configuration $\boldsymbol{z}^1 \in \{0,1\}^N$; `// random or from data set`
    for $s \in \{2, \cdots, N_{\text{sample}}\}$ **do**
        choose $i \in \{1, \cdots, N\}$; `// randomly or fixed order`
        compute $p_\theta(z_i | \boldsymbol{z}_{\setminus i}^{s-1}) = \sigma(u_i)$;
        sample $z_i^s \sim p_\theta(z_i | \boldsymbol{z}_{\setminus i}^{s-1})$;
        add $\boldsymbol{z}^s = (z_1^{s-1}, \cdots, z_{i-1}^{s-1}, z_i^s, z_{i+1}^{s-1} \cdots, z_N^{s-1})$ to S;
return $S = \{\boldsymbol{z}^s\}_{s=1}^{N_{\text{sample}} \cdot N_{\text{chain}}}$;

Gibbs sampling [GG] is a particular MCMC algorithm which relies on the efficient computability of conditional probabilities. Algorithm 1 describes Gibbs sampling for simulating the equilibrium state of a BM, given parameters θ. In every iteration the Gibbs sampler chooses a binary variable z_i and updates its value to a random sample from the conditional probability given the remaining variables: $\hat{z}_i \sim p_\theta(z_i | \boldsymbol{z}_{\setminus i})$. For the BM the conditional probability of one unit given the rest is a logistic function

$$p_\theta(z_i = 1 | \boldsymbol{z}_{\setminus i}) = \frac{1}{1 + \exp(\sum_j W_{ij} z_j + b_i)} = \sigma(u_i) \tag{3.9}$$

where $u_i = \sum_j W_{ij} z_j + b_i$ is the total input to unit i. By analogy to how neural networks process their input, σ is often called an activation function.

There are some caveats to Gibbs sampling, just like with any MCMC method. Depending on the starting configuration the initial samples of a burn-in phase might have to be discarded to avoid bias. Also running multiple Markov chains can be useful when dealing with islands in multi-modal distributions. Another inconvenience is that for fully-connected BMs, like shown in fig. 3.4 (left), it takes at least N Gibbs steps to completely exchange a configuration, which slows down mixing and produces long auto-correlation times [FI14].

This is one reason that motivates the architectural choice of the RBM [Smo86] as depicted in fig. 3.4 (right). In an RBM visible and hidden units form the layers of a bipartite graph such that the intra-layer weights become zero and only the interactions between them remain: $\tilde{W} \in \mathbb{R}^{N_v \times N_h}$.

Thereby, individual hidden (visible) units become conditionally independent of each other $p(h|v) = \prod_{j=1}^{N_h} p(h_j|v)$ which is a major advantage since it allows for parallel block Gibbs updates of each layer as described in algorithm 2.

Algorithm 2: Block Gibbs sampling for RBM with N_v visible and N_h hidden units drawing N_{sample} samples.

sample list $S = \{\}$;
initial configuration $\boldsymbol{v}^1 \in \{0,1\}^{N_v}$; `// random or from data set`
for s *in* $2, \cdots, N_{\text{sample}}$ **do**
    sample $\boldsymbol{h}^s \sim p_\theta(\boldsymbol{h}|\boldsymbol{v}^{s-1})$; `// hidden block update`
    sample $\boldsymbol{v}^s \sim p_\theta(\boldsymbol{v}|\boldsymbol{h}^s)$; `// visible block update`
    add $\boldsymbol{z}^s = [\boldsymbol{v}^s, \boldsymbol{h}^s]$ to S;
return $S = \{\boldsymbol{z}^s\}_{s=1}^{N_{\text{sample}}}$;

3.3 Spike-based sampling

In the last section we have seen how our generative model, the BM, could work on a general purpose computer. In this section a spike-based sampling algorithm that employs networks of LIF neurons is discussed, making it especially suited for implementation on neuromorphic hardware. We introduce the necessary technical details of the BSS2 chip and how to perform LIF sampling with it.

3.3.1 Neural and LIF sampling

Two challenges come to mind immediately, when thinking about sampling with SNNs:

1. How can the continuous state of membrane potentials $\{\boldsymbol{u}\}$ of the SNN be identified with the discrete sampling space $\{\boldsymbol{z}\}$?
2. How can stochasticity be introduced into deterministic SNNs (e.g. based on eq. (3.1)) such that sampling from the desired distribution $p(\boldsymbol{z})$ occurs?

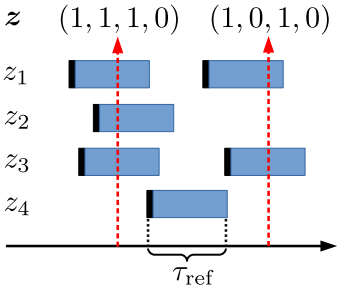


Figure 3.5: Mapping from spikes to binary configurations

Neural sampling

Regarding the first point, there exists a natural choice for interpreting the activity of a spiking neuron as binary random variables $z_k \in \{0,1\}$ by identifying the spike as a transition $z = 0 \to z = 1$ and flipping back at the end of the neuron's refractory period $z = 1 \to z = 0$. Thus, a spike is mapped to a rectangular pulse in the neuron state as visualized in fig. 3.5.

To deal with the second challenge, early versions of neural sampling used intrinsically stochastic neurons with simplified interactions (see Buesing et al. [BBNM11]).

The sampling process consists of reading out the network state over time, where, like in Gibbs sampling a neuron's state is chosen based on its conditional distribution. However, the full transition dynamics are more complicated since they have to incorporate the deterministic refractory mechanism.

A central property of this neural sampling framework is the local computation of the state log-odds by a variable that is identified with the neuron's membrane potential:

$$u_k = \log \frac{p(z_k = 1|\boldsymbol{z}_{\setminus k})}{p(z_k = 0|\boldsymbol{z}_{\setminus k})} \tag{3.10}$$

For a BM this translates into the same activation function as in eq. (3.9). The state updates of other variables thus act like a rectangular PSP on the potential $u_k(t) = b_k + \sum_{i=0}^{N} W_{ki} z_i(t)$ with a synaptic "time constant" $\tau_{\text{syn}} = \tau_{\text{ref}}$. From these prerequisites one can design and validate a suitable, irreversible, transition operator for a Markov chain that samples from a BM.

The crux of this sampling method is that neurons efficiently communicate their state to each other through spikes and influence their connection partners accordingly. This begs the question as to how one could realize such a sampler on neuromorphic substrates with more realistic neuron models.

LIF sampling

Unlike the above mentioned abstract neural sampling with stochastic neurons, Petrovici et al. [PBB+, PBB+16] introduce a framework with deterministic LIF neurons.

There are some biological implausibilities of the abstract model that make a direct implementation with LIF neurons difficult. For instance the reset mechanism for the membrane potential is absent and spike transmission is instantaneous. However, the most important aspect is the intrinsic stochasticity which stands in contrast to in vitro neurons [MS95] and the LIF model.

The approach in [PBB+] is to create a stochastic LIF neuron that obeys eq. (3.10) (up to linear transformation) by subjecting it to external noise input. In the brain this activity could stem from the larger cortical area that specific functional units are embedded in. This noise is Poisson-distributed with rates ν and originates from a balanced number of excitatory and inhibitory sources .

As a result the neurons are elevated into a high-conductance state (HCS) such that the membrane time scale is much smaller than the synaptic/refractory one: $C_m/g_l = \tau_m \ll \tau_{\text{syn}} = \tau_{\text{ref}}$. This ensures that when freely evolving, the membrane potential u (see eq. (3.1)) closely follows the effective counterpart u_{eff} which now includes the noise input in addition to the leak and recurrent network input.

Under these assumptions u_{eff} was shown to follow an Ornstein-Uhlenbeck process (OU)

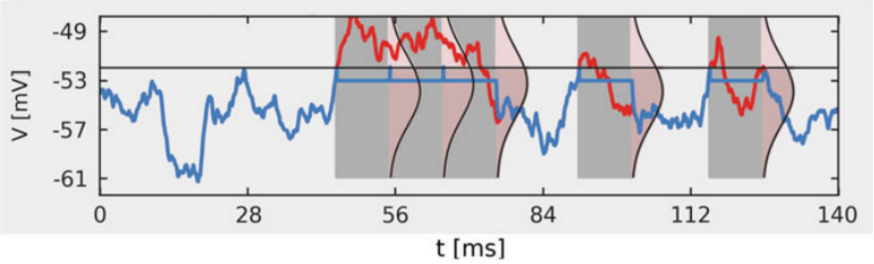


Figure 3.6: Membrane trace of u (blue) and u_{eff} (red) in the HCS. Grey areas show refractory periods, pink areas illustrate the Gaussian distribution of u_{eff} due to Poisson noise influence. (Taken from [Pet16])

[UO30]:

$$\mathrm{d}u = \frac{1}{\tau_{\text{syn}}}(\bar{u} - u_{\text{eff}}(t))\mathrm{d}t + \sigma \mathrm{d}W_t \tag{3.11}$$

where W_t is the Wiener process (Gaussian random walk). Intuitively speaking, this means that the steady state distribution of the membrane potential, outside of refractory periods, is Gaussian distributed with mean $\bar{u} = E_l + \tau_{\text{syn}} \sum_p w_p \nu_p / g_l$ and variance $\sigma^2 = \tau_{\text{syn}} \sum_p w_p^2 \nu_p / 2 g_l^2$ where index p denotes the noise source.

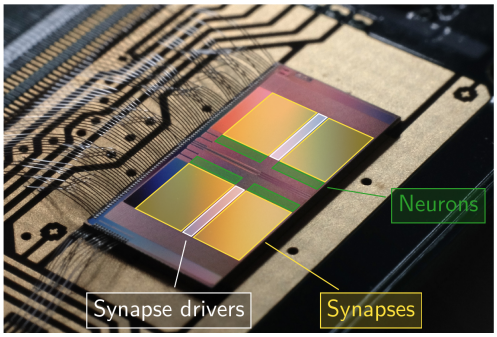
In a rather involved calculation, [PBB+] goes on to show that the activation function of a LIF neuron approximates a logistic function in the mean effective potential:

$$p(z_k = 1 | \bar{u}_k) = \sigma\left(-\frac{\bar{u}_k - \bar{u}_k^0}{\alpha}\right) \tag{3.12}$$

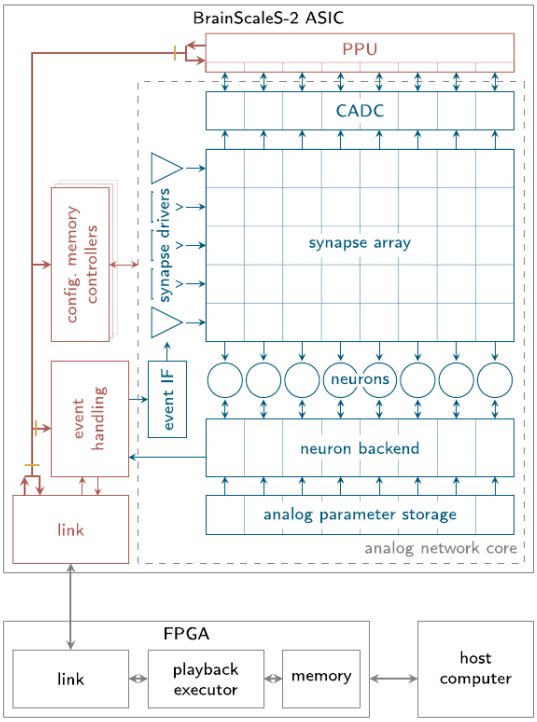
The offset $\bar{u}_k^0$ characterizes the point where $p(z_k = 1) = 0.5$. Implicitly, a temperature of $T = 1$ is introduced such that the scaling factor α has the physical interpretation of the Boltzmann constant. With this description a translation between parameters in the LIF sampling implementation and the abstract parameters of the corresponding BM becomes possible.

For biases this is especially easy as they transform with $b_k = (\bar{u}_k^b - \bar{u}_k^0)/\alpha$. Weight translation should take the impact of spikes from presynaptic neurons into account when comparing the domains. In [PBB+16] this is done by equating the integral of the PSPs: $w_{ij}/\alpha = \int_0^{\tau_{\text{ref}}} PSP(t)\mathrm{d}t = W_{ij}\tau_{\text{ref}}$.

Under these assumptions previous work has shown to be able to approximate arbitrary distributions, achiving DKLs $\approx 10^{-2} - 10^{-3}$ when comparing with Gibbs sampling up to $\mathcal{O}(1000)$ neurons [LMB+18, Bau20].



(a)



(b)

Figure 3.7: (a) analog part of BSS2, HICANN-X (taken from [CBB+21]), (b) schematic of BSS2 (taken from [GBC+20])

3.3.2 Sampling on neuromorphic hardware

While the idea of dedicated hardware that implements neural functionality dates back to the 80's [Mea89], the recent interest in neural networks fueled by deep learning in conjunction with advances in neuroscience have lead to a "cambrian explosion" [HP19] in the number and diversity of available neuromorphic hardware systems.[3] The architectures reach from purely digital systems like SpiNNaker [PPG+13, MHF19] and Intel's Loihi to [DSL+18] to mixed-signal types like the BrainScaleS hardware [SBG+10, SBDW20] developed by the Electronic Visions group at Universität Heidelberg of which we exclusively used the BSS2 version [SBDW20, GBC+20, BCP+21].

Technical specifications of BSS2

The BSS2 chip is depicted in fig. 3.7a. The system is an ASIC consisting of an analog core, named High Input Count Analog Neural Network (HICANN-X), digital comodules for readout and configuration and additional plasticity processing units (PPU)

[3] this diversity is further exaggerated by the ubiquitous use of the word "neuromorphic"

which enable the on-chip execution of local learning rules. A host computer can communicate and configure BSS2 via a field-programmable gate array (FPGA). A schematic of those components is shown in fig. 3.7b.

The analog core can implement up to 512 current based physical LIF neurons[4] and an array of 256 synapse drivers. We configure the chip to implement an all-to-all connectable network of 256 neurons. In particular, this matches two neighboring synapses to implement signed synapses. The resolution of the analog synaptic weights is 6-bit plus the sign, i.e. $w_{\mathrm{hw}} \in \{-63, \cdots, 63\}$, that of biases is 10-bit $b_{\mathrm{hw}} \in \{0, \cdots, 1023\}$. These values map to the synaptic strength in nA and leak potential in mV in an assumed monotonic fashion with an unknown scaling factor depending on the chip configuration. This will be discussed below.

The chips analog nature allows for circuit time constants that are $10^3 - 10^4$ times faster than biology shifting them from the ms to the $\mu\mathrm{s}$ scale and justifying the name *accelerated* neuromorphic computing.

On-chip analog-to-digital converters (ADC) allow for reading out state variables, of which we only used membrane voltages of the neuron circuits. Spikes are detected individually by digital neuron backends and communicated as time stamped events for external handling.

Implementation: HXSampling software and calibration

The Python backend for LIF sampling on BSS2 is called *HXSampling* and was provided by Sebastian Billaudelle, Benjamin Cramer and Andreas Baumbach.

It features the possibility for setting up networks of arbitrary connectivity with up to 64 (192) logical neurons for version 1 (version 2) of the BSS2. The reduction of logical network sizes below the total of 256 neurons is a result of the 64 logical noise sources that occupy some of the available synapse drivers. The associated Poisson-distributed spike trains are generated on-chip, reducing the required communication bandwidth.

Before running an experiment, a chip ID and the associated FPGA have to be specified. This setup needs to be calibrated which entails setting the following parameters: leak E_l, reset u_R, threshold u_T, membrane time constant τ_m, synaptic time constant τ_{syn}, membrane capacitance C_m, refractory period τ_{ref} and two technical parameters determining the gain factor for the synaptic input circuitry, $I_{\mathrm{gm}}^{\mathrm{syn}}$ and b_{syn}.

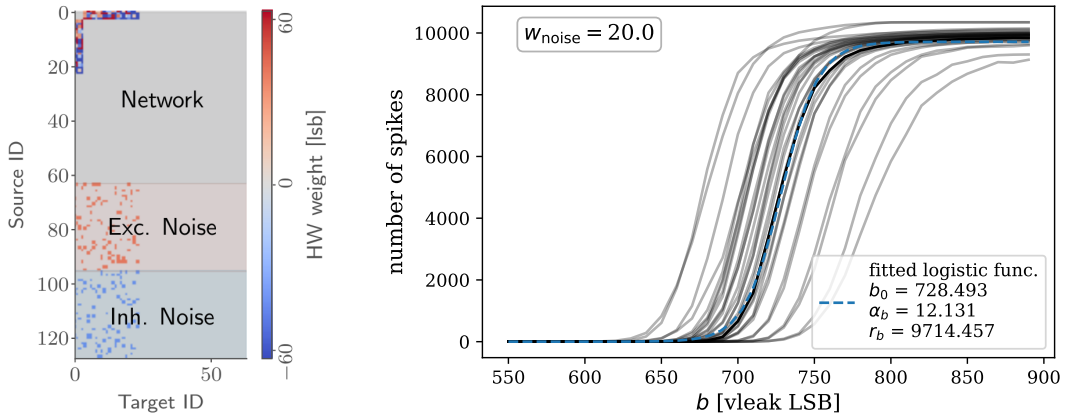
Calibration parameters have changed over time as the BSS2 system evolves. The following is a set of typical calibration parameters for version 2 provided to calix [5]: $E_l = u_R = 80\,\mathrm{lsb}$[6], $u_T = 120\,\mathrm{lsb}$, $\tau_m = 0.5\,\mu\mathrm{s}$, $\tau_{\mathrm{syn}} = \tau_{\mathrm{ref}} = 10\,\mu\mathrm{s}$, $I_{\mathrm{gm}}^{\mathrm{syn}} = 350\,\mathrm{lsb}$, $C_m = 10\,\mathrm{lsb}$, $b_{\mathrm{syn}} = 600\,\mathrm{lsb}$.

With such a calibration configuration, the selected setup can be calibrated, which

[4] actually the more general class of AdEx-IF neurons can be implemented, see [BG05]

[5] The calibration framework developed by Johannes Weis.

[6] lsb stands for "least significant bit" and is for our purposes arbitrary.



(a) (b)

Figure 3.8: (a) Visualization of synaptic weight matrix of an RBM with additional noise connections in version 1 of BSS2 (taken from [CBB+21]), (b) Measured activation functions by scanning the bias of individual neurons. Fitting a logistic function yields a good approximation and symmetric deviations. For technical reasons offsets of the activation function are randomly distributed.

produces a reusable file which is required to initialize a *HXSampler* object. It establishes a connection to the chip and provides a high-level wrapper for specifying weights and biases, running experiments and retrieving the spike-based samples. To initialize *HXSampler*, further noise and sampling parameters have to be chosen.

The following are our standard noise settings (see table B.2). The noise weight $w_{\text{noise}} = 15 - 25$ specifies the synaptic strength of the noise synapses which stimulate the network neurons with a noise rate of $\nu_{\text{noise}} \approx 80\,\text{kHz}$. The noise multiplier $m_{\text{noise}} = 4$ specifies the number of inhibitory and excitatory sources that are connected to each network neuron. The specific connectivity is randomly chosen from the two pools of 32 sources providing excitatory and inhibitory input respectively. Figure 3.8a shows the layout of the complete network matrix.

Finally, the sampling parameters have to be specified (see also table B.2). The chip runtime $T = 0.1\,\text{s}$ is the typical time for which the LIF sampling process is emulated. The amount of samples extracted from running the chip once is determined by the sampling interval $dt = \tau_{\text{ref}}/2 = 5\,\mu\text{s}$, i.e. $N^{\text{single}}_{\text{sample}} = \lfloor T/dt \rfloor = 20000$. In most experiments chip execution is repeated and samples are aggregated over repetitions in order to mitigate problems discussed in section 6.1. The respective parameter for the number of repetitions is N_{reps}, such that the total number of samples is $N_{\text{sample}} = N_{\text{reps}} \cdot N^{\text{single}}_{\text{sample}}$.

One can check the calibration by measuring activation functions and verifying that they look reasonable, i.e. approximately symmetric. In fig. 3.8b such a measurement is shown where the leak E_l is scanned which is the parameter that implements the bias of

the sampling neurons.

3.4 Learning algorithms

In this thesis, we have employed both gradient-based and gradient-free optimization algorithms to optimize the BM for quantum patterns. Both types minimize an objective function $C(\theta)$: for QST the DKL with respect to a target data distribution is minimized, in the case of VGS the expected value of the Hamiltonian serves as the cost metric.

Starting from random variational parameters, i.e. network weights and biases, an iterative scheme improves the solution step by step until a convergence criterion is met. Initialization of network weights are typically done by symmetric random sampling around zero from a uniform random distribution with half-width w_{init}, $w_{ij} \sim \mathcal{U}(-w_{\text{init}}, w_{\text{init}})$, while bias parameters very initialized at the repective inflection point of the neurons' activation function $b_k = b_k^0$. For neuromorphic hardware a typical value is $w_{\text{init}} = 50\,\text{lsb}$, with $w_{\text{max}} = 63\,\text{lsb}$ marking the maximum value of network weights. For software RBMs the value typically assumed values much smaller than one $W_{\text{init}} \lessapprox 0.1$.

3.4.1 Gradient-based

The gradient of the BM

Let's briefly look at how patterns can be learned with the gradient of the BM. The derivative of the distribution $p_\theta(\boldsymbol{v})$ encoded by a BM with respect to a weight W_{ij} is given by

$$\frac{\partial \log p_\theta(\boldsymbol{v})}{\partial W_{ij}} = \sum_{\boldsymbol{h}} z_i z_j \frac{p_\theta(\boldsymbol{z})}{p_\theta(\boldsymbol{v})} - \sum_{\boldsymbol{z'}} z'_i z'_j p_\theta(\boldsymbol{z'}) = \langle z_i z_j \rangle_{p_\theta(\boldsymbol{h}|\boldsymbol{v})} - \langle z_i z_j \rangle_{p_\theta(\boldsymbol{z})} \quad (3.13)$$

The first term can be interpreted as expected value when the visible units are clamped to $\boldsymbol{v}$, whereas the second term describes an average for the free running BM. An analogous expression holds for a bias b_k.

This expression determines how to adjust the weights in order to enhance the probability of configuration $\boldsymbol{v}$. Doing this for a given set of patterns $\mathcal{S} = \{\boldsymbol{v}_s\}_{s=1}^{N_s}$ in proportion to their frequency in the underlying distribution $p^*(\boldsymbol{v})$ is called the *maximum likelihood method*.

The inference problem of finding optimal parameters θ^* by minimizing the negative log-likelihood over models θ is equivalent to minimizing the DKL:

$$\theta^* = \text{argmin}_\theta[-\sum_{\boldsymbol{v}} p^*(\boldsymbol{v}) \log p_\theta(\boldsymbol{v})] = \text{argmin}_\theta[D_{KL}(p^* \| p_\theta)] \quad (3.14)$$

The gradient of the DKL with respect to W_{ij} and b_k is straightforward when combined with eq. (3.13),

$$\frac{\partial D_{KL}(p^*\|p_\theta)}{\partial W_{ij}} = -\sum_{\{\boldsymbol{v}\}} p^*(\boldsymbol{v}) \frac{\partial \log p_\theta(\boldsymbol{v})}{\partial W_{ij}} = -\langle \langle z_i z_j \rangle_{p_\theta(\boldsymbol{h}|\boldsymbol{v})} \rangle_{p^*(\boldsymbol{v})} + \langle z_i z_j \rangle_{p_\theta(\boldsymbol{z})} \quad (3.15)$$

where the first term turns into a double average sampling $\boldsymbol{v}$ from the true distribution p^* and $\boldsymbol{h}$ from the conditional model $p_\theta(\boldsymbol{h}|\boldsymbol{v})$. The first term is often referred to as a *wake* phase, where the network perceives the environment p^*, while the second describes the *sleep* phase, where it is decoupled from the environment.

The above gradient can be used to implement a *local learning rule* since every unit only requires knowledge about its neighboring connections. In contrast to learning rules involving global network information, local updates are *biologically plausible* and thus especially suitable for *on-chip* implementation in neuromorphic hardware.

Stochastic gradient descent

We denote the full gradient with respect to the model parameters with $\Delta\theta = \nabla_\theta C(\theta)$ from now on. In our case stochasticity is an inherent part of the cost function due to the sampling process involved.

Vanilla gradient descent is the standard approach of updating the parameters in the opposite direction of the gradient vector at the current position in parameter space, thereby lowering the cost:

$$\theta_{e+1} \leftarrow \theta_e - \eta_e \Delta\theta_e \quad (3.16)$$

Since the gradient only provides local guidance, it is advisable to scale its components according to the roughness of the cost landscape. The scaling factor η is called the *learning rate*. It determines the step size, corresponding to the resolution at which the cost surface is being probed and bounds the number of steps that need to be computed to reach convergence. We typically employed a exponentially decaying learning rate $\eta_{e+1} = \eta_e \cdot \gamma_{\text{lr}}$ such that $1/(1-\gamma_{\text{lr}})$ sets a time scale of optimization steps. Our standard parameters are: $\eta_1 = 1$, $\gamma_{\text{lr}} = 0.999$ (see table B.3). For software RBM simulations the learning rate was typically smaller by a factor of 10.

There are countless improvements and extensions of the standard gradient descent. One of the most widely used schemes is adaptive moment estimation (ADAM) [KB17]. ADAM is a first-order method that estimates mean m_t and variance v_t of the gradient by exponential running averages with respective decay rates β_1 and β_2:

$$m_{e+1} \leftarrow \frac{\beta_1}{1-\beta_1^e} m_e + \frac{1-\beta_1}{1-\beta_1^e} \Delta\theta_e \quad (3.17)$$

$$v_{e+1} \leftarrow \frac{\beta_2}{1-\beta_2^e} v_e + \frac{1-\beta_2}{1-\beta_2^e} \Delta\theta_e^2 \quad (3.18)$$

Here $\Delta\theta_t^2$ is the component-wise square of the gradient.

The parameters are updated according to the inverted relative error of the gradient

$$\theta_e \leftarrow \theta_{e+1} - \eta_e \frac{m_e}{\sqrt{v_e} + \epsilon_{\mathrm{ADAM}}} \tag{3.19}$$

where ϵ_{ADAM} is a small positive value.

Since $|\Delta\theta_e/\sqrt{v_e}| \leq 1$ the update implicitly implements an adaptive step size for every parameter that becomes small for noisy gradients and large for low relative error.

Standard hyperparameters for ADAM are listed in table B.3.

Natural gradient descent

A "natural" extension to simple gradient methods are ones that incorporate knowledge about the curvature of the cost landscape. Natural gradient descent (NGD) is a method for likelihood-based models that estimates the Fisher information matrix (FIM) [PB14]. The FIM is the negative expected value of the Hessian thereby providing information about the curvature of the optimization landscape. NGD ideally results in faster convergence, however, it is still a gradient method and therefore there is no guarantee to find a global optimum. Let us denote the derivative of the log-likelihood $O_k(\boldsymbol{v}) = \partial_k \log p_\theta(\boldsymbol{v})$. Then, the FIM is the expected covariance matrix of $O_k(\boldsymbol{v})$:

$$I_{k,k'} = \langle O_k(\boldsymbol{v}) O_{k'}(\boldsymbol{v}) \rangle_{p_\theta(\boldsymbol{v})} - \langle O_k(\boldsymbol{v}) \rangle_{p_\theta(\boldsymbol{v})} \langle O_{k'}(\boldsymbol{v}) \rangle_{p_\theta(\boldsymbol{v})} \tag{3.20}$$

$$= \langle \partial_k \log p_\theta(\boldsymbol{v}) \partial_{k'} \log p_\theta(\boldsymbol{v}) \rangle \tag{3.21}$$

where we have used that $\langle O_k(\boldsymbol{v}) \rangle = \partial_k \sum_{\boldsymbol{v}} \log p_\theta(\boldsymbol{v}) = 0$.

The natural gradient parameter updates are obtained by inverting I and applying it to the gradient $\Delta\theta$:

$$\theta^{e+1} = \theta^e - \eta I_e^{-1} \Delta\theta^e \tag{3.22}$$

Second-order information does not come for free: more samples are needed to estimate the FIM of dimension $\mathbb{R}^{\dim\theta^2}$, which increases the computational effort. Additionally, the matrix inversion usually requires elaborate regularization techniques.

It should be noted that [CT17] used a method called stochastic reconfiguration for a direct wave function representation with complex RBMs. This method is equivalent to NGD when dealing with probabilities and real parameters.

3.4.2 Gradient-free: Differential Evolution

Evolutionary algorithms (EA) are a large class of gradient-free optimization methods that roughly borrow from the principle of biological evolution. A population $\{\boldsymbol{\theta}\}_{i=1}^{N_p}$ of trial solutions ("individuals") is proposed. Initially, they are randomly selected and

subsequently evolved by crossover and/or random changes ("mutation"). Trial vectors are compared based on a fitness function F ("selection").

One instance of an EA is the global optimization algorithm called Differential Evolution (DE) which works by combining vector differences in each iteration [PSL05]. For updating trial vector $\boldsymbol{\theta}$ three other trial vectors $\boldsymbol{a},\ \boldsymbol{b},\ \boldsymbol{c}$ are randomly chosen from the population. These three vectors are combined to $\boldsymbol{\theta'}$ that replaces $\boldsymbol{\theta}$ if $F(\boldsymbol{\theta'}) \geq F(\boldsymbol{\theta})$. The construction occurs component-wise with a fixed crossover rate of CR and differential weight DW:

$$\theta'_i = \begin{cases} a_i + DW(b_i - c_i) & \text{if } CR < u \sim \mathcal{U}(0,1) \\ \theta_i & \text{else} \end{cases} \tag{3.23}$$

After a termination criterion is met the trial vector with the highest $F(\boldsymbol{\theta})$ is returned. The DE strategy described above is called "rand/bin" due to the random choice of the support vector $\boldsymbol{a}$ and the binomial crossover of new trial components.

Since no assumptions are made about F this method is especially suited for opaque non-differentiable systems, e.g. physical experiments.

4 Towards Neuromorphic Quantum State Tomography

This chapter shows our results on learning to represent known target quantum states with BMs using both conventional Gibbs sampling as well as LIF sampling on BSS2.

It is a continuation of the recent work of Czischek et al. [CBB+21], where the BSS2 chip has been used to learn Bell states $|\psi_{\text{Bell}}\rangle = (|00\rangle + |11\rangle)/\sqrt{2}$ demonstrating that the SNN is able to capture their non-classical correlations. They further encoded the N-particle generalization, the Greenberger-Horne-Zeilinger (GHZ) state, $|\psi^N_{\text{GHZ}}\rangle = (|0\rangle^{\otimes N} + |1\rangle^{\otimes N})/\sqrt{2}$ for $N = 3$ and $N = 4$ particles showing that the method struggles to scale when dealing with these larger states. We reproduced these experiments and improved upon them. Furthermore, we have explored the representability of ground states and steady states of the TFIM.

4.1 Learning target states - the brute force method

Although QST is about sample efficiency in this thesis we did not use experimental nor synthetic data sets. Instead we have assumed infinite data, i.e. perfect knowledge of the target distribution $p^*(\boldsymbol{v})$.

In this case one can get rid of the expected value over the data set in eq. (3.15) by expanding the first term with $p_\theta(\boldsymbol{v})$. This eliminates the conditional probability and allows for rewriting the expression as expected value of the joint distribution over a reweighted target $p^*/p_\theta(\boldsymbol{v})$:

$$\Delta W_{ij} = \langle \langle z_i z_j \rangle_{p_\theta(\boldsymbol{h}|\boldsymbol{v})} \rangle_{p^*(\boldsymbol{v})} - \langle z_i z_j \rangle_{p_\theta(\boldsymbol{z})} \tag{4.1}$$

$$= \sum_{\{\boldsymbol{z}\}} z_i z_j p_\theta(\boldsymbol{h}|\boldsymbol{v}) p_\theta(\boldsymbol{v}) \frac{p^*(\boldsymbol{v})}{p_\theta(\boldsymbol{v})} - \langle z_i z_j \rangle_{p_\theta(\boldsymbol{z})} \tag{4.2}$$

$$= \Big\langle z_i z_j \frac{p^*(\boldsymbol{v})}{p_\theta(\boldsymbol{v})} \Big\rangle_{p_\theta(\boldsymbol{z})} - \langle z_i z_j \rangle_{p_\theta(\boldsymbol{z})} \tag{4.3}$$

$$= \left\langle \left(\frac{p^*(\boldsymbol{v})}{p_\theta(\boldsymbol{v})} - 1 \right) z_i z_j \right\rangle_{p_\theta(\boldsymbol{z})} \tag{4.4}$$

An analogous expression is obtained for the biases:

$$\Delta b_k = \left\langle \left(\frac{p^*(\boldsymbol{v})}{p_\theta(\boldsymbol{v})} - 1 \right) z_k \right\rangle_{p_\theta(\boldsymbol{z})} \tag{4.5}$$

This learning rule thus provides a method to learn a target distribution without the need of a "wake" phase, where the states of neurons would be clamped. While this is sufficient to explore the representability of small quantum state distributions there are important drawbacks. First, replacing the conditional expectation comes at the cost of knowing the fraction $p^*/p_\theta(\boldsymbol{v})$ for every configuration $\boldsymbol{v}$ with non-negligible contribution. In the many-body limit this is exponentially costly both in terms of measuring sufficiently accurate target distributions $p^*(\boldsymbol{v})$, as well as densely sampling the model $p_\theta(\boldsymbol{v})$. That is why we came to call these types of learning rules *brute-force*. Another important consequence is that the appearance of the joint model distribution in the expected value combines information from all visible units thereby turning the previously local learning rule into a global one.

The complete procedure for learning target states in this manner is described in algorithm 3. After the calibration and initialization of BSS2, the DKL is minimized by alternating between on-chip LIF sampling and turning the gradient estimation on the host computer into parameter updates with the respective optimizer o. This is a type of *in-the-loop learning* in contrast to potential algorithms that make use of *on-chip learning* capabilities. Also, note that the model distribution is estimated from the same samples as the gradient itself.

Algorithm 3: Gradient-based in-the-loop learning of target state p^* for N_{epoch} steps. At every step N_{sample} configurations are generated using BSS2.

1	calibrate chip setup ;	`// BSS2`
2	provide target $p^*(\boldsymbol{v})$;	`// Host`
3	initialize parameters θ_1 ;	`// Host`
4	**for** $e \in \{1, \cdots, N_{\text{epoch}}\}$ **do**	
5	configure parameters θ_e on chip ;	`// BSS2`
6	sample model data $\mathcal{S} = \{\boldsymbol{z}\}_s^{N_{\text{sample}}} \sim p_\theta(\boldsymbol{z})$;	`// BSS2`
7	create histogram $p_\theta(v) = \frac{1}{N_{\text{sample}}} \sum_{\boldsymbol{z}_s \in S} \mathbb{1}\left[\boldsymbol{v}_s = \boldsymbol{v}\right]$;	`// Host`
8	calculate gradient $\Delta\theta_e = \partial_\theta D_{KL}(p^*(\boldsymbol{v}) \| p_\theta(\boldsymbol{v}))$;	`// Host`
9	update parameters $\theta_{e+1} \leftarrow \theta_e - \eta_e o(\Delta\theta_e)$;	`// Host`
10	generate samples after convergence ;	`// BSS2`
11	evaluate metrics and expected values ;	`// Host`

Choosing the optimization method

Besides the above gradient-based algorithm we looked into using evolutionary algorithms, specifically DE, as an alternative optimization method. These gradient-free methods are suitable for blackboxes, i.e. systems that are not differentiable or whose gradient is unknown and thus can not be optimized with gradient descent. In general the BSS2 can be seen as such a blackbox, however, for the specific case of LIF sampling we know that the SNN approximates a BM and thus gradient-based methods are applicable, too.

In fig. 4.1 an exemplary comparison between learning curves of DE and ADAM the DKL over the course of training are shown. The target state is the Bell state $|\psi_{\text{Bell}}\rangle$ probabilistically expressed with the tetrahedral POVM such that the visible layer contains $N_v = 4$ units. In this encoding the 4 POVM outcomes for each spin correspond to the joint configurations of two neurons in the BM. The employed RBM architecture has $N_h = 4$ which is providing more parameters $N_{\text{params}} = 16 + 4 + 4 = 24$ than the 15 degrees of freedom in the two-qubit density matrix. For both ADAM and DE we used the standard optimizer parameters given in table B.3 except for a reduced population size of $N_p = 24$.

We directly compare the learning progress with cost function evaluations N_{eval} which in the case of ADAM is equal to N_{epoch}. Every function evaluation entails drawing $N_{\text{sample}} = 60000$ samples with standard sampling settings (see table B.2). While there might still be better hyperparameters to improve the performance of DE, the point of showing this plot is that the gradient-based method is orders of magnitudes more efficient in converting function evaluations into useful parameter updates. This is expected due to the extra information about the cost landscape enabling it to quickly reach a final of DKL $\approx 5 \cdot 10^{-3}$.

Based on these results we decided to use gradient-based optimization for the remainder of our experiments.

What happens when training with sampled gradients?

In order to assess the learning process of algorithm 3 we did numerical experiments where we trained RBMs with Gibbs sampling. Again, we used the Bell state's SIC-POVM distribution as a simple example with $N_v = N_h = 4$. Figure 4.2(a)-(c) shows the learning curves of DKL as well as classical and quantum infidelity for increasing N_{sample}. The total number of samples was collected distributed over 16 Markov chains. Thus, the number of samples per visible configuration increases is a power of ten. In the left panels the orange points are the DKLs based on the sampled distribution at every epoch. The blue points (left) and the curves on the right panels are based on the exact distribution which is still feasible to calculate for this small system.

One can observe that the sample DKL initially drops until it converges at around 1000 epochs to a value given by the finite sample size $\propto 1/N_{\text{sample}}$. In contrast the metrics based on exact calculation using the RBM parameters continue to improve with

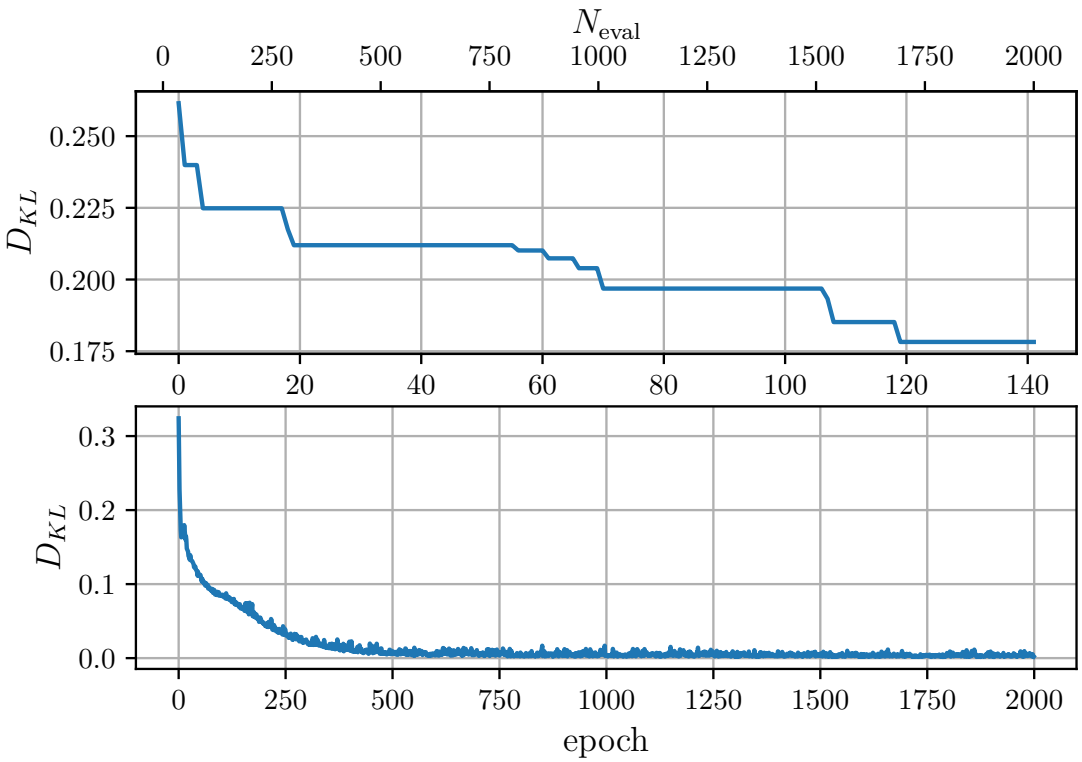


Figure 4.1: Example of training BSS2 on the Bell state with DE (top) and ADAM (bottom). Equivalent number of function evaluations N_{eval} are shown (1 per epoch for ADAM, $\mathcal{O}(N_p)$ per epoch for DE).

the sample-based gradient estimates. This shows that even a low number of samples is sufficient for obtaining informative updates. However, having more samples per update seems to improve the quality of the gradient resulting in faster learning which can be seen when looking up the number of epochs required to reach a fixed DKL.

4.2 Results for state representation

In the following sections we show the results of learning a variety of quantum states on BSS2, from product to entangled state, from pure to mixed state and from ground to steady state. All states have in common that they are related to the state space of the TFIM.

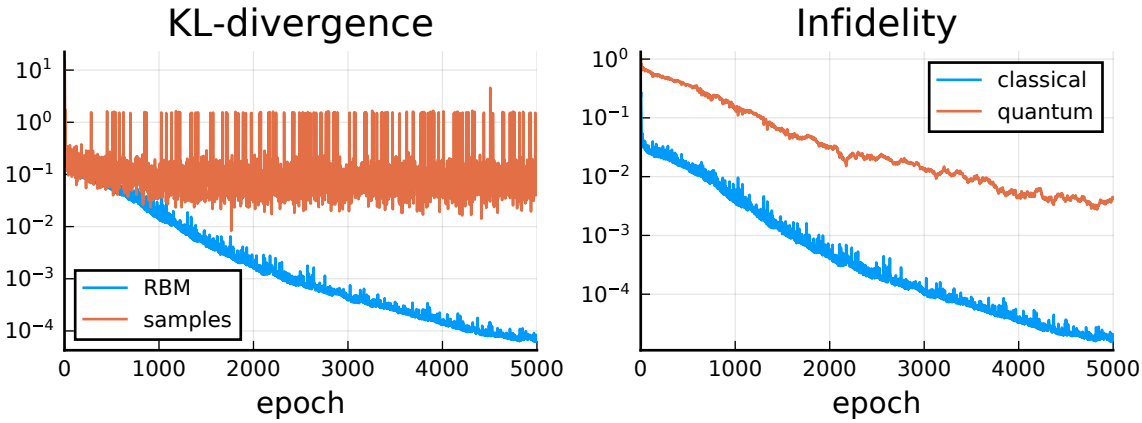
4.2.1 The spectrum of TFIM ground states

By changing the ratio between spin coupling and external field h/J in the TFIM one can move between its paramagnetic and ferromagnetic phase. For $h/J \gg 1$ the ground state approaches a product state where all spins are aligned in field direction. On the other end, for $h/J \ll 1$ the ordered phase exhibits a degeneracy between all spins aligning in positive or negative z-direction akin to a GHZ-like state. For $h \to 0$ this degeneracy is spontaneously broken in favor of one possibility. In this sense, the ground states of the TFIM interpolate between the extremes of uncorrelated product states $|\psi_+^N\rangle = (|0\rangle + |1\rangle)/\sqrt{2})^{\otimes N}$ and strongly-correlated entangled states $|\psi_{\text{GHZ}}^N\rangle$.

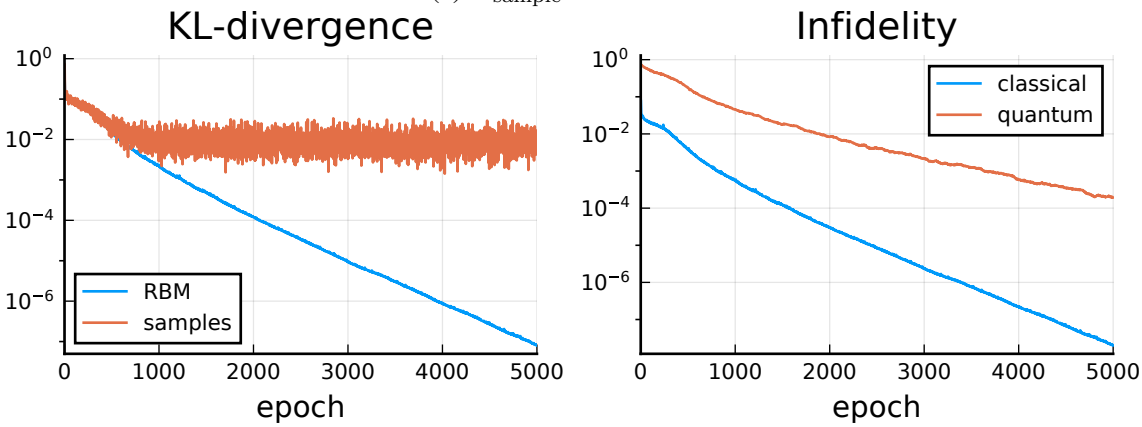
Figure 4.3a shows the Shannon entropy of the distributions obtained for different ground states of the TFIM ($N = 4$) for both the SIC-POVM and the z-basis. For comparison the entropy of a uniform distribution p_{uni} are also shown for each case. For the SIC-POVM the uniform distribution represents the maximally mixed state $\rho_{\text{mixed}} = 4^{-N}\mathbb{I}$ where all four POVM elements have identical projections. In the z-basis the uniform distribution is the equal superposition $|\psi_+^N\rangle$.

The relative entropies between the ground state encodings and the respective uniform distributions are shown in Figure 4.3b. For the SIC-POVM the distribution stays quite close to a uniform distribution with a minimum when approaching $h/J = 0$. In contrast at this point the z-basis distributions are very peaked since two configurations are dominating for $h \to 0$ and thus there is a large relative entropy.

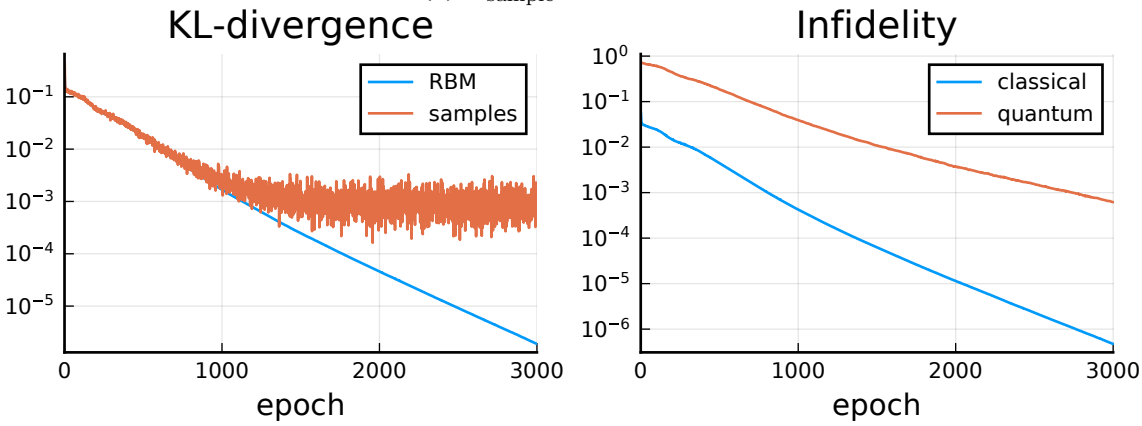
Naively, one could assume that BMs are better at representing distributions with higher Shannon entropy, the uniform distribution being the easiest one and worse at lower entropies, the delta distribution being the hardest. The reasoning is that one initializes the parameters randomly close to zero and a BM with constant or even zero parameters would indeed perfectly encode a uniform distribution. On the other hand, it is outright impossible to represent delta distributions where configurations with vanishing probability mass occur, due to the positivity of the Boltzmann factor. We will see



(a) $N_{\text{sample}} = 16 \cdot 100$

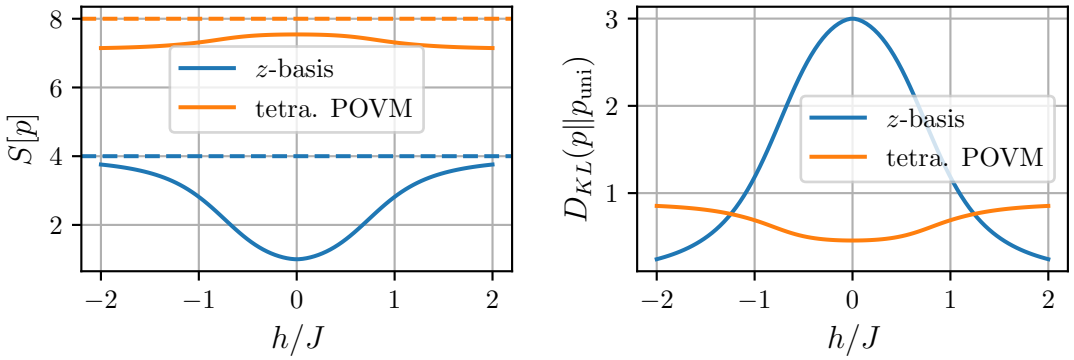


(b) $N_{\text{sample}} = 16 \cdot 1000$



(c) $N_{\text{sample}} = 16 \cdot 10000$

Figure 4.2: Numerically learning the Bell state with an RBM with $N_h = 4$. Left: Comparing the D_{KL} when Gibbs sampling and when analytically computing the marginal distribution $p_\theta(\boldsymbol{v})$. The parameter updates are based on sampled gradients where N_{sample} is the number of samples per epoch. Right: Fidelity and quantum fidelity exactly calculated based on the RBM parameters.



(a) (b)

Figure 4.3: (a) Shannon entropy of the TFIM ground state ($N = 4$) expressed as measurement distribution in the z-basis and in the tetrahedral POVM frame as a function of h/J (excluding $h = 0$). Dashed lines indicate entropies of respective uniform distributions p_{uni}. (b) DKL between p and p_{uni}.

by example that at least for the POVM representation this is not valid.

4.2.2 Product states

Product states are characterized by the property that their wave functions $|\psi\rangle = |\psi_1\rangle \otimes |\psi_2\rangle$ or their density matrices factorize: $\rho = \rho_1 \otimes \rho_2$. For spin systems that are in a product state over each site, the degrees of freedom grow linearly with N.

Intuitively, a description in terms of a probability distribution should exhibit this property as a factor distribution: $p(\boldsymbol{v}) = \prod_{i=1}^{N} p(v_i)$ which in the case of the BM distribution could be achieved by switching off the interaction $W = 0$. Then the relative importance of each site is given by the visible biases. This is sufficient in the z-basis representation where each neuron represents one spin. For the SIC-POVM where one spin is mapped to two binary units the factorization should be over marginals $p_\theta(\boldsymbol{v}) = \prod_{i=1}^{N} p_{\theta_i}(v_{2i}, v_{2i+1})$.

Figure 4.4 shows the example of learning the product state $|\psi\rangle = |1\rangle^{\otimes N}$ with SIC-POVMs and an RBM architecture with $N_h = 10$, sampling $N_{\text{reps}} = 5$ times per step.

With 10 hidden units we can represent systems up to six spins with a DKL $\lessapprox 10^{-1}$. Since N_h is fixed, the ratio $N_{\text{params}}/4^N$ goes down exponentially, from 2 for $N = 2$ to 0.02 for $N = 6$. Thus, all RBMs except for the smallest are underparameterized when assuming a generic state. Note that the quantum fidelity shown in fig. 4.4b is not a good measure in case of the SIC-POVM, since negative eigenvalues lead to fidelities larger than one.

Figure 4.4c and fig. 4.4d show the magnetizations $\langle\sigma_a\rangle = \sum_i \langle\sigma_a^i\rangle/N$ and correlators $C_{aa}(d) = \sum_i \langle\sigma_a^i \sigma_a^{i+d}\rangle/N$ for the largest state $|1\rangle^{\otimes 6}$. The light blue shade indicates

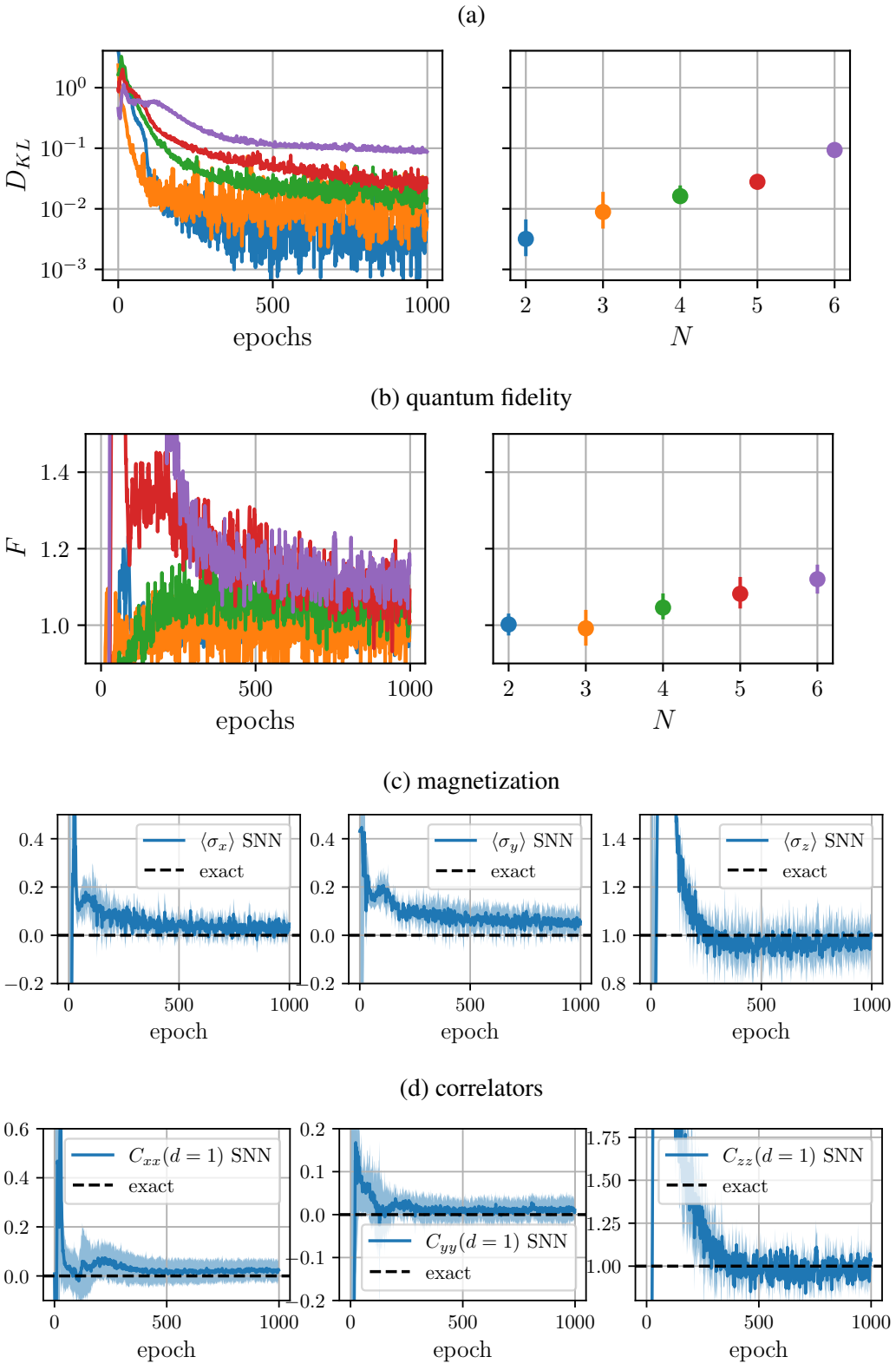


Figure 4.4: Learning the product state $|1\rangle^{\otimes N}$ using an RBM with $N_h = 10$ and $N_{\text{sample}} = 10^5$.

the variance over sites. Despite the large DKL these observables lie close to the exact values. However, deviations exist as in case of observables in x- and y-direction which lie systematically above the exact values.

4.2.3 Entangled states

Entangled systems, in contrast to product states, exhibit strong correlations. The Bell pairs $|\psi_\pm\rangle = (|01\rangle \pm |10\rangle)/\sqrt{2}$ and $|\phi_\pm\rangle = (|00\rangle \pm |11\rangle)/\sqrt{2}$ are prime examples of entangled quantum states. In fact, they are maximally entangled since when one spin is measured, the state of its partner is known with certainty. It is well-known that these states exhibit correlations that are classically forbidden since they violate Bell inequalities [Bel64].

Intuitively, the strong quantum correlations are expected to carry over to the probabilistic description which would manifest as correlated spike patterns in the SNN. If this is so the finite correlation strength in an analog system like BSS2 could potentially curtail the representability of such states.

We reproduced experiments of [CBB+21] where the Bell state $|\psi_{\text{Bell}}\rangle = |\phi_+\rangle$ and its N-particle generalization $|\psi^N_{\text{GHZ}}\rangle$ was encoded on the BSS2 system. Besides the RBM two other architectures were used. The partially-restricted BM (PRBM) has additional intra-layer interactions between visible nodes. The two-layer BMs (dubbed "deep BMs", DBM) shown in fig. 4.5 retain the intra-layer restriction and add an additional hidden layer that only interacts with the first.

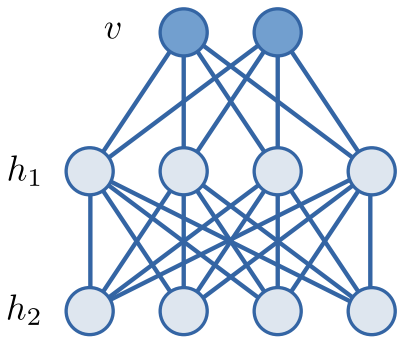


Figure 4.5: BM with two restricted hidden layers (DBM)

The number of parameters N_{params} depends differently on the number of hidden units for each architecture as shown in table 4.1.

Learning the Bell state $(|00\rangle + |11\rangle)/\sqrt{2}$

Figure 4.6a compares four architectures in learning the Bell state based on the number of parameters required to reach a certain DKL. The number of hidden units

Table 4.1: Total number of parameters as function of the number of visible N_v and hidden N_h units per architecture.

	RBM	PRRBM	DBM
N_{params}	$N_v N_h + N$	$N_v(N_v+1)/2 + N_v N_h + N$	$N_v N_{h_1} + N_{h_1} N_{h_2} + N$

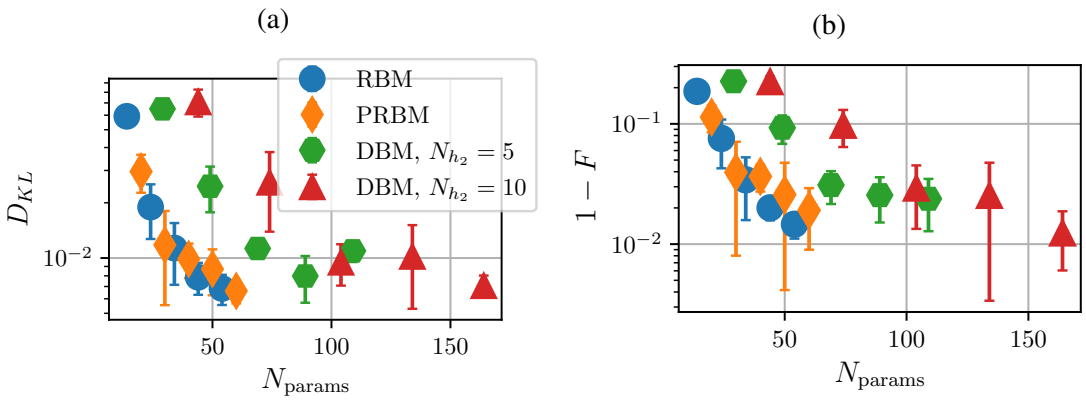


Figure 4.6: Learning the Bell state with different architectures. Statistics are over 5 training runs drawing $N_{\text{sample}} = 2 \cdot 10^5$ samples per epoch.

were chosen from $N_h \in \{2, \ldots, 10\}$. In case of the DBMs only N_{h_1} was varied in two realizations with $N_{h_2} \in \{5, 10\}$. Per epoch $N_{\text{reps}} = 6$ standard sample runs were performed. The data shown are median values with 15$^{\text{th}}$/85$^{\text{th}}$ percentiles across 5 training runs.

We observe that all architectures are able to reach DKL $= 6 \cdot 10^{-3}$. However, for this the shallow networks (RBM, PRBM) require only around 50 parameters, while the DBMs need $\gtrapprox 100$. This is surprising when assuming that a larger number of parameters translates directly to representational power. However, one possible explanation is that the h_1-layer constitutes an information bottleneck for the h_2-layer for $N_{h_1} < N_{h_2}$, such that the additional hidden units can not be fully utilized. Accordingly full representational power would be reached at $N_{h_1} = N_{h_2} = 5$ with $N_{\text{params}} = 59$ and $N_{h_1} = N_{h_2} = 10$ with $N_{\text{params}} = 164$. The additional parameters in the PRBM seem to be put to use since its curve does not deviate significantly from that of the RBM.

The quantum fidelity is shown in fig. 4.6b despite not being a good measure when using SIC-POVMs. However, in contrast to the learning behavior for the product state in fig. 4.4b here it nicely approaches one from below. Due to this correlation with the DKL it seems reasonable to assert that a fidelity of around 98% can be achieved. In any case, we are interested in the capacity to correctly capture observables which is shown for larger systems below.

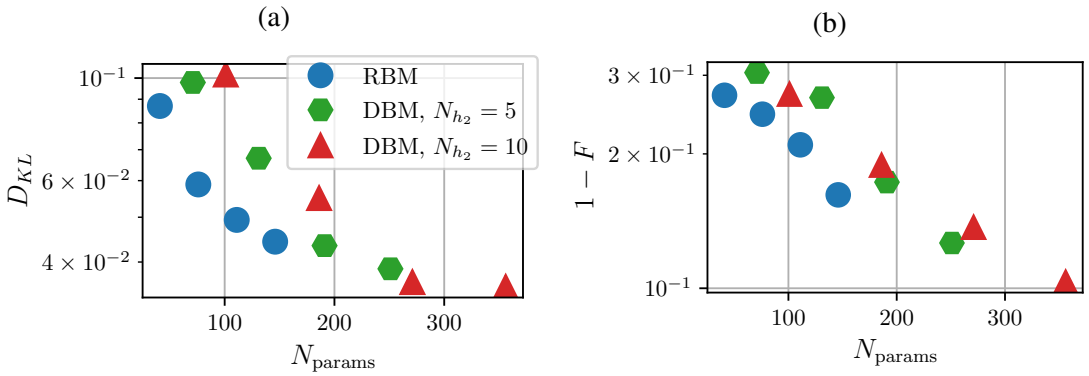


Figure 4.7: Learning the GHZ state with different architectures. Final mean of single run with $N_{\text{sample}} = 4 \cdot 10^5$ samples per epoch is shown.

The GHZ state $(|000\rangle + |111\rangle)/\sqrt{2}$

For the GHZ state we ran the same experiment with $N_h \in \{5, 10, 15, 20\}$ and an increased number of $N_{\text{reps}} = 12$ sampling repetitions. Figure 4.7 shows the resulting parameter dependence of DKL and quantum fidelity.

The gap between RBM and DBM curves is again visible, reaching around DKL $\approx 4 \cdot 10^{-2}$ and $F \approx 90\%$. This is half an order of magnitude worse compared to the equally sized product state.

One aspect of working with a constantly evolving system like BSS2 is that the outcome of experiments also varies over time. Due to the numerous adjustments on different software and hardware levels it is difficult to infer which exactly are responsible for a certain observation in the user accessible observables. Figure 4.8 which exemplifies these effects in the learning capability of the GHZ state aggregates over all these changes.

One finds that over the course of the year 2020 there was a significant improvement in terms of the DKL and fidelity reached for a fixed number of hidden units. We attribute the main improvement to a chip replacement that happened in October and to the switch from chip version 1 to version 2 which happened in late November. The reference are data from Czischek et al. [CBB+21] taken in April.

Despite the improvement, we cannot overlook the saturation of the DKL which starts at $N_h = 30$. This is not to be expected from equivalent numerical simulations and is connected to technical limitations of LIF sampling on BSS2.

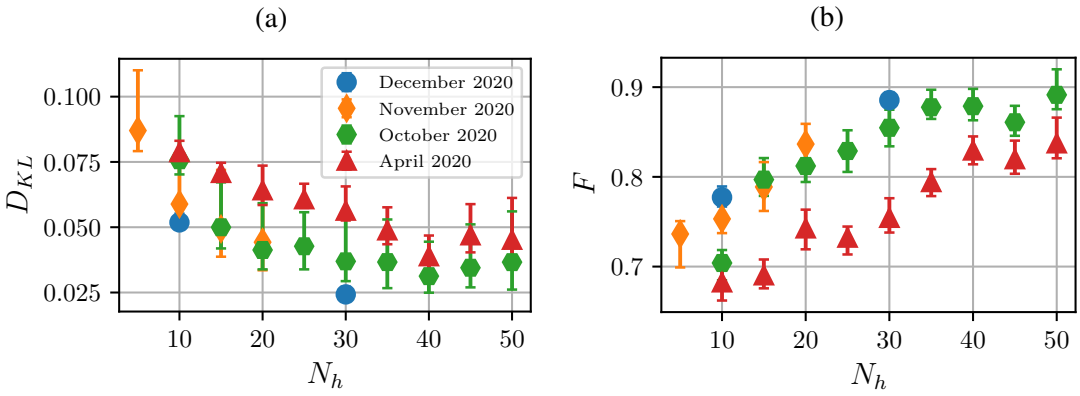


Figure 4.8: DKL and quantum fidelity after learning the GHZ state with an RBM with N_h hidden units over the course of the year 2020 (April 2020 are the data from [CBB+21]

The hardness of four entangled spins $(|0000\rangle + |1111\rangle)/\sqrt{2}$

To further examine the scaling for strongly entangled states, we looked at $|\psi^4_{\text{GHZ}}\rangle$. Figure 4.9 shows a single experiment with $N_h = 30$ and $N_{\text{reps}} = 24$, thus doubling the number of samples for the network size where performance saturated for the GHZ state.

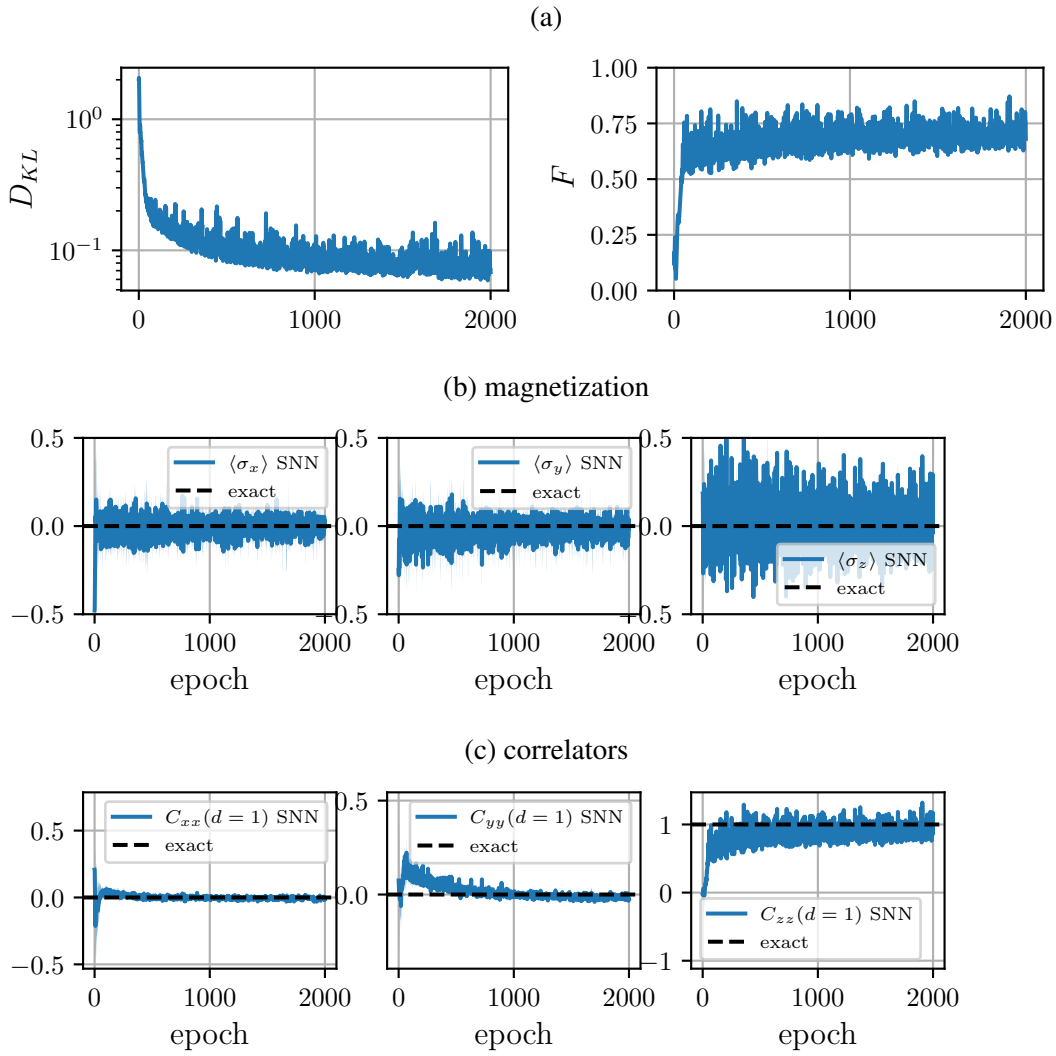


Figure 4.9: Learning the GHZ-like state $|\psi^4_{\text{GHZ}}\rangle$ with an RBM with $N_h = 30$ and $N_{\text{reps}} = 24$.

Still the metrics further deteriorate only reaching DKL $\approx 10^{-1}$ and $F \approx 70\%$ which is almost an order of magnitude worse than the equally sized product state. Even though the ansatz is overparameterized, it turns out to be harder to learn GHZ states than it is to learn product states with with an underparameterized model. This suggests that the SIC-POVM distriutions of entangled states indeed exhibit more complex correlations.

Despite the worse DKL, magnetizations and correlators are learned well and quickly. One observes that compared with the product state example the variance between sites is smaller. Instead the noise between training steps is quite large, in particular, for the z-direction.

4.2.4 Ground states of the TFIM

Product states and strongly entangled states are the extreme ends of a spectrum in terms of correlation strength. We will now turn to the intermediate regime, specifically, we will look at the ground states of the TFIM at the quantum critical point $h/J = 1$.

The result for learning the $N \in \{3, 4, 5\}$ ground state expressed with the SIC-POVM is shown in fig. 4.10. An RBM with $N_h = 40$ was used with $N_{\text{reps}} = 10$ sample repetitions.

For $N = 3$ one finds a DKL similar to that for the GHZ state. However, a DKL $\approx 10^{-1}$ is only reached with $N = 5$, roughly suggesting that the ground state is easier to represent than GHZ states, while it is harder than product states.

Again we find a correlation between DKL and quantum fidelity which hints that the ground state and its observables are better represented here compared to the equally sized GHZ state which shows lower fidelity.

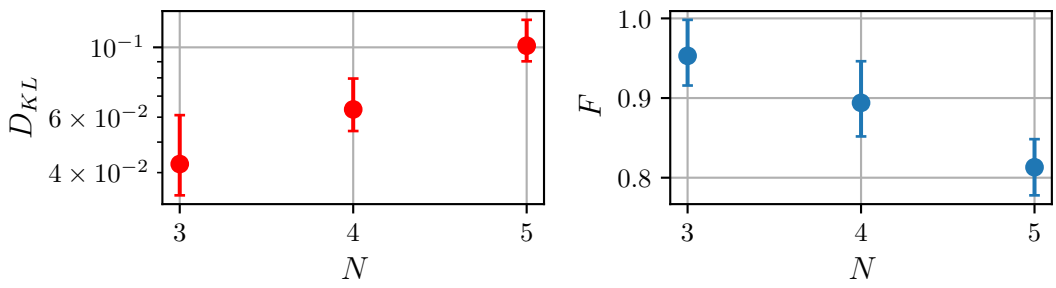
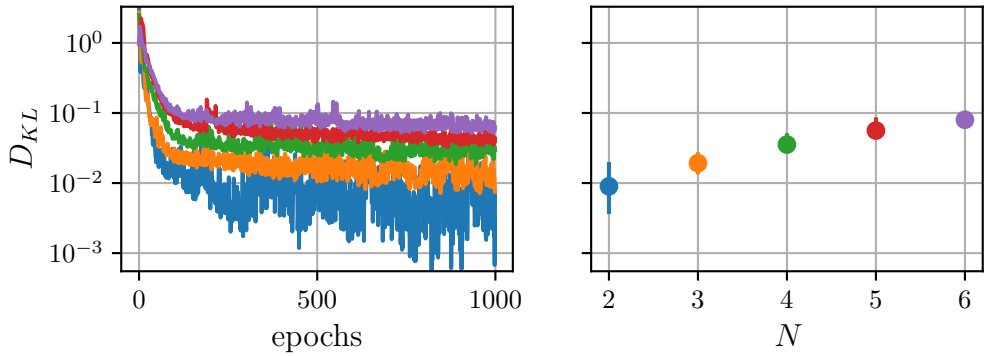


Figure 4.10: Learning SIC-POVM representation of the TFIM ground state at the critical point with an RBM, $N_h = 40$, $N_{\text{sample}} = 2 \cdot 10^5$. Single runs with statistics over last 200 epochs.

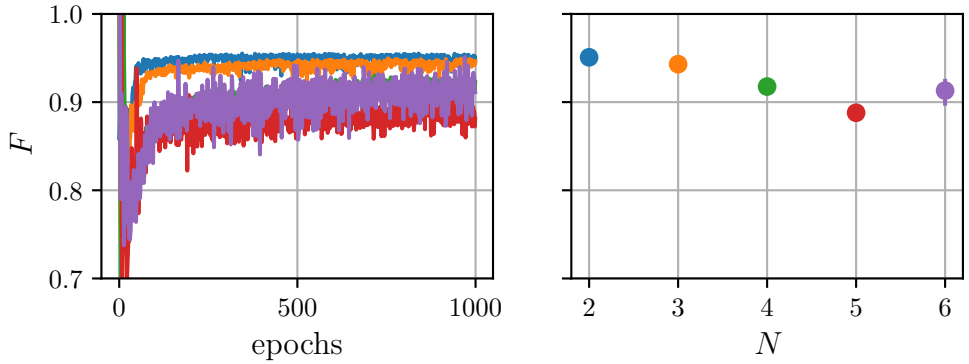
4.2.5 Steady states of the TFIM

So far, we have only dealt with pure states. Since SIC-POVMs are built for a general mixed state setting we have employed them to represent steady states of the TFIM.

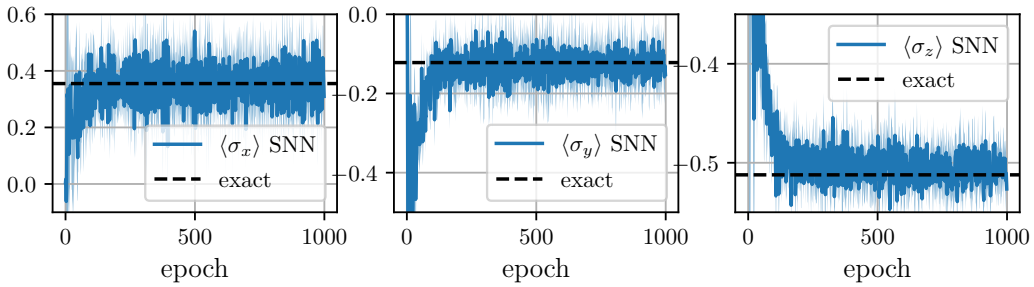
For this we assumed a Lindbladian time evolution as in eq. (2.6) with local jump operators $\Gamma_k = \sigma^-$ and homogeneous decay rates $\gamma_k \equiv \gamma$. The steady state ρ_s is reached in the long time limit as the unique eigenstate of the Liouvillian operator with eigenvalue zero: $\mathcal{L}\rho_s = 0$. As initial state we chose the product $|1\rangle^{\otimes N}$. With these settings we used the default steady state solver of the *QuTiP* library [JNN12] to obtain the target states.



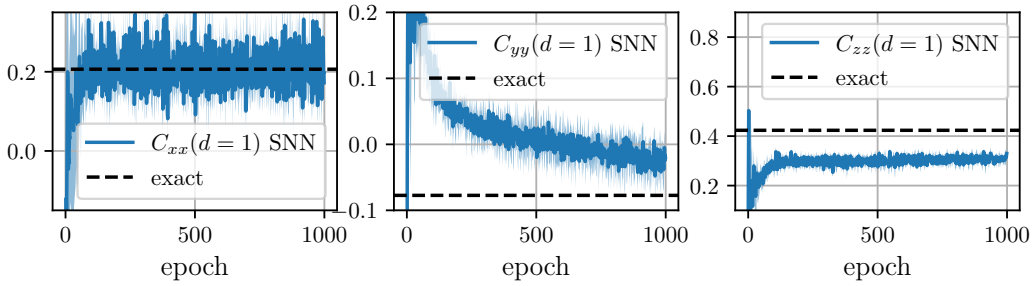
(a)



(b) quantum fidelity



(c) magnetization



(d) correlators

Figure 4.11: Learning curves for steady state of the TFIM with σ^- dissipation at $\gamma = 0.5$ starting in the product $|1\rangle^{\otimes N}$

Figure 4.11 shows results for the critical point $h/J = 1$ and $\gamma = 0.5$. The architecture is an RBM with $N_h = 30$ and $N_{\text{reps}} = 10$.

As in the previous cases the DKL increases quickly when going to larger systems while keeping N_h fixed reaching a DKL $\approx 10^{-1}$ for $N = 6$. While magnetizations are well captured, the two-body correlators for y- and z-direction deviate significantly from the numerical values. This is not an outlier when viewing the full set of experiments that we have done which cover dissipation rates $\gamma \in \{0.25, 0.5, 0.75, 1.0\}$.

4.2.6 TFIM ground states in the *z*-basis

Using the SIC-POVM we have only been able to learn systems with very limited sizes. There are many reasons for this which we will discuss in more detail in chapter 6. For now we will instead replace the representation and view the TFIM ground states in the z-basis as pure states with non-negative real amplitudes. With this simplification the number of spins N corresponds exactly to the number of visible units N_v. By definition this representation encodes well-defined quantum states and hence the quantum fidelity is a valid distance measure.

Using an RBM with $N_h = 20$ hidden units and $N_{\text{reps}} = 10$ sample repetitions we targeted the TFIM ground states with $N \in \{3, \ldots, 10\}$. Thus, system sizes $N > 6$ are underparameterized with respect to the number of wave function coefficients.

Figure 4.12 shows that the representation stays well below a DKL $= 10^{-1}$ and above a fidelity of $F = 99\%$ for the largest system with $N = 10$. In the next chapter, we take this good representability as the basis for a neuromorphic implementation of VGS.

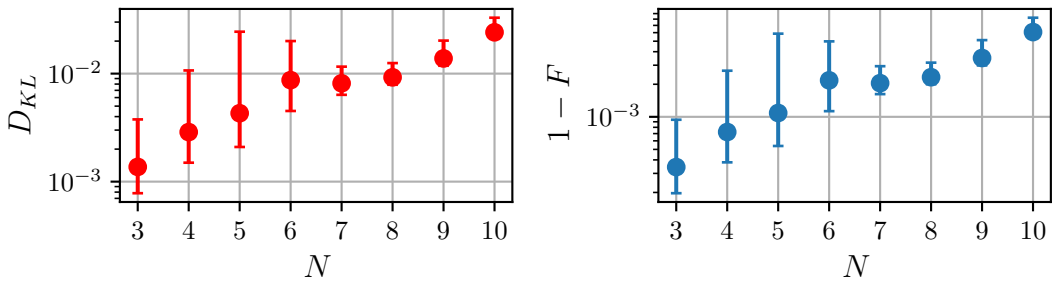


Figure 4.12: Learned z-basis representation of the TFIM ground state at the critical point with $N_h = 20$, $N_{\text{sample}} = 2 \cdot 10^5$. Single runs with statistics across last 200 epochs.

4.3 Discussion

In this chapter we got an impression of what is currently possible in terms of quantum state representations with BSS2.

We found that the Bell state, as a simple benchmark for SIC-POVMs, can be learned very accurately with just $N_h = 4$ hidden units, with the maximum representation quality of larger system sizes quickly degrading. Since we have not exhaustively scanned the regime of large number of hidden units (> 30 - 40) there is still a possibility that marginal improvements can be made.

However, we cannot hide the fact that, for example, the GHZ state is not learnable with a DKL of less than 10^{-2} even when in the overparameterized regime and providing thousands of samples per configuration. Suboptimal mixing behavior is also not a plausible explanation, since samples were aggregated across multiple Markov chains by repeated sampling experiments.

This suggests that the underlying limitation is not related to the theoretical representation power of the BM that is being emulated. Instead, the cause seems to lie in technical limitations of the sampling process on BSS2 which become more severe in the case of large target distributions and/or many hidden units.

5 Searching and Finding Ground States

In the following chapter we will present our results on utilizing the BSS2 chip for variational ground state search. First, the straightforward gradient-based learning algorithm for the z-basis state representation is discussed. Based on software simulations with Gibbs sampling we compare the first-order gradient method to the natural gradient and discuss the difference between 01 and -11 representation. With BSS2 in the loop we apply our method to the TFIM, evaluating the energy difference and relevant observables for the phase transition. Throughout a RBM ansatz was used where the visible units are directly mapped to spin configurations.

5.1 An energy minimization algorithm for BrainScaleS-2

In the previous chapter we focused on representing known target distributions by minimizing its DKL to the state encoded in the BMs. In this chapter we will apply almost the same algorithm to learn a target state that is, in principle, *unknown* for large system sizes, namely the ground state of a given Hamiltonian H.

In the following we assume H to be a stoquastic Hamiltonian such that its ground state wave function has real and non-negative amplitudes. Then our variational ansatz for the ground state by a probability distribution over the z-basis does not loose phase information: $|\psi_\theta\rangle = \sum_{\boldsymbol{v}} \sqrt{p_\theta(\boldsymbol{v})}|\boldsymbol{v}\rangle$. For our purposes, $p_\theta(\boldsymbol{v}) \equiv p(\boldsymbol{v})$ is the marginal distribution of the BM emulated by the SNN on BSS2 and as such is normalized.

The objective function which we minimize is the variational energy of the encoded state $|\psi_\theta\rangle$: $C(\theta) = E_\theta = \langle\psi_\theta|H|\psi_\theta\rangle$ The derivative of E_θ with respect to a connection parameter W_{ij} is readily obtained:

$$\partial_{W_{ij}} E_\theta = \partial_{W_{ij}} \sum_{\boldsymbol{v}\boldsymbol{v}'} \sqrt{p(\boldsymbol{v})p(\boldsymbol{v}')} H_{\boldsymbol{v}\boldsymbol{v}'} \tag{5.1}$$

$$= \sum_{\boldsymbol{v}\boldsymbol{v}'} \sqrt{\frac{p(\boldsymbol{v}')}{p(\boldsymbol{v})}} H_{\boldsymbol{v}\boldsymbol{v}'} \partial_{W_{ij}} p(\boldsymbol{v}) \tag{5.2}$$

$$= \sum_{\boldsymbol{v}\boldsymbol{v}'} \sqrt{\frac{p(\boldsymbol{v})}{p(\boldsymbol{v}')}} H_{\boldsymbol{v}\boldsymbol{v}'} \left(\sum_{\boldsymbol{h}} z_i z_j p(\boldsymbol{v}') - p(\boldsymbol{v}) \sum_{\boldsymbol{z}'} z'_i z'_j p(\boldsymbol{z}') \right) \tag{5.3}$$

$$= \sum_{\boldsymbol{v}} \sum_{\boldsymbol{h}} E^{\mathrm{loc}}_{\boldsymbol{i}} z_i z_j p(\boldsymbol{z}) - \sum_{\boldsymbol{i}} E^{\mathrm{loc}}_{\boldsymbol{v}} p(\boldsymbol{v}) \sum_{\boldsymbol{z}'} z'_i z'_j p(\boldsymbol{z}') \tag{5.4}$$

$$= \sum_{\boldsymbol{z}} \left(E^{\mathrm{loc}}_{\boldsymbol{v}} - E_\theta \right) z_i z_j p(\boldsymbol{z}) \tag{5.5}$$

$$= \left\langle \left(E^{\mathrm{loc}}_{\boldsymbol{v}} - E_\theta \right) z_i z_j \right\rangle_{p(\boldsymbol{z})} \tag{5.6}$$

Analogous to the above, the derivative with respect to bias parameters b_k is given by:

$$\partial_{b_k} E_\theta = \left\langle \left(E^{\mathrm{loc}}_{\boldsymbol{v}} - E_\theta \right) z_k \right\rangle_{p(\boldsymbol{z})} \tag{5.7}$$

From eq. (5.2) to eq. (5.3) we have used the symmetry of the Hamiltonian. In eq. (5.4) the local energy $E^{\mathrm{loc}}_{\boldsymbol{v}} = \sum_{\boldsymbol{v}'} H_{\boldsymbol{v}\boldsymbol{v}'} \sqrt{p(\boldsymbol{v}')}/\sqrt{p(\boldsymbol{v})}$ is introduced and the variational energy appears in eq. (5.5) due to the relation $E_\theta = \sum_{\boldsymbol{v}} E^{\mathrm{loc}}_{\boldsymbol{v}} p(\boldsymbol{v})$.

To deal with the numerical problem of a vanishing entry $p_\theta(\boldsymbol{v})$ we have introduced a small parameter ϵ which essentially introduces a bias towards a uniform distribution: $E^{\mathrm{loc}}_{\boldsymbol{v}} = \sum_{\boldsymbol{v}} H_{\boldsymbol{v}\boldsymbol{v}'} \sqrt{p_{\boldsymbol{v}'} + \epsilon}/\sqrt{p_{\boldsymbol{v}} + \epsilon}$. This is typically choosen as $\epsilon = 10^{-12}$.

With this expression we adapt the target learning algorithm 3 by replacing the gradient calculation to obtain algorithm 4 which performs VGS. Note that the local energy is calculated by means of the full histogram of visible configurations such that the same caveats in terms of scalability apply. Also the variational energy is a global quantity of the model and as such this learning rule can not be expressed in terms of local updates.

In order to track the accuracy of the algorithm, the exact ground states $|\psi_0\rangle$ and their energies E_0 are obtained with exact diagonalization. While low energy deviations $|E - E_0|$ are an indication that the algorithm has found the ground state, we also consider the DKL between the exact target distribution and the encoded distribution and their state overlap, i.e. the quantum fidelity.

Stochastic reconfiguration vs ADAM

In the previous chapter, we had already excluded DE, the gradient-free optimizer and selected the ADAM optimizer instead due to its efficiency and fast convergence.

Algorithm 4: Gradient-based in-the-loop learning of variational ground states for N_{epoch} steps. At every step N_{sample} configurations are generated using BSS2.

calibrate chip setup ; // BSS2
provide Hamiltonian H; // Host
initialize parameters θ_1 ; // Host
for $e \in \{1, \cdots, N_{\text{epoch}}\}$ **do**
    configure parameters θ_e on chip ; // BSS2
    sample model data $\mathcal{S} = \{\boldsymbol{z}\}_s^{N_{\text{sample}}} \sim p_\theta(\boldsymbol{z})$; // BSS2
    create histogram $p_\theta(\boldsymbol{v}) = \frac{1}{N_{\text{sample}}} \sum_{\boldsymbol{z}_s \in S} \mathbb{1}\left[\boldsymbol{v}_s = \boldsymbol{v}\right]$; // Host
    calculate local energy $E_{\boldsymbol{v}}^{\text{loc}} = \sum_{\boldsymbol{v}'} H_{\boldsymbol{v}\boldsymbol{v}'} \sqrt{\frac{p(\boldsymbol{v}')+\epsilon}{p(\boldsymbol{v})+\epsilon}}$; // Host
    calculate gradient $\Delta\theta_e = \partial_\theta E_\theta$; // Host
    update parameters $\theta_{e+1} \leftarrow \theta_e - \eta_e o(\Delta\theta_e)$; // Host
generate samples after convergence ; // BSS2
evaluate metrics and expected values ; // Host

However, in previous publications, stochastic reconfiguration (SR), which for our approach is equivalent to natural gradient descent (section 3.4.1), has often been used in the context of energy minimization, in particular, in the case of complex RBMs [CT17], [BSD20]. We have therefore also tested this optimizer in software for our approach with real RBMs where SR is equivalent to NGD.

Figure 5.1 shows both optimizers employed in algorithm 4 to the TFIM with $N = 5$ at the critical point. To regularize the Fisher matrix, we used Tikhonov regularization [TGSY95] with an initial $\epsilon_{\text{reg}} = 1 \times 10^{-4}$ and a decay of $\gamma_{\text{reg}} = 0.99$. We used standard parameters for ADAM, Gibbs sampling and initialization (see appendix B) and for both optimizers an exponential learning rate schedule with initial value $\eta = 0.1$ was employed. A RBM with $N_h = 5$ slightly overparameterizes the problem in order to just compare optimization performance.

Indeed, we find a faster convergence of the natural gradient method, both for exactly calculated updates and for those estimated with samples. ADAM requires about an order of magnitude more steps to achieve the same absolute energy errors per spin.

In both cases, we see possible convergence scattered over orders of magnitude of energy difference. This highlights the weakness of gradient-based methods to get stuck in local minima. In the ADAM learning curves, there is a characteristic step in the first $\mathcal{O}(100)$ steps of all runs that we had not seen in case of target learning ground states. This seems to be an extended flat region of the energy landscape to which first order methods are susceptible, while the second-order natural gradient is not. However, in some natural gradient runs we also see plateaus and step-like learning curves, albeit at much lower energies, indicating that a qualitatively similar problem can occur there.

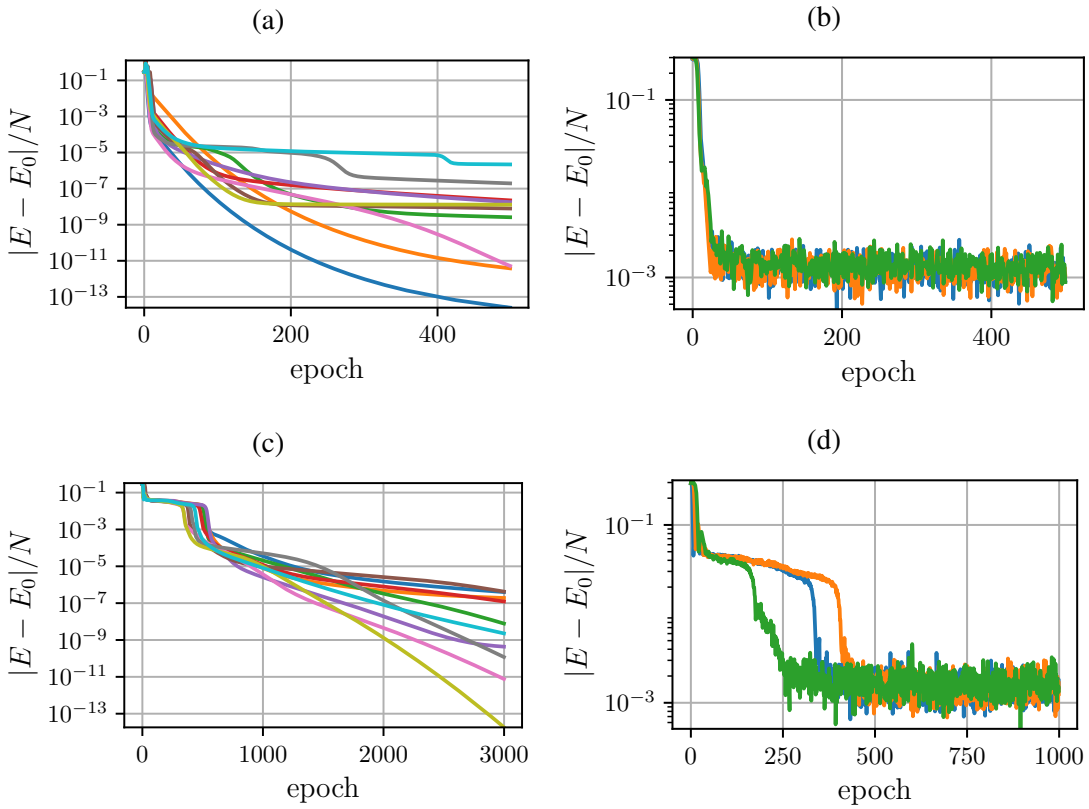


Figure 5.1: Learning the ground state of the TFIM with parameters $N = 5$, $h/J = 1$ using a RBM with $N_h = 5$. (a) Natural gradient descent and (c) ADAM with exact updates (10 initializations) and (b), (d) respectively with Gibbs sampling (3 initializations).

When optimizing based on sample estimates, the energy error is bounded by the amount of samples that are used every epoch, in this case at $\Delta E/_{\text{,}}N \approx 1 \times 10^{-3}$. Thus, only the energy landscape above this barrier plays a role. This is why we again see the step-feature in case of ADAM, while it is invisible to the natural gradient.

We were able to successfully use both methods to learn ground states on BSS2. An exemplary comparison between the two is shown in fig. 5.2 where the ground state for $N = 4$ and $h/J = 1$ was learned with RBMs of similar, overparameterized, size and comparable N_{sample}. Standard optimization parameters were used for ADAM (table B.3), while NGD used an initial $\epsilon_{\text{reg}} = 1 \times 10^{-4}$, a regularization decay of $\gamma_{\text{reg}} = 0.999$ and a learning rate of $\eta = 0.5$.

While NGD learns faster in the beginning it still takes over 500 epochs to converge to $\Delta E/N \approx 10^{-3}$. ADAM, on the other hand, is stuck in its plateau for the first few hundred epochs, but when it finally starts to converge further, it improves quickly, reaching

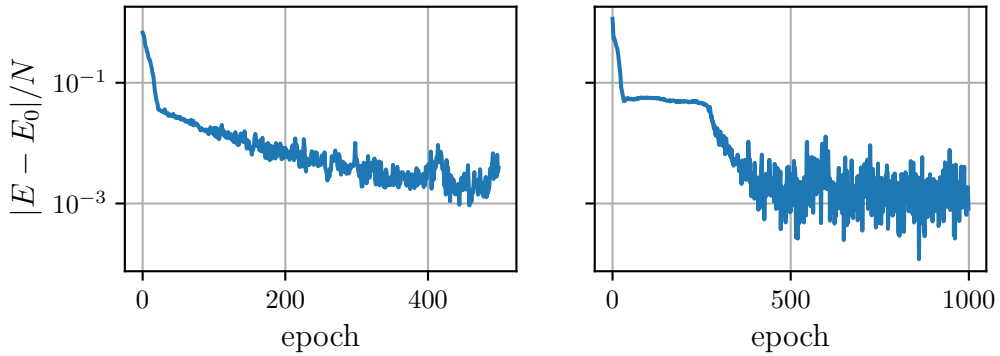


Figure 5.2: Comparing NGD (left) and ADAM (right) for searching the ground state at $N = 4$, $h/J = 1$ with BSS2. Comparable models and sample numbers were used.

equal performance in fewer number of steps.

For larger systems we observed that both NGD and ADAM take more steps to converge. However, NGD is also slower in terms of wall-clock time due to the additional burden of sampling and inverting the FIM.

Therefore we decided against using NGD which has the further disadvantage that the regularization parameters might depend on the system size requiring additional finetuning.

Differences between neural- and spin-units

In principle, it is not important which numerical values are attributed to the binary states in the BM, since these can be converted into each other with the help of a linear transformation. If one speaks of neural units $z_{\text{neural}} \in \{0, 1\}$ is meant, if it is about spins $z_{\text{spin}} \in \{-1, 1\}$ is natural.

We have tested both possibilities for algorithm 4 in numerical simulations. The ADAM optimizer with standard parameters (table B.3) is used for both settings and parameter updates as well as distributions are calculated exactly for TFIM $N = 5$, $h/J = 1$ with a RBM with $N_h = 5$. Both simulations were repeated 10 times with standard initialization.

While we again see plateaus for "neural units" during the first 1000 epochs, these disappear in case of the "spin units". As a result the "spin unit" BMs converge much faster, comparably, to NGD.

Thus, while the computational power is not changed by switching representations, the properties of the energy landscape are evidently changed favorably for this specific optimizer.

The "neural units" are natural for the LIF emulation of BMs. This is because the ac-

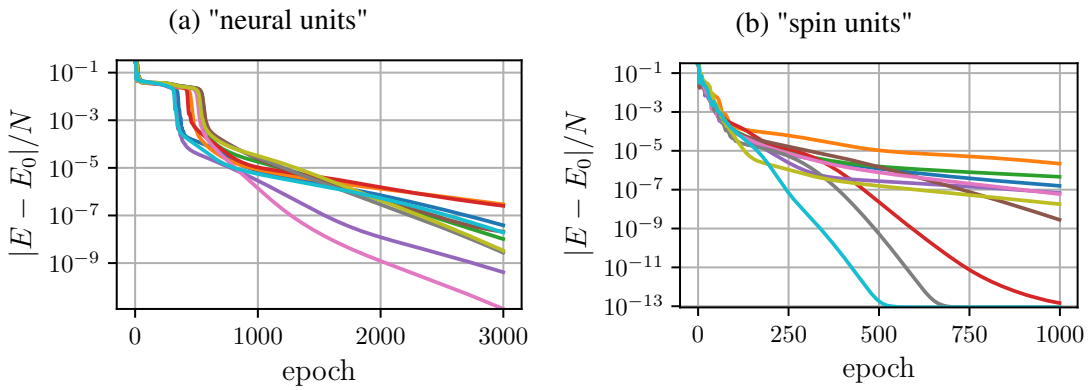


Figure 5.3: Learning the ground state at $N = 5$, $h/J = 1$ using a RBM with (a) "neural units" $\{0, 1\}$ and (b) "spin units" $\{-1, 1\}$, $N_h = 5$, exact updates and distributions, 10 initializations.

tivity of spiking neurons ranges between silent ("zero") and bursting ("one"). However, one can still utilize "spin units" while calculating gradients and bookkeeping parameters on the host computer. When configuring the hardware one just has to translate $z_{\text{neural}} = 0.5(z_{\text{spin}} + 1)$.

We have tested this approach and found it to work as expected to converge slightly faster for small systems ($N < 4$). However, for larger systems no gains were observed and the translation sometimes completely failed to converge. We concluded that the additional conversion is not worthwhile.

5.2 Results for the TFIM

In this section we present results on applying the algorithm introduced above to search for ground states of the TFIM. First, we focus on learnability in terms of system sizes and required model capacity. We then proceed to evaluate relevant observables to the phase transition of the TFIM.

Finding ground states at the critical point

In section 4.2.6 we had seen that we can represent ground states up to at least $N = 10$ in the z-basis with BSS2. Employing algorithm 4 we checked whether it is possible to *reach* these representations when minimizing the energy.

In fig. 5.4 the number of spins N is increased from $N = 3$ to $N = 10$ for $h/J = 1$ while keeping the number of hiddens (roughly) fixed. Learning is performed with an RBM with $N_h = 40$ and $N_{\text{sample}} = 2 \cdot 10^5$ samples are drawn in each epoch for $N = \{3, \ldots, 8\}$ and respectively $N_h = 50$ and $N_{\text{sample}} = 4 \cdot 10^5$ for $N \in \{9, 10\}$. Even though

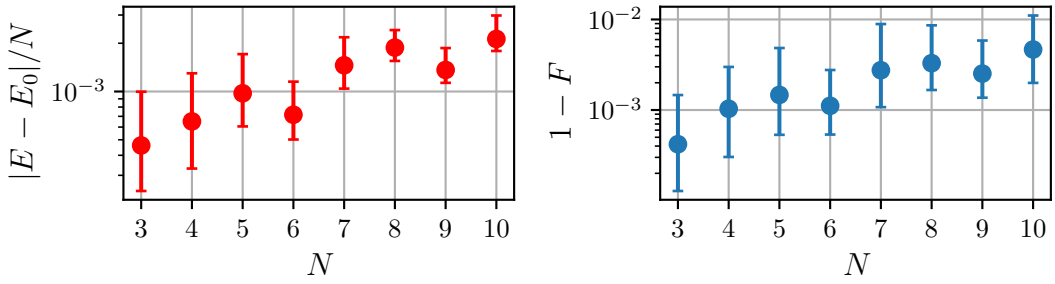


Figure 5.4: Median energy difference and infidelity over last 200 epochs after convergence of ground state search at the critical point. For $N \in \{3, \ldots, 8\}$ $N_h = 40$ and $N_{\text{sample}} = 2 \cdot 10^5$ and for $N \in \{9, 10\}$ $N_h = 50$, $N_{\text{sample}} = 4 \cdot 10^5$.

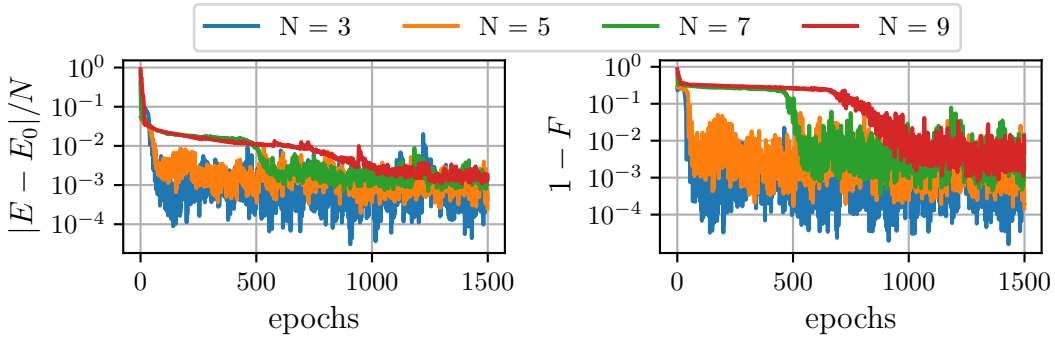


Figure 5.5: Learning curves of energy difference and infidelity for ground state search in the TFIM at the critical point. For $N \in \{3, \cdots, 8\}$ $N_h = 40$ and $N_{\text{sample}} = 2 \cdot 10^5$ and for $N \in \{9, 10\}$ $N_h = 50$, $N_{\text{sample}} = 4 \cdot 10^5$.

the increased number of hidden neurons and samples for sizes $N = 9, 10$ was necessary to ensure convergence of the learning algorithm, compared to $N < 9$ the RBM has less parameters than the number of wave function coefficients ($509 < 2^9$ and $560 < 2^{10}$).

Again a quantum fidelity greater than 99% can be achieved and even up to 99.9% for systems $N < 6$. However, the curve did not move noticeably downward despite the use of twice as many neurons as compared to fig. 4.12. This hints again at the underlying limitations which cause the representability to decrease with larger systems and distributions.

Looking at the learning curves in fig. 5.5 we recognize the plateaus predicted by our numerical simulations. These cause the optimization to take progressively longer as the system grows. Note that the number of epochs was cut at 1500, although up to 2500 steps were calculated for $N = 9, 10$.

Since the fidelity imposes a bound on all expected value errors it is reasonable to ex-

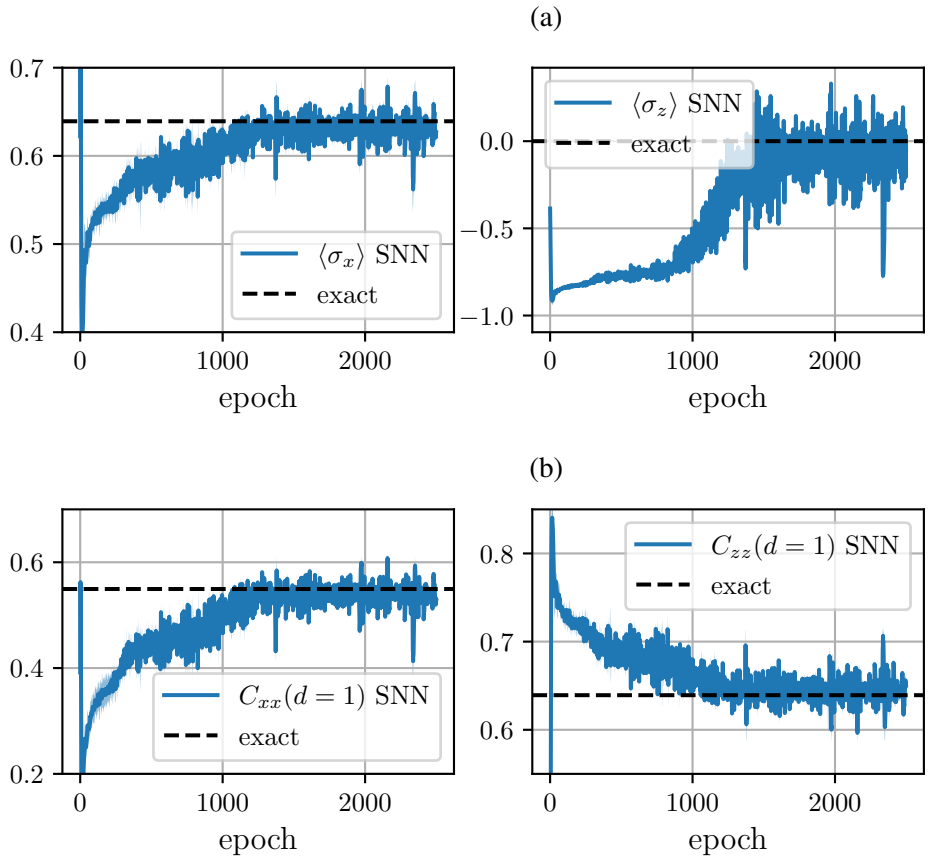


Figure 5.6: (a) magnetization and (b) two-body correlator $C_{aa}(d = 1)$ during learning of the TFIM ground state with $N = 10$.

pect good agreement of the learned observables. Figure 5.6 verifies this for $N = 10$ with magnetizations and two-body correlators in x- and z-direction. Note that the expected values in y-direction are identically zero with zero variance since we are working with real valued wave function coefficients.

Dependence on the model complexity

In order to assess the required number of hidden units for a good variational representation depending on the system size we have performed a grid search over $(N, N_h) = (\{3, \ldots, 8\}, \{5, 10, 20\})$ drawing $N_{\text{sample}} = 2 \cdot 10^5$ samples for 1500 epoch each.

The results are shown in fig. 5.7 in terms of median energy error per spin and fidelity over the last 200 epochs. While $N_h = 5$ is sufficient to accurately describe the systems for $N < 6$, above that a sharp drop in fidelity to below 90% and respective increase in energy error is observed. Increasing to $N_h = 10$ enables the model to also capture the

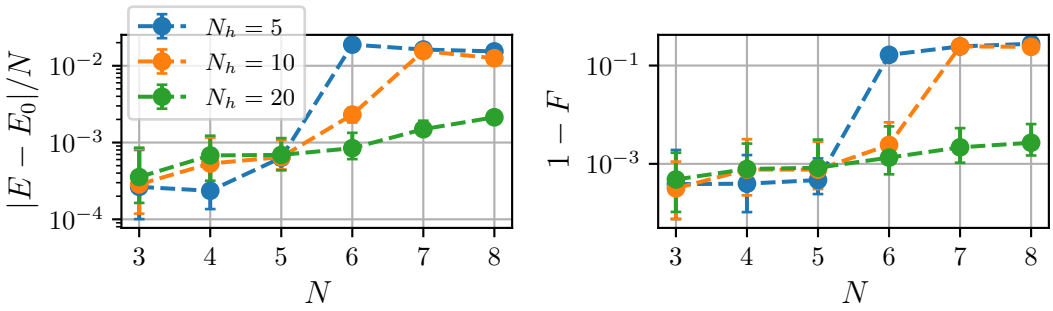


Figure 5.7: Median energy difference and infidelity after convergence for ground state search in the TFIM at the critical point. For $N \in \{3, \cdots, 8\}$, $N_h \in \{5, 10, 20\}$ and $N_{\text{sample}} = 2 \cdot 10^5$.

Table 5.1: Parameter settings for learning different h/J.

h/J	0.1	0.5	0.9	1.0	1.25	5.0	10.0
$N_{\text{sample}}/10^5$	2	2	4	2	2	2	2
N_h	50	30	40	40	30	20	30
N_{params}	458	278	368	368	278	188	278

ground state of $N = 6$ with $F > 99\%$, while $N_h = 20$ is required for $N = 7, 8$.

One would think that more hidden units could decrease the slope of the green curve ($N_h = 20$) further, also bringing the large systems above 99.9% fidelity. However, comparing with fig. 5.4 where $N_h = 40$ was used there is no significant difference in either energy error or fidelity, suggesting that model capacity is not the dominating limiting factor.

Probing the quantum phase transition

Above we have established that the ground state can be learned well at the critical point. However, due to finite size effects the actual phase transition point can be shifted. Thus, we further tested our approach at additional points along the h/J dimension, specifically $h/J \in \{0.1, 0.5, 0.9, 1, 1.25, 5, 10\}$ in order to observe the quantum phase transition between the ferromagnetic and polarized regimes.

For each data point slightly different network topologies and sampling parameters were used which are summarized in table 5.1. Note that these parameters were not optimized and most models are overparameterized and likely oversampled. For $h = 0.9$ more samples were required in order to learn the symmetric ground state.

In fig. 5.8 the two key observables indicative of this transition are shown, namely, the magnetization $\langle \sigma_x \rangle$ and the correlation length of zz-correlations ξ_{zz}. As the field strength goes from 0.1 to 10 the ground state becomes increasingly polarized in x-

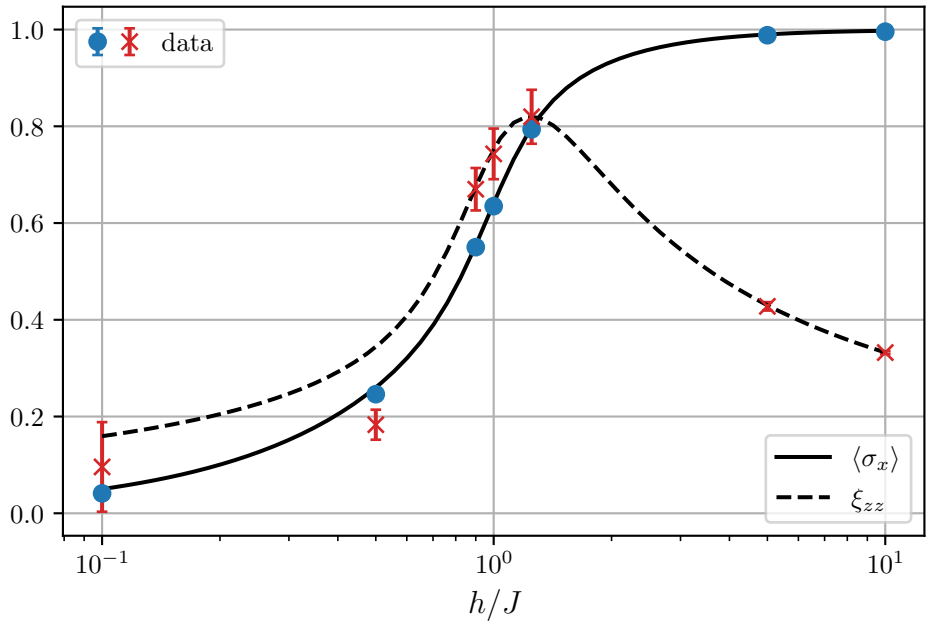


Figure 5.8: Diagram of the phase transition between ordered and disordered regime. The x-magnetization (solid) and fitted correlation length ξ_{zz} (dashed) are shown. In the former case errors are standard deviations over the last 200 learning steps, in the latter case error bars indicate the standard deviation of the parameter fit.

direction.

The x-magnetizations were averaged over the last 200 epochs of learning showing good agreement with the exact solution (solid line).

In fig. 5.9 the spin-spin correlations in z-direction C_{zz} are shown as a function of the spin distance d and one can see non-vanishing correlations due to finite size except in case of the almost fully polarized state at $h/J = 10$ where the spin interactions are dominated. The correlation lengths ξ_{zz} are extracted by fitting the data points of each field strength with the following function (shown as dotted lines),

$$\hat{C}_{zz}(d) = A \exp\left(-d/\xi_{zz}\right) + B \tag{5.8}$$

where the additional parameters A and B account for finite sizes effects.

The fit parameters ξ_{zz} and their standard deviations are shown in fig. 5.8 together with the corresponding theoretical values (dashed line). Due to finite size effects we observe that the correlation length has a maximum at $h/J \approx 1.25$ marking the phase transition point and closely matching the theoretical prediction. While the points $h \geq 0.9$ agree well with both observables, $\langle\sigma_x\rangle$ and ξ_{zz}, the points at $h = 0.1, 0.5$ have significant deviations for the correlation length.

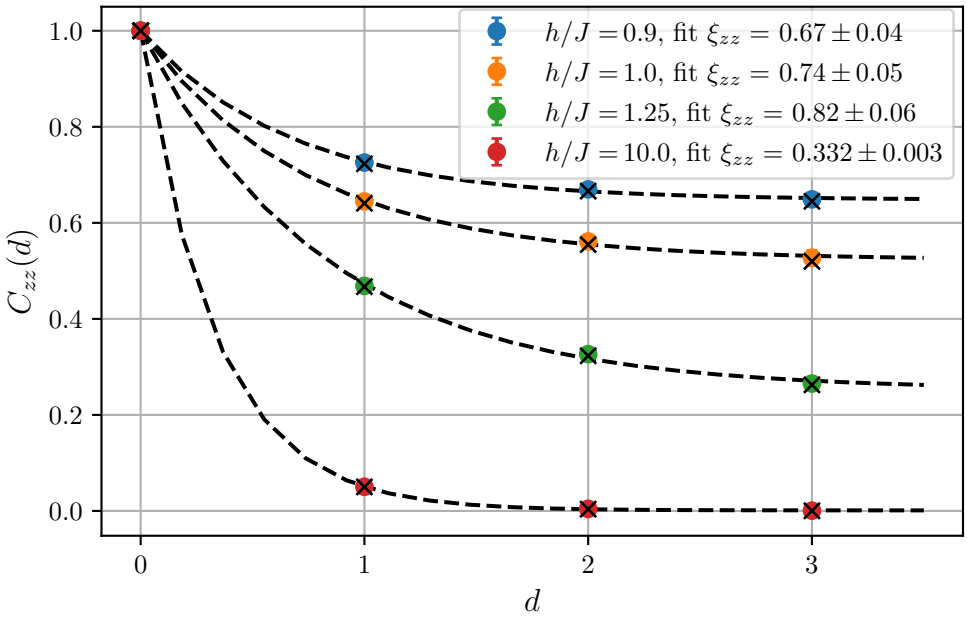


Figure 5.9: The correlator C_{zz} as a function of spin distance d. The exponential correlation decay is characterized by fitting the length scale ξ_{zz}.

The reason for these deviations is that the symmetric ground state distribution of the wave function becomes harder to learn for $h \to 0$ as it then increasingly approaches a GHZ-like superposition of the two extreme configurations $|1\rangle^{\otimes N}$ and $|0\rangle^{\otimes N}$. Unfavorably, such a distribution requires both highly synchronous activity (large excitatory weights) and synchronous inactivity (large inhibitory weights). This hardness manifested itself in an increased need for samples at $h/J = 0.9$ in order to well represent the symmetric ground state. The points below, $h/J = 0.1, 0.5$ are even further in the ferromagnetic phase which makes learning these strong correlations prohibitive. This can be seen in fig. 5.10b where the overlap with the symmetric ground state is significantly decreased. In contrast, fig. 5.10a shows that the relative energy error is, with $\Delta E/|E_0| \approx 10^{-3}$, in line with the remaining points. This behavior can be understood from looking at the learned variational states.

The final distributions at $h/J \in \{0.1, 0.5, 0.9\}$ are compared in fig. 5.11a. It reveals that while the symmetric ground state is learned at $h/J = 0.9$, at lower fields a symmetry broken distribution is obtained that favors the correlated activity of all visible neurons.

While in fig. 5.11b the points $h/J = 0.9, 5$ match the theory curves well, the symmetry broken distributions at low h/J only agree with the outer parts of the positive branch as the occupancy of non-extreme z-magnetization shrinks. The remaining magnetization states are sampled with a roughly constant probability of 10^{-5} corresponding

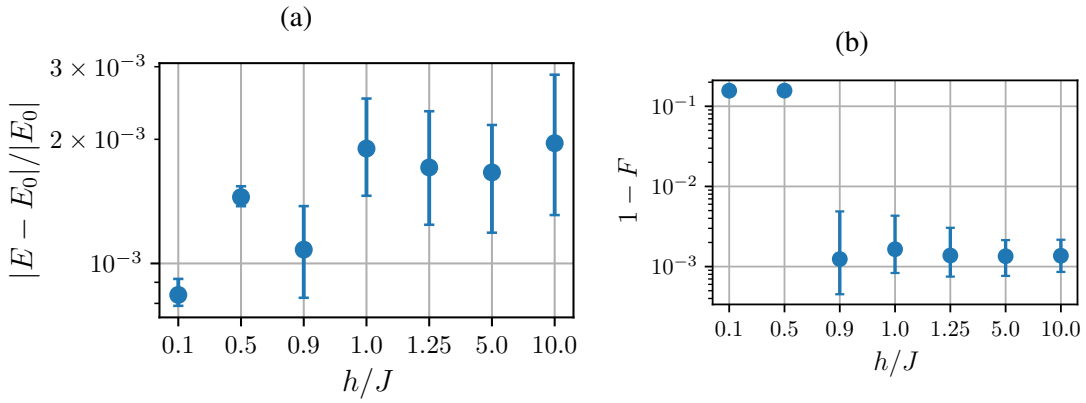


Figure 5.10: (a) Relative energy error and (b) infidelity as a function of h/J

to $\mathcal{O}(1)$ samples. As a result the largest absolute deviations from the symmetric ones occur at $m = 4$ which holds almost all the probability mass compared to less than half in case of the symmetric solution.

Spontaneous symmetry breaking occurs in the TFIM when approaching $h \to 0$ where the energy gap between the two states vanishes enough that small fluctuations of the system are of the same order of magnitude and trigger the local collapse of the superposition into one of the two configurations $|0\rangle^{\otimes N}$ and $|1\rangle^{\otimes N}$.

Where those fluctuations come from and when symmetry breaking sets in depends on the specific substrate that simulates/emulates the TFIM. In our case the standard parameter initialization seems to be biased towards the $|1\rangle^{\otimes N}$ symmetry broken distribution setting up a minimum in the energy landscape which is already favored for $h/J = 0.1, 0.5$. However, we found that the symmetry can also be broken in favor of the $|0\rangle^{\otimes N}$ state by lowering neuron activity with negative biases.

5.3 Discussion

In summary, we have presented a proof-of-principle of neuromorphic VGS for stoquastic quantum spin systems on BSS2. With our method we have analyzed the TFIM phase transition and found good agreement with theory up to $N = 10$. However, there are a few points that constrain the broader applicability of this method.

First, we found that scalability becomes independent of the number of hidden units at around $N_h = 20 - 30$ indicating that an unknown problem is restricting the accessible system sizes. We examine possible technical causes in section 6.1.

Secondly, learning the symmetric ground states became harder as the field strength h was decreased to the point where we observed spontaneous symmetry breaking as outcome of the optimization. The main problem stems from the state encoding in the

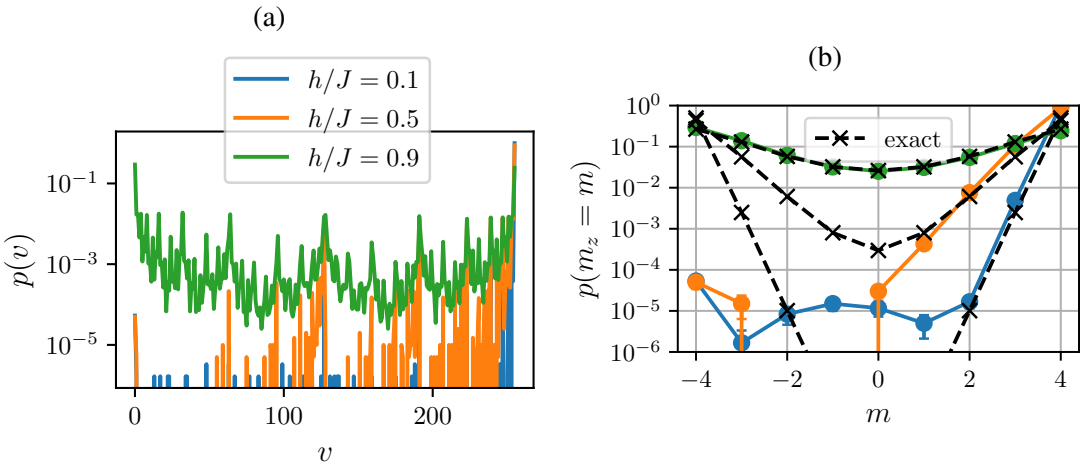


Figure 5.11: Symmetry breaking for low h (a) in the learned distributions and (b) in the distribution of z-magnetization.

z-basis such that extreme correlations are required to represent the ground state in this regime. A possible solution would be to choose a different encoding either on the level of the quantum basis or in terms of the spike coding.

In addition, we assumed the Hamiltonian subject to the ground state search to be stoquastic. This assumption is a strong simplification as it excludes quantum systems where a sign problem can occur. In order to make our ansatz more general we would need to also model the phases of the wave function coefficients, e.g. with an additional SNN like in [TMC+18]. Importantly, the POVM ansatz is not suitable without progress on the question of positivity.

Finally, the presented method is not scalable from the algorithmic perspective due to the reliance on the full set of wave function coefficients. However, the choice of this algorithm is indebted to the current limitation to in-the-loop learning with BSS2 and the lack of a full theory of LIF sampling. Indeed, with the capability of implementing local learning rules on-chip the global learning rule presented in algorithm 4 could be approximated with methods like event-driven constrastive divergence [NDP+14], predictive coding [WB17] or direct feedback alignment [CPGR19, LZZ+20]. Furthermore, given an accurate prescription to translate hardware parameters to LIF parameters to BM parameters one could efficiently compute the wave function coefficents without the need of exponentially many samples like in [CT17].

6 Limitations and Prospects

In the previous chapters we have seen how to represent quantum states (chapter 4) and how to find ground states (chapter 5) with the BSS2 chip. However, there are several challenges that limit the applicability of our methods and in particular their scalability. In this chapter we will look specifically at the technical limitations imposed by the BSS2 chip and the implications for further work on QST and VGS. Finally, we present an idea for steady state search in a dissipative OQS utilizing marginal POVM distributions. This is one of the approaches we have explored to improve the algorithmic scalability of our learning methods.

6.1 Limitations of the BrainScaleS-2 chip

Low precision parameters

On the BSS2 system the analog weights have 6-bit values with an additional sign bit implemented by the physical separation between an excitatory and inhibitory synapse. The biases are 10-bit values which is the resolution at which the leak potential E_l can be configured.

These design choices are not arbitrary, but reflect various trade-offs between chip resources like the size of analog and digital memory that stores these parameters, the inherent variability of analog circuitry and the precision required by applications. Viewed from the application side we are interested in whether and how much the performance of LIF sampling is affected.

All else being equal a lower parameter precision leads to a more course grained space of representable distributions. A continuous random distribution is then increasingly less probable to be well representable as the resolution is decreased. We performed a simple numerical experiment where we randomly initialized RBMs ($N_v = 20$, $N_h = 40$) with different weight magnitudes (dynamic range) $w_{\max}$ and compared the parameters drawn from the continuous distribution with the distributions obtained by various degrees of discretization. The weights are discretized to equidistant values on the interval $[-w_{\max}, w_{\max}]$. For 1-bit only the two interval edges are possible. For x-bit discretization with $x > 1$ the grid of possible values spans $2^x - 1$ states containing 0. Note that all biases were set to the center of the activation functions and kept at 10-bit precision.

Figure 6.1a shows how the discretized distribution exponentially approaches its continuous origin as the precision of the parameters is increased.

When we do this same experiment on BSS2 (fig. 6.1b) we can obviously only increase the resolution up to 7-bit. Also instead of exact distributions we perform LIF sampling for a network of $N_v = 8$, $N_h = 20$. We again find a quick decrease in DKL which, however, plateaus at 6/7-bit precision.

The lack of decrease from 6-bit to 7-bit is surprising since the number of possible states per weight has again doubled. On the other hand, we recognize that at that point we have reached the typical regime of DKL $\simeq 10^{-2}$ where the sampling precision saturates for large networks. In that view, one conclusion is that at the maximum precision of 7-bit LIF sampling is not limited by parameter precision.

It is worth noting that the number of hidden units was fixed here and no training was performed in order to isolate the effect of low precision weights. Thus, parameter precision might still affect the overall training trajectory or be more important for smaller hidden layers.

From the perspective of representability of neural networks, increasing the number of hidden units is another argument for why low precision weights are not an absolutely limiting factor. Indeed, there is a lot of work on quantized neural networks, artificial [CBD14, SWCN] or spiking [PPS+12], and it has been shown that while low precision parameters usually lose a modest amount of accuracy, this loss can be compensated for by increasing the network size while retaining advantages in energy efficiency and memory requirements [HAT+16].

Boltzmann approximation

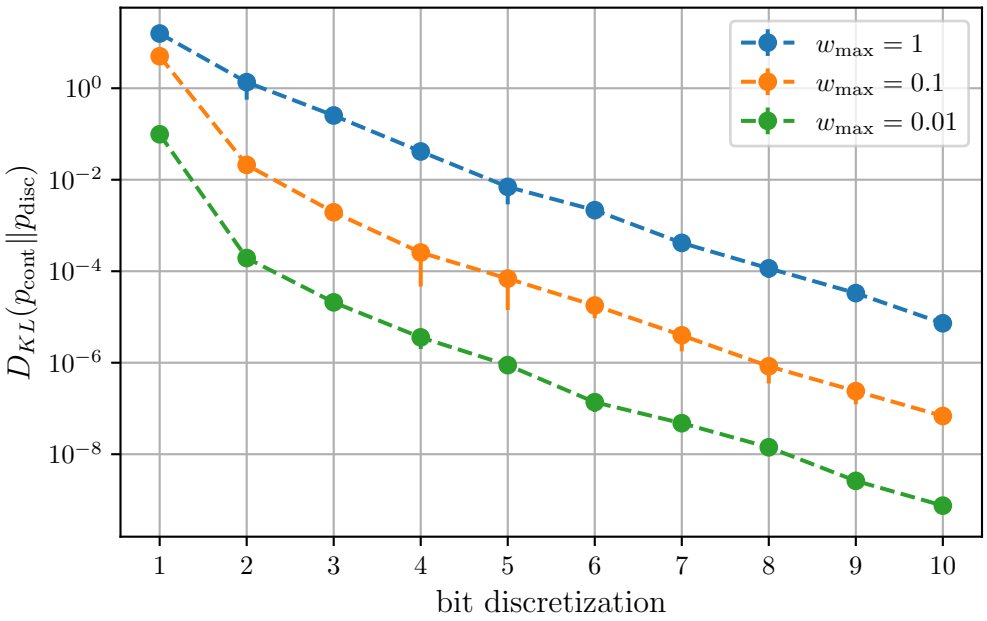
The LIF sampling scheme for Boltzmann distributions that we discussed in section 3.3.1 is an approximation Gibbs sampling for BMs. By setting equal the integrals of the PSPs of both domains and using the regression parameters obtained by fitting the activation functions, one can translate weights and biases from the LIF to the BM domain.

However, the basis of this translation is the assumption that every spike has the same influence (proportional to the synaptic weights) on membrane potentials of receiving neurons and that this impact does not extend for longer than τ_{ref}. But the PSPs are (almost) exponential for LIF neurons in the HCS thus extending their influence beyond the refractory period. This overshoot causes the impact of spikes to depend on the previous membrane history undermining the translation calculation [1].

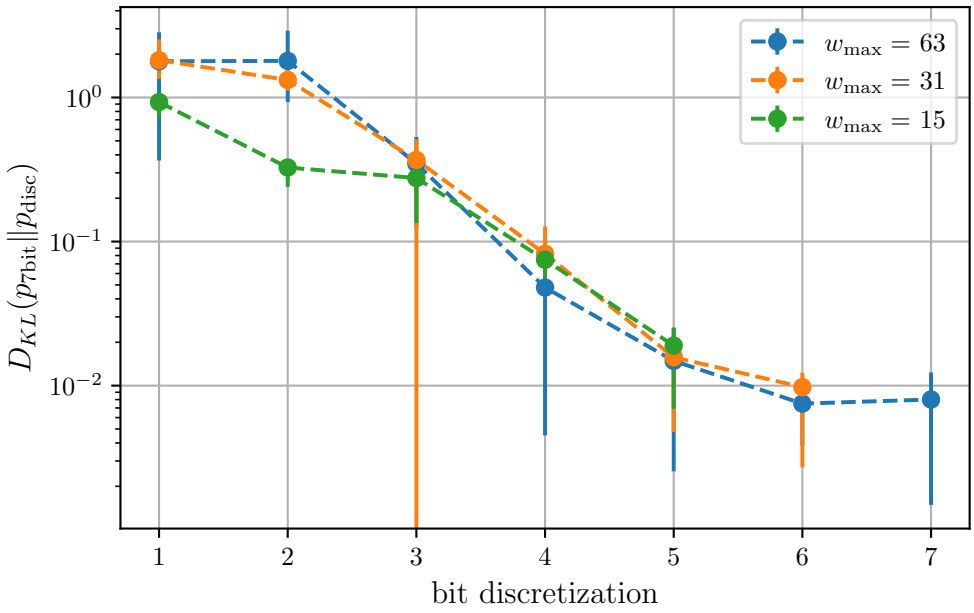
From a practical viewpoint the translation is further complicated by the precision of the chip calibration and its inherent variability.

In fig. 6.2 we show the DKL between LIF sampling on BSS2 and the translated analytical BM distribution for small network sizes. One can not compare the relative entropies between system sizes. However, it is evident that the translation is a crude

[1] [PBB+16] uses short term plasticity to compensate for this. While such circuitry is available on BSS2 it was not calibrated for our usecase yet.



(a) Exact DKL between discretized and continuous parameter RBMs with $N_v = 20$ and $N_h = 40$. Mean over 10 initializations with half width $w_{\max}$.



(b) DKL between very low precision (<7-bit) and 7-bit RBMs on BSS2 with $N_v = 8$ and $N_h = 20$. Mean over 10 initializations with half width $w_{\max}$.

Figure 6.1

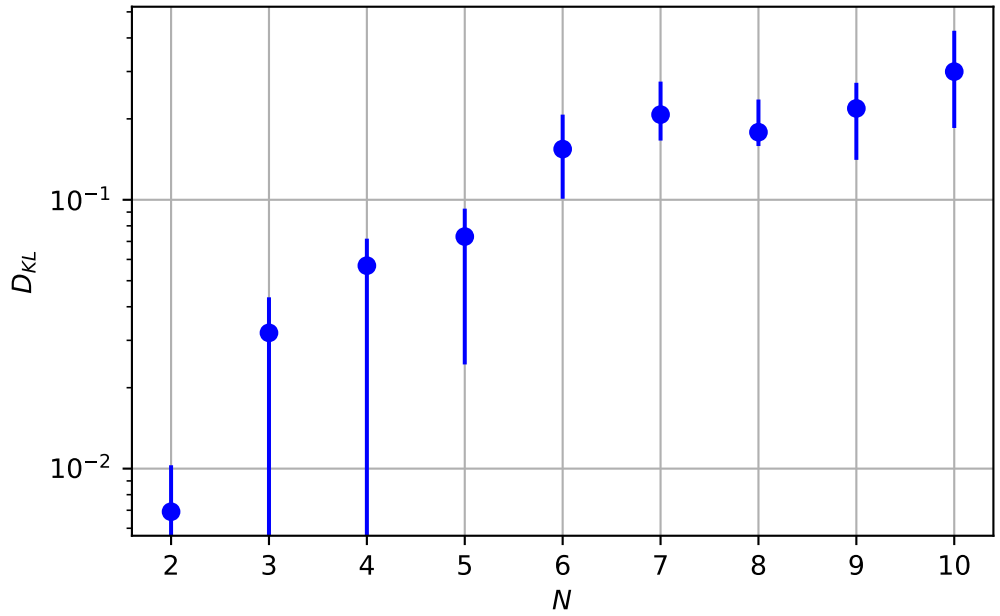


Figure 6.2: Comparing distributions of the LIF sampler and its abstract BM translation for network size N measured by the $D_{KL}(p_{\mathrm{BM}} \| p_{\mathrm{LIF}})$

approximation based on the absolute values of the DKL.

Thus, as expected, it follows that the LIF network on BSS2 only samples from an approximation of a Boltzmann distribution. While extended models have been developed that go beyond the initial LIF theory [Gür18, Bau20], so far a detailed description of the approximation error, yet alone a correction of distribution is missing.

This limits the possibilities of BSS2 as a co-processor for fast sampling of BMs since it is not compatible with numerical methods that work in the BM-domain. For example, many quantum Monte Carlo methods rely on reweighting [CT17, CPGG19, LCCC20] where (unnormalized) probabilities of specific configurations need to be evaluated. In the case of RBMs these correspond to Boltzmann factors, i.e. the energy functional, that would be efficiently computable given the weights and biases of the network. For example, were accurate translations possible algorithm 4 would become scalable because calculation of the local energy would not have to rely on the full visible distribution anymore. Instead, one could explicitly sum over all terms that are coupled to a configuration v by the sparse Hamiltonian.

Another limitation concerns the gradients which we also calculate based on the Boltzmann assumption. Those will as a result be slightly wrong in every iteration building up an additional optimization error, which is currently impossible to quantify.

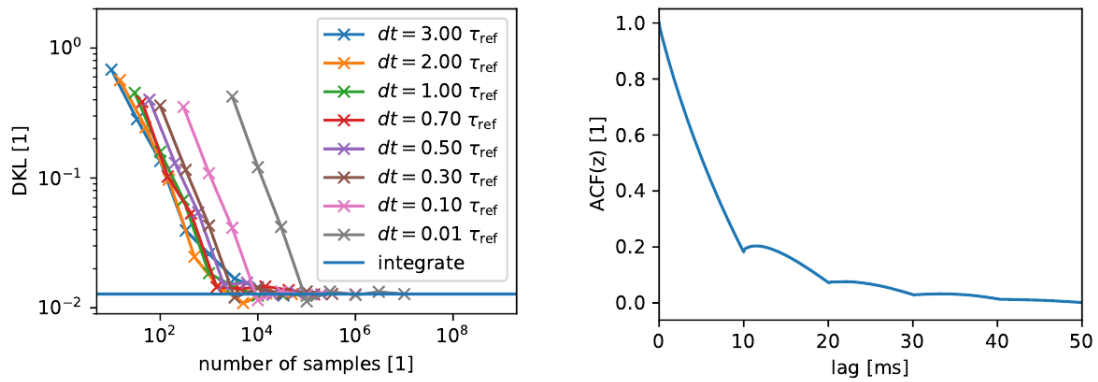


Figure 6.3: Effect of sampling interval dt (left) and autocorrelation function of states z (right) in a simulation of LIF sampling (Taken from [Bau20]).

Autocorrelation of LIF sampling

In order to compare LIF sampling to other MCMC methods it is desirable to provide a notion of when the continuous process has evolved to a new, uncorrelated sample state. In other words, how long is the autocorrelation time of the state variables z?

Since z are directly connected to the membrane potentials via eq. (3.10) it is conceivable that the autocorrelation time is the dominating time scale of changes in u. In the HCS this is the synaptic/refractory time as $\tau_m \ll \tau_{\text{syn}} = \tau_{\text{ref}}$ (roughly $10\,\mu\text{s}$). This estimate is shown to be valid in fig. 6.3 (right) [2].

In practice it is thus advantageous to choose the sampling interval on the order of magnitude of τ_{ref}. But which is the optimal value for dt? Figure 6.3 (left) empirically shows that the sample correlations start to affect sampling efficiency when choosing $dt \lessapprox 0.5\tau_{\text{ref}}$. On the other hand choosing values $dt \geq \tau_{\text{ref}}$ is a waste of samples since a new uncorrelated sample is already available. Theoretically, one can determine the optimal value by making use of the Shannon sampling theorem [Sha49]. It states that a signal with maximum frequency of f can be perfectly reconstructed with a sampling interval of at least $1/2f$. In our case the signal corresponds to the network state $z(t)$ whose maximum frequency is given by the inverse of the refractory period $1/\tau_{\text{ref}}$. Thus, the optimal interval between samples is $dt = \tau_{\text{ref}}/2 \approx 5\,\mu\text{s}$ which is our standard choice (see table B.2).

It should be noted, that due to BSS2's accelerated dynamics this is not a limitation, but rather a strength, when it comes to quickly producing many samples. On the other hand, if you want to use BSS2 not in isolation but as part of a larger system, its particularly fast dynamics could be a downside in terms of synchronizing the communication.

[2] the kinks are an additional, but unrelated problem caused by a bursting network neuron

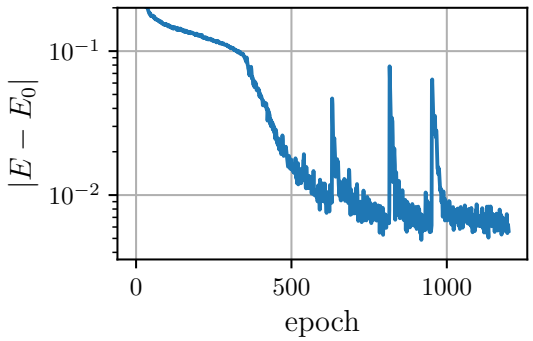


Figure 6.4: Dying neurons while learning the ground state for $N = 7$ via a RBM with $N_h = 30$.

Dying neurons and parameter instabilities

BSS2 as a whole is still under development. As such errors, bugs and noise are unavoidable.

Here we just highlight two problems that were still present in version 2 of BSS2 which is what the majority of experiments in this thesis are based on.

The first is an issue dubbed "dying neurons" which is a behavior where neurons stop spiking at some point during training. This behavior interferes with optimization and, ironically, leads to spikes in the learning curve. If hidden units fail in this way (and still enough remain) the network can recover as shown in fig. 6.4, where the three spikes mark the death of three different neurons (which afterwards stay silent until the chip and the connection to it are restarted). If a visible unit fails this is not recoverable since half of the spin configurations can not be expressed anymore. In this case the training has to be restarted.

The second issue is hypothesized to originate from a type of parameter instability. It manifests itself during individual sampling runs where parameters are supposedly shifting during the sampling duration. Accordingly, the encoded distribution is altered such that the maximum useful sampling time is limited.

In fig. 6.5 the effect on the DKL after learning the Bell state is shown. The data show median and $15^{\text{th}}/85^{\text{th}}$-percentiles of 10 sampling experiments each of duration $T = 3\,\text{s}$.

The red curve tracks the DKL between the model distributions of all trial runs across sampling time. The initial decrease indicates that trials are consistently sampling from the same distribution. In version 1 of BSS2 this curve would level off at around $0.1\,\text{s}$ which is a sign that parameters have shifted differently between trials. Version 2 decreased these variations and as a result the trials at least stay consistent up to $\mathcal{O}(1\,\text{s})$.

The blue, yellow and green curves show the DKL with the target state by aggregating differently sized subsets of the 10 experiments. First of all, we notice that there is no major difference between the curves which confirms that trial-to-trial differences are

negligible. The DKLs drop until around $0.3\,\mathrm{s}$ to around 10^{-2} in agreement with the final DKL observed after training. However, continuing the sampling *increases* the DKL again. Since we have excluded trial-to-trial variation as an explanation this must be caused by a shift of the implemented model distribution and hence the configured parameters.

Increasing this parameter stability of long sampling experiments through improved hardware is important because it would enable to cut down the overhead time of reconfiguration between the repeated sampling process that is currently used and enable the sampler to overcome the current inherent lower bound in terms of DKLs.

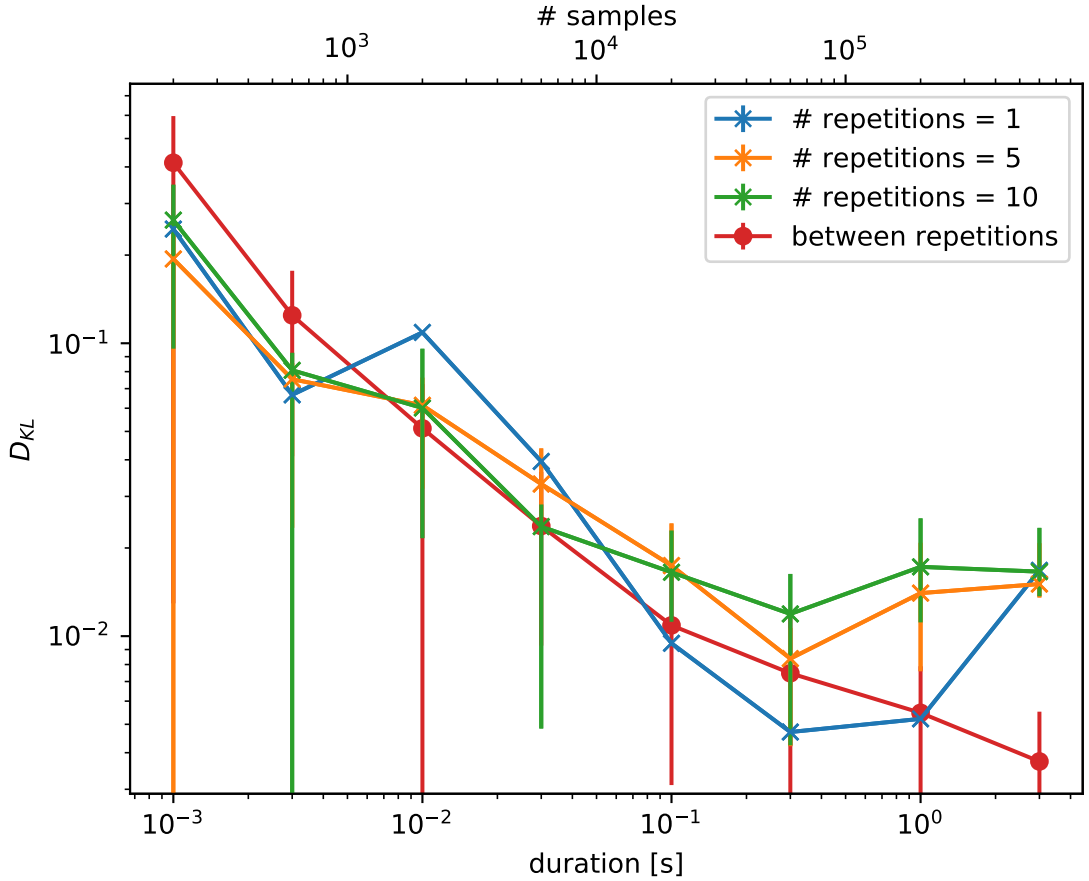


Figure 6.5: Final DKL after learning the Bell state as function of sampling time/number of samples. All points are based on 10 sampling experiments of $3\,\mathrm{s}$ length. Blue, orange and green show statistics (median, 15/85-percentiles) over a subset of 1, 5 and all 10 runs with respect to the target distribution. Red shows the DKL between sampling distributions of the 10 runs.

Time spent "in the loop"

All results in this thesis have been obtained by in-the-loop learning, i.e. while sampling was performed on the BSS2 chip, the evaluation of observables and gradients was performed by the connected host computer.

One advantage of in-the-loop training is that one can process the whole network readout thus enabling the implementation of global learning rules like in algorithm 3 and algorithm 4 [3]. In principle, one can even stop learning at any time and resume at a later time if the configuration can be transferred to the later execution. The corresponding drawback is the communication latency when reading out spikes and configuring the chip.

Figure 6.6 shows the relative time chunks that are spent during each epoch and sampling experiment. To be comparable to the durations provided in [CBB+21] we have chosen an equally sized network performing VGS for the TFIM with $N = 4$ and employing an RBM with $N_h = 24$. An epoch consists of a parameter update where configuration of the chip takes place, N_{reps} sampling experiments and the evaluation of gradients and observables. Individual sampling experiments comprise the raw chip runtime, the spike readout and subsequent conversion to samples.

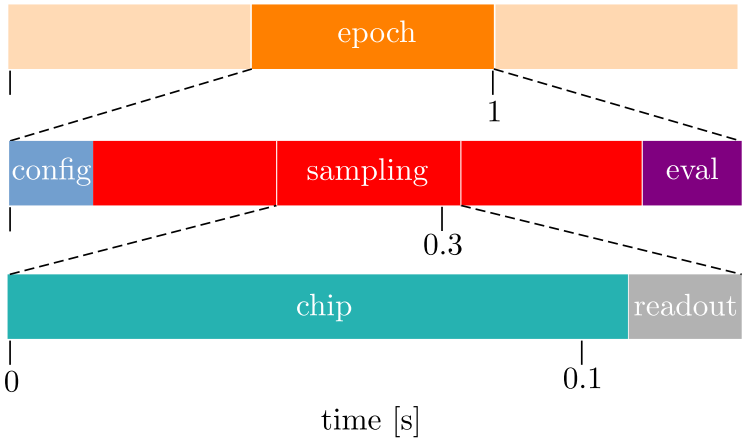


Figure 6.6: Duration of different steps for in-the-loop training (here: VGS for $N = 4$ with $N_h = 24$ to make it comparable to [CBB+21]). A full epoch consists of configuring parameters (blue), multiple sampling experiments (red) and the evaluation of gradients and observables on the host computer (lila). One sampling experiment consists of BSS2 chip runtime (turquoise) and sample readout (grey)

The configuration time is a fixed cost while readout as well as evaluation scale lin-

[3] and also that there is a user-friendly high-level Python interface.

early with the chip runtime. While the chip runtime is independent of the network size, the readout depends linearly on it and evaluation goes with the number of samples. Overall the extra time of in-the-loop learning is not a real limitation on current research since it only takes around 20% of the total training time, which could be improved upon. Compared to the $\approx 80\%$ reported in [CBB+21], (SI, fig. 5d), this is already a large improvement which stems primarily from getting rid of the FPGA buffering and optimized evaluation code.

6.2 Exploiting marginal POVM distributions

In this section preliminary results are presented on an alternative variational approach for learning steady states of OQS. The idea rests on the factorizability in the IC-POVM framework. When dealing with low-order interactions, like is the case for spin Hamiltonians, one can trace over the non-interacting spins and thereby, for each interaction, achieve an exponential decrease in the size of the required variational probability distribution. While this is not possible for wave functions due to their complex nature, the normalization of POVM distributions allows for leveraging this property.

We start out with the Lindblad master equation, eq. (2.6), which can be re-expressed in probabilistic terms by expanding the density matrix into a set of IC-POVM operators $M^{\boldsymbol{a}}$ such that $\rho = \sum_{\boldsymbol{ab}} p_{\boldsymbol{b}} M^{\boldsymbol{a}} T^{-1}_{\boldsymbol{ab}}$. The master equation and respective Liouvillian then read,

$$\frac{\mathrm{d}p_{\boldsymbol{a}}}{\mathrm{d}t} = \sum_{\boldsymbol{b}} L_{\boldsymbol{ab}} p_{\boldsymbol{b}} = \sum_{\boldsymbol{b}} (A_{\boldsymbol{ab}} + B_{\boldsymbol{ab}}) p_{\boldsymbol{b}} \text{ with} \tag{6.1}$$

$$A_{\boldsymbol{ab}} = -i \operatorname{Tr}(H \left[N^{\boldsymbol{b}}, M^{\boldsymbol{a}}\right]) \tag{6.2}$$

$$B_{\boldsymbol{ab}} = \sum_k \frac{\gamma_k}{2} \operatorname{Tr}(2\Gamma_k N^{\boldsymbol{b}} \Gamma_k^\dagger M^{\boldsymbol{a}} - \Gamma_k^\dagger \Gamma_k \{N_{\boldsymbol{b}}, N_{\boldsymbol{a}}\}) \tag{6.3}$$

where $N^{\boldsymbol{b}} = \sum_{\boldsymbol{a}} M^{\boldsymbol{a}} T^{-1}_{\boldsymbol{ab}}$.

L is a real matrix such that $\sum_{\boldsymbol{ab}} L_{\boldsymbol{ab}} p_{\boldsymbol{b}} = 0$, preserving probability mass. For time-independent Hamiltonians and strong enough dissipation rates the time evolution of the system has a single fixed point, the steady state p_f:

$$\frac{\mathrm{d}p}{\mathrm{d}t}|_f = L p_f = 0 \tag{6.4}$$

The steady state can thus be directly found by calculating the unique eigenvector of L with eigenvalue 0. One approach to achieve this is to define an optimization problem via the norm of the master equation that can be solved by making a variational ansatz p_θ and searching for a state that minimizes the update $\|\dot{p_\theta}\|$.

In order for the optimization to work on BSS2 in a scalable fashion we have to be able to calculate the loss function and the gradient from samples only.

This restriction prohibits the use of the one-norm $\|\dot{p}_\theta\|_1 = \sum_{\boldsymbol{a}} |\sum_{\boldsymbol{b}} L_{\boldsymbol{ab}} p_\theta^{\boldsymbol{b}}|$, since the outer sum can not be sampled without reweighting by a dense sampling of the whole distribution. However, we will see that by using the two-norm and assuming a local Hamiltonian, we can circumvent this problem. Hence, we seek to solve

$$p_f = \text{argmin}_\theta \|Lp_\theta\|_2^2 = \text{argmin}_\theta p_\theta L^T L p_\theta = \text{argmin}_\theta \sum_{\boldsymbol{abc}} p_\theta^{\boldsymbol{a}} L_{\boldsymbol{ba}} L_{\boldsymbol{bc}} p_\theta^{\boldsymbol{b}} \tag{6.5}$$

We abbreviate the squared Liouvillian that appears as $\Lambda = L^\dagger L = A^T A + A^T B + B^T A + B^T B$.

As in previous chapters we are assuming a BM (see eq. (3.7)) to model the POVM distribution such that the gradient with respect to a weight connection reads

$$\frac{\mathrm{d}p_\theta}{\mathrm{d}W_{ij}} = \sum_{\boldsymbol{z}=[\boldsymbol{b},\boldsymbol{h}]} \left(\sum_{\boldsymbol{a}} \Lambda_{\boldsymbol{ab}} p_\theta^{\boldsymbol{a}} - \|\dot{p}_\theta^{\boldsymbol{a}}\|_2^2 \right) z_i z_j p_{\boldsymbol{z}} \tag{6.6}$$

This expression would suffice to implement the brute force scheme where the exponentially large POVM distribution $p_\theta^{\boldsymbol{a}}$ is densely sampled at every update step. However, given that the Hamiltonian consists of low-order interactions one should be able to leverage the linearity of the POVM formalism and restrict expectation values to the relevant marginal distributions.

In general, a r-local Hamiltonian is one where interactions between up to r spins are present:

$$H = \sum_{i_1,\ldots,i_r} J_{i_1,\ldots,i_r} \Sigma_{i_1,\ldots,i_r} + \ldots + \sum_{ij} J_{ij} \Sigma_{ij} + \sum_i J_i \Sigma_i \tag{6.7}$$

where J are are the interaction coefficients and Σ are product r-body interactions.

We consider at most spin pair interactions, i.e. $r = 2$ with $\Sigma_{ij} = \sigma^i \otimes \sigma^j$ where σ^i is an arbitrary Pauli matrix acting on spin i. Inserting this Hamiltonian with up to 2^{nd} order into Λ the first term reads:

$$A_{\boldsymbol{ba}} A_{\boldsymbol{bc}} = \sum_{ijkl} J_{ij} J_{kl} \,\mathrm{Tr}\left(\sigma^i \sigma^j \left[N^{\boldsymbol{a}}, M^{\boldsymbol{b}}\right]\right) \mathrm{Tr}\left(\sigma^k \sigma^l \left[N^{\boldsymbol{c}}, M^{\boldsymbol{b}}\right]\right) + \tag{6.8}$$

$$\sum_{ijk} 2 J_{ij} J_k \,\mathrm{Tr}\left(\sigma^i \sigma^j \left[N^{\boldsymbol{a}}, M^{\boldsymbol{b}}\right]\right) \mathrm{Tr}\left(\sigma^k \left[N^{\boldsymbol{c}}, M^{\boldsymbol{b}}\right]\right) + \tag{6.9}$$

$$\sum_{ij} J_i J_k \,\mathrm{Tr}\left(\sigma^i \left[N^{\boldsymbol{a}}, M^{\boldsymbol{b}}\right]\right) \mathrm{Tr}\left(\sigma^j \left[N^{\boldsymbol{c}}, M^{\boldsymbol{b}}\right]\right) \tag{6.10}$$

While we can say that the first line contains the 4-body terms, the second the 3-body terms and the third the 2-body terms, we need to factorize each term in order to make use of marginal distributions.

All operators inside the traces can be readily factorized $M^{\boldsymbol{a}} = \otimes_n M^{a_n}$, $N^{\boldsymbol{b}} = \otimes_n N^{b_n}$ and $\sigma^i = \otimes_k \sigma^{2-\delta_{k,i}}$. Thus the trace-terms can be written in the form

$$\begin{aligned}
\mathrm{Tr}(\sigma_i \sigma_j \left[N^{\boldsymbol{b}}, M^{\boldsymbol{a}}\right]) &= \mathrm{Tr}(\otimes_n \sigma_n^i \sigma_n^j N^{b_n} M^{a_n} - \otimes_n \sigma_n^i \sigma_n^j M^{a_n} N^{b_n}) && (6.11)\\
&= \prod_n \mathrm{Tr}(\sigma_n^i \sigma_n^j N^{b_n} M^{a_n}) - \prod_n \mathrm{Tr}(\sigma_n^i \sigma_n^j M^{a_n} N^{b_n}) && (6.12)\\
&= (\prod_{n \neq i \neq j} \mathrm{Tr}\, N^{b_n} M^{a_n}) \cdot D^{ij}_{\boldsymbol{ab}} && (6.13)\\
\text{with } D^{ij}_{\boldsymbol{ab}} &= (\prod_{n \in \{i,j\}} \mathrm{Tr}\, \sigma^n N^{b_n} M^{a_n} - \prod_{n \in \{i,j\}} \mathrm{Tr}\, \sigma^n M^{b_n} N^{a_n}) && (6.14)
\end{aligned}$$

The first factor can be rewritten as product of Kronecker delta's using eq. (2.17). Naming the second factor above $D^{\boldsymbol{ab}}_{ij}$, we can express the first summand of $L^\dagger L$ compactly:

$$A_{\boldsymbol{ba}} A_{\boldsymbol{bc}} = \sum_{ijkl} J_{ij} J_{kl} (\prod_{n \neq i \neq j} \delta_{a_n b_n} \prod_{n \neq k \neq l} \delta_{b_n c_n}) D^{ij}_{\boldsymbol{ba}} D^{kl}_{\boldsymbol{bc}} + 2^{\text{nd}} \,\&\, 3^{\text{rd}} \text{ terms} \qquad (6.15)$$

To obtain $A^T A$ we still have to sum over $\boldsymbol{b}$. In the case of $i \neq j \neq k \neq l$ this is especially easy since the constraints put in place by the δ's completely determine $\boldsymbol{b}$ from $\boldsymbol{a}$ and $\boldsymbol{c}$:

1. $b_n = a_n = c_n \forall n \notin \{i, j, k, l\}$

2. $b_n = a_n$ for $n = k \wedge n = l$

3. $b_n = c_n$ for $n = i \wedge n = k$

Hence, the sum over $\boldsymbol{b}$ can be omitted by directly computing $\boldsymbol{b}$ from $\boldsymbol{a}$ and $\boldsymbol{c}$. The sums over $\boldsymbol{a}$ and $\boldsymbol{c}$ will also be restricted to the combinations obeying constraint 1.

One can also see this by summing out over $\boldsymbol{b}$ while contracting the δ's. Consider an

example term where $N = 3$, $i = 1$ and $j = 2$:

$$\sum_{\boldsymbol{b}} \left(\prod_{n\neq i, m\neq j} \delta_{a_n,b_n}\delta_{a_m,b_m} \right) D^1_{\boldsymbol{ba}} D^2_{\boldsymbol{bc}} \tag{6.16}$$

$$= \sum_{b_1,b_2,b_3} \delta_{a_2,b_2}\delta_{a_3,b_3}\delta_{b_1,c_1}\delta_{b_3,c_3} \tag{6.17}$$

$$\cdot (\operatorname{Tr}\sigma^1 N^{b_1} M^{a_1} - \operatorname{Tr}\sigma^1 M^{a_1} N^{b_1})(\operatorname{Tr}\sigma^2 N^{b_2} M^{a_2} - \operatorname{Tr}\sigma^j M^{a_2} N^{b_2}) \tag{6.18}$$

$$= \delta_{a_2,b_2}\delta_{b_1,c_1}\delta_{a_3,c_3} D^1_{\boldsymbol{ba}} D^2_{\boldsymbol{bc}} = \delta_{a_3,c_3} \sum_{b_1} \delta_{b_1,c_1} D^1_{\boldsymbol{ba}} \sum_{b_2} \delta_{a_2,b_2} D^2_{\boldsymbol{bc}} \tag{6.19}$$

$$= \delta_{a_3,c_3} (\operatorname{Tr}\sigma^1 N^{c_1} M^{a_1} - \operatorname{Tr}\sigma^1 M^{a_1} N^{c_1})^* (\operatorname{Tr}\sigma^2 N^{c_2} M^{a_2} - \operatorname{Tr}\sigma^2 M^{a_2} N^{b_2}) \tag{6.20}$$

When one or two pairs between the sets $\{i, j\}$ and $\{k, l\}$ coincide, the sum over doubled variables b_x does not vanish since they are not coupled to $\boldsymbol{a}$ and $\boldsymbol{c}$. However, since there are at most two coincidences, this sum only amounts to at most four terms.

Given a sample $\boldsymbol{a_s}$ we would like to know which configurations it couples to in each term.

When neglecting coincidences, in the first term of $A^T A$ the above constraints reduce the number of coupled configurations $\tilde{\boldsymbol{c}} = (c_i, c_j, c_k, c_l)$ to at most $4^4 = 256$ (128 at one, 64 at two coincidences). Similarly, the second term and third terms of $A^T A$ reduce the sum to variables $\tilde{\boldsymbol{c}} = (c_i, c_j, c_k)$ and $\tilde{\boldsymbol{c}} = (c_i, c_j)$ respectively:

$$\sum_{\boldsymbol{cb}} A_{\boldsymbol{ba}} A_{\boldsymbol{bc}} p_{\boldsymbol{c}} = \sum_{ijkl} J_{ij} J_{kl} \sum_{c_i,c_j,c_k,c_l} p(c_i, c_j, c_k, c_l) \sum_{b_x} D^{ij}_{\boldsymbol{ba}} D^{kl}_{\boldsymbol{bc}} \tag{6.21}$$

$$+ \sum_{ijk} 2 J_{ij} J_k \sum_{c_i,c_j,c_k} p(c_i, c_j, c_k) \sum_{b_x} D^{ij}_{\boldsymbol{ba}} D^{k}_{\boldsymbol{bc}} \tag{6.22}$$

$$+ \sum_{ijk} J_i J_j \sum_{c_i,c_j} p(c_i, c_j) \sum_{b_x} D^{i}_{\boldsymbol{ba}} D^{j}_{\boldsymbol{bc}} \tag{6.23}$$

Here the sum over b_x is meant as sum over the double occuring indices. Note that these also lower the order of the needed marginal distribution. Given a sample $\boldsymbol{a_s}$ we can now calculate $\left[A^T A p_{\text{marginal}}\right]_{\boldsymbol{a_s}}$.

The other three terms in Λ can in principle be treated in a similar fashion. After replacing the remaining outer sum of the minimization objective with a sampling average one obtains:

$$\|\dot{p_{\theta}}_2^2\| = \sum_{\boldsymbol{ac}} p_{\boldsymbol{a}} \Lambda_{\boldsymbol{ac}} p_{\boldsymbol{c}} = \frac{1}{N} \sum_{\boldsymbol{a_s}} \left[\Lambda p_{\text{marginal}}\right]_{\boldsymbol{a_s}} \tag{6.24}$$

Instead of densely sampling the exponentially large POVM distributions it suffices to sample all marginals up to 4$^{\text{th}}$ order. The total number of probabilities that have to be

represented is thus:

$$4^4\binom{N}{4} + 4^3\binom{N}{3} + 4^2\binom{N}{2} \in \mathcal{O}(N^4) \tag{6.25}$$

In addition, a generative model could be employed to try to capture these with an even better scaling in terms of the number of model parameters.

Thus, using marginal POVM distributions could enable an efficient search for steady states. We have not tested this idea numerically yet due to time limitations. However, it seems plausible that a working algorithm can be built from the above blueprint. Whether the method will be efficient in practice remains to be seen.

7 Conclusion & Outlook

A summary of the last three chapters might be: exploring the viability of expressing quantum spin states with SNN-based generative models that approximate (R)BMs, emulated on a specific neuromorphic chip, namely BSS2.

Following the previous results by Czischek et al. [CBB+21], in chapter 4 we found that for both the POVM and the z-basis representation one can accurately learn small system sizes of $N \lessapprox 6$ and $N \lessapprox 10$, respectively. Thereby, the highly entangled GHZ states proved to be more difficult to accurately represent than equally sized product states, ground states and steady states of the TFIM. Beyond these small systems the **technical limitations** of BSS2 discussed in section 6.1, especially analog parameter drift and dying neurons, prevented a high-fidelity encoding.

Knowing that progress in mixed-signal neuromorphic hardware will eventually make these limitations disappear, in chapter 5 we developed a method for searching ground states of stoquastic Hamiltonians. Testing on TFIM spin chains of sizes up to $N = 10$ we confirmed good agreement with theory with the exception of when spontaneous symmetry breaking occurs at low field strengths h.

Both of the approaches, for QST and VGS require access to the full visible state distribution which grows exponentially with the system size. Independent of the hardware specific issues, there is a need for lifting this **algorithmic limitation**. One attempt to do so for the specific problem of global steady state search, presented in section 6.2, makes use of marginal IC-POVM distributions.

Regarding both types of limitation, there are several ways for addressing them in future research. Today there already exist a number of purely digital neuromorphic systems like SpiNNaker-2 [MHF19] and Loihi [DSL+18] for the efficient large-scale emulation of LIF networks. These systems are deterministic by design and thus allow studying the scalability of the presented methods independent of the instabilities experienced on mixed-signal or analog neuromorphic hardware. Furthermore, digital neuromorphic systems could also help to bridge the gap between the Boltzmann domain and the LIF domain in terms of parameter translation since the additional inaccuracy due to variation in analog hardware parameters vanishes. Thereby, the direct calculation of Boltzmann factors from the weight and bias parameters could become feasible and thus alleviate the need for densely sampling the visible distribution.

Another promising idea for better algorithms is the use of local learning rules. There are two reasons for this. First, it has been shown that global gradient-based update rules, like error backpropagation in feedforward ANNs, can be approximated by local

learning rules. Relevant techniques are predictive coding [WB17] and direct feedback alignment [CPGR19, LZZ+20]. An example for training RBMs with a local learning rule is contrastive divergence [Hin12] of which an event-driven version has been proposed [NDP+14]. The second reason is that local on-chip learning is widely supported by most modern neuromorphic platforms and would thus also eliminate the overhead of in-the-loop learning approaches.

Finally, algorithmic improvements could be enabled by novel encodings of NQS with SNNs. For example, recently, a new learning rule for SNNs based on first-spike encoding has been demonstrated on BSS2 [GKB+21]. In addition, a straightforward idea for encoding not only the amplitudes, but also phases of the wave function would be to use additional output units or even a second network like in [TMC+18]. Lastly, phasor networks represent an avenue for encoding complex numbers with SNNs. It was shown that these networks, which consist of *resonate*-and-fire neurons with complex dynamical variables, can be implemented by integrate-and-fire SNNs and can robustly leverage spike-timing codes [FS19]. If successful, these approaches to representing complex values in SNNs could enable the extension of the presented VGS method to non-stoquastic systems and even pave the way for entirely new applications in quantum physics.

A Lists

List of Figures

2.2 Schematic of Hilbert space and the subspace of relevant physical states. 12
2.1 A pure (blue) and mixed (red) spin state on the Bloch sphere 12
2.3 Visualization of the TFIM in one dimension. 13
2.4 Phase diagram of the TFIM 13
2.5 Ground state energy per site of the TFIM. 14
2.6 Tetrahedral POVM on Bloch sphere. 19
2.7 Bell state as SIC-POVM distribution. 20
2.8 Qplex 20

3.1 Sketch of a biological neuron. Image adapted from [Haa12] 22
3.2 Comparing the artificial and spiking neuron 23
3.3 Equivalent circuit of the LIF neuron 24
3.4 Boltzmann machine and restricted Boltzmann machine 27
3.5 Mapping from spikes to binary configurations 29
3.6 Membrane trace of LIF sampling unit. 31
3.7 BrainScaleS-2 picture and schematic. 32
3.8 LIF sampling network matrix and activation functions. 34

4.1 Comparing differential evolution and gradient descent. 42
4.2 Learning the Bell state with Gibbs sampling. 44
4.3 Shannon entropy and difference to uniform distribution of TFIM ground state. 45
4.4 Learning the product state $|1\rangle^{\otimes N}$. 46
4.5 Deep Boltzmann machine 47
4.6 Learning the Bell state with different architectures. 48
4.7 Learning the GHZ state with different architectures. 49
4.8 Learning the GHZ state in 2020. 50
4.9 Learning the 4-spin GHZ state. 51

4.10 Learning SIC-POVM representation of the TFIM ground state. 52
4.11 Learning steady states of the TFIM. 53
4.12 Learning z-basis representation of the TFIM ground states. 54

5.1 Comparing natural and vanilla gradient descent for ground state search in software. 59
5.2 Comparing natural gradient and ADAM on BrainScaleS-2. 60
5.3 short 61
5.4 Relative energy error and infidelity of learned TFIM ground states for $N \in \{3, \ldots, 10\}$. 62
5.5 Learning curves for energy error and infidelity of learned TFIM ground states for $N \in \{3, \ldots, 10\}$. 62
5.6 Magnetization and two-body correlator during ground state learning for $N = 10$. 63
5.7 Energy error and infidelity for TFIM ground states at the critical point. . 64
5.8 Diagram of the phase transition between ordered and disordered regime. 65
5.9 Decay of correlator C_{zz} as a function of spin distance d. 66
5.10 (a) Relative energy error and (b) infidelity as a function of h/J 67
5.11 Symmetry breaking for low h. 68

6.1 Weight discretization experiment in software and hardware. 71
6.2 Translating and comparing LIF sampler and abstract BM. 72
6.3 Effect of sampling interval and autocorrelation of states z. 73
6.4 Dying neurons in learning curve. 74
6.5 Long sampling experiment showing parameter drift. 75
6.6 Wall-clock time of training process. 76

List of Tables

4.1 Total number of parameters as function of the number of visible N_v and hidden N_h units per architecture. 48

5.1 Parameter settings for learning different h/J. 64

B.1 Calibration parameters for HICANN-X 86
B.2 LIF sampling settings for HICANN-X 86
B.3 Optimization algorithm parameters for BSS2 86

B Parameter settings

Parameter	Value
E_l	80 lsb
u_R	80 lsb
u_T	120 lsb
τ_{mem}	$0.5\,\mu\text{s}$
τ_{syn}	$10\,\mu\text{s}$
τ_{ref}	$10\,\mu\text{s}$
$I_{\text{gm}}^{\text{syn}}$	350 lsb
C_m	10 lsb
b_{syn}	600 lsb

Table B.1: Calibration parameters for HICANN-X

Parameter	Value
T	$0.1\,\text{s}$
dt	$5\,\mu\text{s}$
w_{noise}	15-25 lsb
ν_{noise}	$80\,\text{kHz}$
m_{noise}	4
N_{reps}	3-10

Table B.2: LIF sampling settings for HICANN-X

Optimizer	Parameters
gradient descent	$\eta_1 = 1$, $\gamma_{\text{lr}} = 0.999$
ADAM	$\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon_{\text{ADAM}} = 10^{-8}$
NGD	$\epsilon_{\text{reg}} = 10^{-4}$, $\gamma_{\text{reg}} = 0.99$
DE	$CP = 0.7$, $DW = 0.5$, $N_p = 10 \times N_{\text{params}}$
initialization	$w_{ij} \sim \mathcal{U}(-W_{\text{init}}, W_{\text{init}})$ with $W_{\text{init}} = 50\,\text{lsb}$, $b_k = b_k^0$

Table B.3: Optimization algorithm parameters for BSS2

C Acronyms

ADAM adaptive moment estimation. 36, 37, 41, 42, 57, 59, 60, 86

ANN artificial neural networks. 6, 7, 18, 23, 24, 82, 89

ASIC application specific integrated circuits. 7

BM Boltzmann machines. 8, 25–31, 35, 39, 41, 43, 45, 47, 55, 56, 60, 68, 70, 72, 78

BSS2 BrainScaleS-2. 7, 8, 22, 29, 32–34, 39–43, 47, 49, 55, 56, 58–61, 67–74, 76, 78, 82, 83, 85, 86, 89

DE Differential Evolution. 38, 41, 42, 57, 86

DKL Kullback-Leibler divergence. 16, 18, 31, 35, 36, 40, 41, 43, 47–52, 54, 55, 57, 70–72, 74, 75

FIM Fisher information matrix. 37, 60

GHZ Greenberger-Horne-Zeilinger. 39, 43, 49–52, 55, 82, 84

HCS high-conductance state. 30, 31, 70, 73

LIF leaky integrate-and-fire. 7, 8, 22, 24, 29–31, 33, 34, 39–41, 49, 60, 68–70, 72, 73, 82, 84–86

MCMC Markov chain Monte Carlo. 27, 28, 73

ML machine learning. 6, 18, 25, 26

MPS Matrix Product States. 18

NGD Natural gradient descent. 37, 58–60, 86

NQS Neural Quantum States. 6, 7, 83

OQS open quantum systems. 11, 69, 77

OU Ornstein-Uhlenbeck process. 30

POVM positive operator-valued measurements. 6, 8, 15, 16, 18–20, 41, 43, 45, 68, 69, 77, 78, 80–82, 84

PSP postsynaptic potentials. 7, 23, 25, 31, 70

QM Quantum mechanics. 9, 10, 17

QST quantum state tomography. 6, 9, 18, 35, 39, 69, 82

RBM restricted Boltzmann machine. 6, 18, 28, 29, 34–37, 41, 44–47, 49–52, 54, 56, 58–61, 69, 71, 72, 74, 76, 83

RNN recurrent neural network. 6, 18

SNN spiking neural networks. 7, 8, 24, 26, 29, 39, 41, 47, 56, 68, 82, 83

TFIM Transverse Field Ising Model. 8, 12–14, 39, 43, 52–54, 56, 58, 60, 61, 63, 67, 76, 82, 84, 85

VGS variational groundstate search. 6, 9, 17, 35, 54, 57, 67, 69, 76, 82, 83

VMC Variational Monte Carlo. 17, 18

D Acknowledgments

I would like to express my gratitude to all sentient beings who supported me during this project. First, thank you to Martin Gärttner and Andreas Baumbach for great supervision, discussions and ideas. Second, I am thankful to Stefanie Czischek for starting the "quantum-neuromorphic project", answering all my questions and helping me get started. I thank Thomas Gasenzer for being the second advisor. Thanks to the many-body quantum dynamics group for helpful discussions and fun games. This work would not be possible without the Electronic Vision(s) group; thanks for creating BrainScaleS and providing access and support for it. Thanks to everyone who participated in the ANN Journal Club and the SynQS Seminar for nice talks and discussions. And finally, thanks to my family for infinite support.

The work carried out in this Master Thesis used systems, which received funding from the European Union's Horizon 2020 Framework Programme for Research and Innovation under the Specific Grant Agreements Nos. 785907 and 945539 (Human Brain Project, HBP)

E Bibliography

[AFSZ17] Marcus Appleby, Christopher A. Fuchs, Blake C. Stacey, and Huangjun Zhu. Introducing the Qplex: A novel arena for quantum theory. *The European Physical Journal D*, 71(7):197, July 2017.

[AHS85] David H. Ackley, Geoffrey E. Hinton, and Terrence J. Sejnowski. A Learning Algorithm for Boltzmann Machines*. *Cognitive Science*, 9(1):147–169, January 1985.

[AM12] Adolfo Avella and Ferdinando Mancini, editors. *Strongly Correlated Systems: Theoretical Methods*. Springer Series in Solid-State Sciences. Springer-Verlag, Berlin Heidelberg, 2012.

[AMNK20] Shahnawaz Ahmed, Carlos Sánchez Muñoz, Franco Nori, and Anton Frisk Kockum. Quantum State Tomography with Conditional Generative Adversarial Networks. *arXiv:2008.03240 [quant-ph]*, December 2020.

[Bau20] Andreas Baumbach. From microscopic dynamics to ensemble behavior in spiking neural networks. page 209, 2020.

[BBNM11] Lars Buesing, Johannes Bill, Bernhard Nessler, and Wolfgang Maass. Neural Dynamics as Sampling: A Model for Stochastic Computation in Recurrent Networks of Spiking Neurons. *PLoS Computational Biology*, 7(11):e1002211, November 2011.

[BC17] Jacob C. Bridgeman and Christopher T. Chubb. Hand-waving and interpretive dance: An introductory course on tensor networks. *Journal of Physics A: Mathematical and Theoretical*, 50(22):223001, May 2017.

[BCP+21] Sebastian Billaudelle, Benjamin Cramer, Mihai A. Petrovici, Korbinian Schreiber, David Kappel, Johannes Schemmel, and Karlheinz Meier. Structural plasticity on an accelerated analog neuromorphic hardware system. *Neural Networks*, 133:11–20, January 2021.

[BDOT07] Sergey Bravyi, David P. DiVincenzo, Roberto I. Oliveira, and Barbara M. Terhal. The Complexity of Stoquastic Local Hamiltonian Problems. *arXiv:quant-ph/0606140*, October 2007.

[Bel64] J. S. Bell. On the Einstein Podolsky Rosen paradox. *Physics Physique Fizika*, 1(3):195–200, November 1964.

[BFK+15] Matthias Bartelmann, Björn Feuerbacher, Timm Krüger, Dieter Lüst, Anton Rebhan, and Andreas Wipf. *Theoretische Physik*. Springer Spektrum, 2015.

[BG05] Romain Brette and Wulfram Gerstner. Adaptive Exponential Integrate-and-Fire Model as an Effective Description of Neuronal Activity. *Journal of Neurophysiology*, 94(5):3637–3642, November 2005.

[Bis06] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Information Science and Statistics. Springer, New York, 2006.

[Bra15] Sergey Bravyi. Monte Carlo simulation of stoquastic Hamiltonians. *arXiv:1402.2295 [quant-ph]*, January 2015.

[BSD20] Marin Bukov, Markus Schmitt, and Maxime Dupont. Learning the ground state of a non-stoquastic quantum Hamiltonian in a rugged neural network landscape. *arXiv:2011.11214 [cond-mat, physics:physics, physics:quant-ph]*, November 2020.

[Car20] Juan Carrasquilla. Machine Learning for Quantum Matter. *Advances in Physics: X*, 5(1):1797528, January 2020.

[CBB+21] Stefanie Czischek, Andreas Baumbach, Sebastian Billaudelle, Benjamin Cramer, Lukas Kades, Jan M. Pawlowski, Markus K. Oberthaler, Johannes Schemmel, Mihai A. Petrovici, Thomas Gasenzer, and Martin Gärttner. Spiking neuromorphic chip learns entangled quantum states. *arXiv:2008.01039 [cond-mat, physics:quant-ph]*, February 2021.

[CBD14] Matthieu Courbariaux, Y. Bengio, and Jean-Pierre David. Low precision arithmetic for deep learning. *3rd International Conference on Learning Representations (ICLR2015)*, December 2014.

[CCX+18] Jing Chen, Song Cheng, Haidong Xie, Lei Wang, and Tao Xiang. Equivalence of restricted Boltzmann machines and tensor network states. *Physical Review B*, 97(8):085104, February 2018.

[CGG18] Stefanie Czischek, Martin Gärttner, and Thomas Gasenzer. Quenches near Ising quantum criticality as a challenge for artificial neural networks. *Physical Review B*, 98(2):024311, July 2018.

[CPGG19] Stefanie Czischek, Jan M. Pawlowski, Thomas Gasenzer, and Martin Gärttner. Sampling scheme for neuromorphic simulation of entangled quantum systems. *Physical Review B*, 100(19):195120, November 2019.

[CPGR19] Brian Crafton, Abhinav Parihar, Evan Gebhardt, and Arijit Raychowdhury. Direct Feedback Alignment With Sparse Connections for Local Learning. *Frontiers in Neuroscience*, 13, 2019.

[CT17] Giuseppe Carleo and Matthias Troyer. Solving the Quantum Many-Body Problem with Artificial Neural Networks. *Science*, 355(6325):602–606, February 2017.

[CTMA19] Juan Carrasquilla, Giacomo Torlai, Roger G. Melko, and Leandro Aolita. Reconstructing quantum states with generative models. *Nature Machine Intelligence*, 1(3):155–161, March 2019.

[Czi20] Stefanie Czischek. *Neural-Network Simulation of Strongly Correlated Quantum Systems*. Springer Theses. Springer International Publishing, Cham, 2020.

[DSL+18] Mike Davies, Narayan Srinivasa, Tsung-Han Lin, Gautham Chinya, Yongqiang Cao, Sri Harsha Choday, Georgios Dimou, Prasad Joshi, Nabil Imam, Shweta Jain, Yuyun Liao, Chit-Kwan Lin, Andrew Lines, Ruokun Liu, Deepak Mathaikutty, Steven McCoy, Arnab Paul, Jonathan Tse, Guruguhanathan Venkataramanan, Yi-Hsin Weng, Andreas Wild, Yoonseok Yang, and Hong Wang. Loihi: A Neuromorphic Manycore Processor with On-Chip Learning. *IEEE Micro*, 38(1):82–99, January 2018.

[FI14] Asja Fischer and Christian Igel. Training restricted Boltzmann machines: An introduction. *Pattern Recognition*, 47(1):25–39, January 2014.

[FMS14] Christopher A. Fuchs, N. David Mermin, and Ruediger Schack. An Introduction to QBism with an Application to the Locality of Quantum Mechanics. *American Journal of Physics*, 82(8):749–754, August 2014.

[FS19] E. Paxon Frady and Friedrich T. Sommer. Robust computation with rhythmic spike patterns. *Proceedings of the National Academy of Sciences*, 116(36):18050–18059, September 2019.

[GBC+20] Andreas Grübl, Sebastian Billaudelle, Benjamin Cramer, Vitali Karasenko, and Johannes Schemmel. Verification and Design Methods for the BrainScaleS Neuromorphic Hardware System. *Journal of Signal Processing Systems*, 92(11):1277–1292, November 2020.

[GG] Stuart Geman and Donald Geman. Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images. page 21.

[GKB+21] Julian Göltz, Laura Kriener, Andreas Baumbach, Sebastian Billaudelle, Oliver Breitwieser, Benjamin Cramer, Dominik Dold, Akos Ferenc Kungl, Walter Senn, Johannes Schemmel, Karlheinz Meier, and Mihai Alexandru Petrovici. Fast and energy-efficient neuromorphic deep learning with first-spike times. *arXiv:1912.11443 [cs, q-bio, stat]*, May 2021.

[Gür18] Nico Gürtler. A Markovian Model of LIF Networks, 2018.

[GWKM02] Wulfram Gerstner, Wulfram, Kistler, and Werner M. *Spiking Neuron Models: Single Neurons, Populations, Plasticity*. August 2002.

[Haa12] Jonathan Haas, 2012. File:Neurons uni bi multi pseudouni.svg.

[HAT+16] Soheil Hashemi, Nicholas Anthony, Hokchhay Tann, R. Iris Bahar, and Sherief Reda. Understanding the Impact of Precision Quantization on the Accuracy and Energy of Neural Networks. *arXiv:1612.03940 [cs]*, December 2016.

[Hin07] Geoffrey E. Hinton. Boltzmann machine. *Scholarpedia*, 2(5):1668, May 2007.

[Hin12] Geoffrey E. Hinton. A Practical Guide to Training Restricted Boltzmann Machines. In Grégoire Montavon, Geneviève B. Orr, and Klaus-Robert Müller, editors, *Neural Networks: Tricks of the Trade*, volume 7700, pages 599–619. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.

[HIW+21] Mohamed Hibat-Allah, Estelle M. Inack, Roeland Wiersema, Roger G. Melko, and Juan Carrasquilla. Variational Neural Annealing. *arXiv:2101.10154 [cond-mat, physics:quant-ph]*, January 2021.

[Hor91] Kurt Hornik. Approximation capabilities of multilayer feedforward networks. *Neural Networks*, 4(2):251–257, January 1991.

[HP19] John L. Hennessy and David A. Patterson. A new golden age for computer architecture. *Communications of the ACM*, 62(2):48–60, January 2019.

[HTF09] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. Undirected Graphical Models. In Trevor Hastie, Robert Tibshirani, and Jerome Friedman, editors, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, pages 625–648. Springer New York, New York, NY, 2009.

[Isi25] Ernst Ising. Beitrag zur Theorie des Ferromagnetismus. *Zeitschrift für Physik*, 31(1):253–258, February 1925.

[JNN12] J. R. Johansson, P. D. Nation, and Franco Nori. QuTiP: An open-source Python framework for the dynamics of open quantum systems. *Computer Physics Communications*, 183(8):1760–1772, August 2012.

[KB17] Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. *arXiv:1412.6980 [cs]*, January 2017.

[KCH+17] Markus Karl, Halil Cakir, Jad C. Halimeh, Markus K. Oberthaler, Michael Kastner, and Thomas Gasenzer. Universal equilibrium scaling functions at short times after a quench. *Physical Review E*, 96(2):022110, August 2017.

[KHF18] Hassan N. Khan, David A. Hounshell, and Erica R. H. Fuchs. Science and research policy at the end of Moore's law. *Nature Electronics*, 1(1):14–21, January 2018.

[LCCC20] Di Luo, Zhuo Chen, Juan Carrasquilla, and Bryan K. Clark. Autoregressive Neural Network for Simulating Open Quantum Systems via a Probabilistic Formulation. *arXiv:2009.05580 [cond-mat, physics:physics, physics:quant-ph]*, September 2020.

[LMB+18] Luziwei Leng, Roman Martel, Oliver Breitwieser, Ilja Bytschok, Walter Senn, Johannes Schemmel, Karlheinz Meier, and Mihai A. Petrovici. Spiking neurons with short-term synaptic plasticity form superior generative networks. *Scientific Reports*, 8(1):10651, July 2018.

[LZZ+20] Jeongjun Lee, Renqian Zhang, Wenrui Zhang, Yu Liu, and Peng Li. Spike-Train Level Direct Feedback Alignment: Sidestepping Backpropagation for On-Chip Training of Spiking Neural Nets. *Frontiers in Neuroscience*, 14, 2020.

[Maa97] Wolfgang Maass. Networks of spiking neurons: The third generation of neural network models. *Neural Networks*, 10(9):1659–1671, December 1997.

[Mea89] C. Mead. Analog VLSI and neural systems. /paper/Analog-VLSI-and-neural-systems-Mead/3f1ff3c401126449300b4f521107484379a1be3b, 1989.

[MHF19] Christian Mayr, Sebastian Hoeppner, and Steve Furber. SpiNNaker 2: A 10 Million Core Processor System for Brain Simulation and Machine Learning. *arXiv:1911.02385 [cs]*, November 2019.

[MMQG20] Danijela Marković, Alice Mizrahi, Damien Querlioz, and Julie Grollier. Physics for neuromorphic computing. *Nature Reviews Physics*, 2(9):499–510, September 2020.

[Moo98] Gordon E Moore. Cramming More Components onto Integrated Circuits. *PROCEEDINGS OF THE IEEE*, 86(1):4, 1998.

[MS95] Z. F. Mainen and T. J. Sejnowski. Reliability of spike timing in neocortical neurons. *Science*, 268(5216):1503–1506, June 1995.

[NC10] Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press, Cambridge ; New York, 10th anniversary ed edition, 2010.

[NDP+14] Emre Neftci, Srinjoy Das, Bruno Pedroni, Kenneth Kreutz-Delgado, and Gert Cauwenberghs. Event-driven contrastive divergence for spiking neuromorphic systems. *Frontiers in Neuroscience*, 7, 2014.

[NFJ+20] Marcel Neugebauer, Laurin Fischer, Alexander Jäger, Stefanie Czischek, Selim Jochim, Matthias Weidemüller, and Martin Gärttner. Neural network quantum state tomography in a two-qubit experiment. *Physical Review A*, 102(4):042604, October 2020.

[Orú19] Román Orús. Tensor networks for complex quantum systems. *Nature Reviews Physics*, 1(9):538–550, September 2019.

[PB14] Razvan Pascanu and Yoshua Bengio. Revisiting Natural Gradient for Deep Networks. *arXiv:1301.3584 [cs]*, February 2014.

[PBB+] Mihai A Petrovici, Johannes Bill, Ilja Bytschok, Johannes Schemmel, and Karlheinz Meier. Stochastic inference with deterministic spiking neurons. page 7.

[PBB+16] Mihai A. Petrovici, Johannes Bill, Ilja Bytschok, Johannes Schemmel, and Karlheinz Meier. Stochastic inference with spiking neurons in the high-conductance state. *Physical Review E*, 94(4):042312, October 2016.

[Pet16] Mihai A. Petrovici. *Form vs. Function: Theory and Models for Neuronal Substrates*. Dissertation, Ruprecht-Karls-Universität Heidelberg, 2015, Heidelberg, überarbeitete fassung edition, 2016.

[PF10] John B. Parkinson and Damian J. J. Farnell. *An Introduction to Quantum Spin Systems*. Lecture Notes in Physics. Springer-Verlag, Berlin Heidelberg, 2010.

[Pfe70] Pierre Pfeuty. The one-dimensional Ising model with a transverse field. *Annals of Physics*, 57(1):79–90, 1970.

[PKB+20] Adriano Macarone Palmieri, Egor Kovlakov, Federico Bianchi, Dmitry Yudin, Stanislav Straupe, Jacob D. Biamonte, and Sergei Kulik. Experimental neural network enhanced quantum tomography. *npj Quantum Information*, 6(1):1–5, February 2020.

[PPG+13] Eustace Painkras, Luis A. Plana, Jim Garside, Steve Temple, Francesco Galluppi, Cameron Patterson, David R. Lester, Andrew D. Brown, and Steve B. Furber. SpiNNaker: A 1-W 18-Core System-on-Chip for Massively-Parallel Neural Network Simulation. *IEEE Journal of Solid-State Circuits*, 48(8):1943–1953, August 2013.

[PPS+12] Thomas Pfeil, Tobias C. Potjans, Sven Schrader, Wiebke Potjans, Johannes Schemmel, Markus Diesmann, and Karlheinz Meier. Is a 4-Bit Synaptic Weight Resolution Enough? – Constraints on Enabling Spike-Timing Dependent Plasticity in Neuromorphic Hardware. *Frontiers in Neuroscience*, 6, 2012.

[Pre18] John Preskill. Quantum Computing in the NISQ era and beyond. *Quantum*, 2:79, August 2018.

[PSL05] Kenneth V. Price, Rainer M. Storn, and Jouni A. Lampinen. *Differential Evolution: A Practical Approach to Global Optimization*. Natural Computing Series. Springer, Berlin ; New York, 2005.

[PVWC07] D. Perez-Garcia, F. Verstraete, M. M. Wolf, and J. I. Cirac. Matrix Product State Representations. *arXiv:quant-ph/0608197*, May 2007.

[RSG21] Moritz Reh, Markus Schmitt, and Martin Gärttner. Time-dependent variational principle for open quantum systems with artificial neural networks. *arXiv:2104.00013 [cond-mat, physics:physics, physics:quant-ph]*, March 2021.

[SAM10] Anders W. Sandvik, Adolfo Avella, and Ferdinando Mancini. Computational Studies of Quantum Spin Systems. In *LECTURES ON THE PHYSICS OF STRONGLY CORRELATED SYSTEMS XIV: Fourteenth Training Course in the Physics of Strongly Correlated Systems*, pages 135–338, Vietri sul Mare, (Italy), 2010.

[SBDW20] Johannes Schemmel, Sebastian Billaudelle, Phillip Dauer, and Johannes Weis. Accelerated Analog Neuromorphic Computing. *arXiv:2003.11996 [cond-mat, q-bio]*, March 2020.

[SBG+10] Johannes Schemmel, Daniel Brüderle, Andreas Grübl, Matthias Hock, Karlheinz Meier, and Sebastian Millner. A wafer-scale neuromorphic hardware system for large-scale neural modeling. In *2010 IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 1947–1950, May 2010.

[SGA+19] Dan Sehayek, Anna Golubeva, Michael S. Albergo, Bohdan Kulchytskyy, Giacomo Torlai, and Roger G. Melko. The learnability scaling of quantum states: Restricted Boltzmann machines. *Physical Review B*, 100(19):195125, November 2019.

[Sha49] C.E. Shannon. Communication in the Presence of Noise. *Proceedings of the IRE*, 37(1):10–21, January 1949.

[Smo86] Paul Smolensky. Information processing in dynamical systems: Foundations of harmony theory. *Parallel Distributed Process*, 1, January 1986.

[SPP+] Catherine D Schuman, Thomas E Potok, Robert M Patton, Douglas Birdwell, Mark E Dean, and Garrett S Rose. A Survey of Neuromorphic Computing and Neural Networks in Hardware. page 89.

[SWCN] Xiao Sun, Naigang Wang, Chia-yu Chen, and Jia-min Ni. Ultra-Low Precision 4-bit Training of Deep Neural Networks. page 12.

[TGSY95] A. N. Tikhonov, A. Goncharsky, V. V. Stepanov, and Anatoly G. Yagola. *Numerical Methods for the Solution of Ill-Posed Problems*. Mathematics and Its Applications. Springer Netherlands, 1995.

[TM18] Giacomo Torlai and Roger G. Melko. Latent Space Purification via Neural Density Operators. *Physical Review Letters*, 120(24):240503, June 2018.

[TMC+18] Giacomo Torlai, Guglielmo Mazzola, Juan Carrasquilla, Matthias Troyer, Roger Melko, and Giuseppe Carleo. Neural-network quantum state tomography. *Nature Physics*, 14(5):447–450, May 2018.

[UO30] G. E. Uhlenbeck and L. S. Ornstein. On the Theory of the Brownian Motion. *Physical Review*, 36(5):823–841, September 1930.

[VGLH21] Agnes Valenti, Eliska Greplova, Netanel H. Lindner, and Sebastian D. Huber. Correlation-Enhanced Neural Networks as Interpretable Variational Quantum States. *arXiv:2103.05017 [cond-mat, physics:quant-ph]*, March 2021.

[WB17] James C. R. Whittington and Rafal Bogacz. An Approximation of the Error Backpropagation Algorithm in a Predictive Coding Network with Local Hebbian Synaptic Plasticity. *Neural Computation*, 29(5):1229–1262, May 2017.

Erklärung:

Ich versichere, dass ich diese Arbeit selbstständig verfasst habe und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe.

Heidelberg, den (Datum) 31.8.21 R. Klassert .....