



Master's thesis

Master's Programme in Computer Science

Analysis and implementation of quantum Boltzmann machines

Leevi Lehtonen

June 6, 2021

FACULTY OF SCIENCE
UNIVERSITY OF HELSINKI

Supervisor(s)

Prof. Jukka K. Nurminen, Dr. Hassan Naseri

Examiner(s)

Prof. Valteri Niemi

Contact information

P. O. Box 68 (Pietari Kalmin katu 5)
00014 University of Helsinki, Finland

Email address: info@cs.helsinki.fi

URL: <http://www.cs.helsinki.fi/>

Tiedekunta — Fakultet — Faculty		Koulutusohjelma — Utbildningsprogram — Study programme	
Faculty of Science		Master's Programme in Computer Science	
Tekijä — Författare — Author			
Leevi Lehtonen			
Työn nimi — Arbetets titel — Title			
Analysis and implementation of quantum Boltzmann machines			
Ohjaajat — Handledare — Supervisors			
Prof. Jukka K. Nurminen, Dr. Hassan Naseri			
Työn laji — Arbetets art — Level		Aika — Datum — Month and year	Sivumäärä — Sidoantal — Number of pages
Master's thesis		June 6, 2021	61 pages
Tiivistelmä — Referat — Abstract			
<p>Quantum computing has an enormous potential in machine learning, where problems can quickly scale to be intractable for classical computation. A Boltzmann machine is a well-known energy-based graphical model suitable for various machine learning tasks. Plenty of work has already been conducted for realizing Boltzmann machines in quantum computing, all of which have somewhat different characteristics.</p> <p>In this thesis, we conduct a survey of the state-of-the-art in quantum Boltzmann machines and their training approaches. Primarily, we examine variational quantum Boltzmann machine, a specific variant of quantum Boltzmann machine suitable for the near-term quantum hardware. Moreover, as variational quantum Boltzmann machine heavily relies on variational quantum imaginary time evolution, we effectively analyze variational quantum imaginary time evolution to a great extent.</p> <p>Compared to the previous work, we evaluate the execution of variational quantum imaginary time evolution with a more comprehensive collection of hyperparameters. Furthermore, we train variational quantum Boltzmann machines using a toy problem of bars and stripes, representing more multimodal probability distribution than the Bell states and the Greenberger-Horne-Zeilinger states considered in the earlier studies.</p> <p>ACM Computing Classification System (CCS) Computing methodologies → Machine learning → Machine learning approaches → Learning in probabilistic graphical models Computer systems organization → Architectures → Other architectures → Quantum computing</p>			
Avainsanat — Nyckelord — Keywords			
quantum computing, machine learning, Boltzmann machine			
Säilytyspaikka — Förvaringsställe — Where deposited			
Helsinki University Library			
Muita tietoja — övriga uppgifter — Additional information			
Algorithms study track			

Acknowledgements

I thank Prof. Jukka K. Nurminen and Dr. Hassan Naseri for supervising this thesis and guiding me through this project. I also thank my employer, Accenture, for providing flexible working conditions and financial support. Additionally, I am grateful for the valuable discussions with Dean Smith and Tim Leonhardt.

I want to also express my gratitude to my parents and my partner for their continuous support and love throughout the journey. Moreover, I am grateful to my friends for their support as well as for providing the much-needed distractions.

Helsinki, Finland, June 2021

Leevi Lehtonen

Contents

1	Introduction	1
2	Background and literature review	4
2.1	Machine learning	4
2.2	Markov random fields	5
2.3	Boltzmann machine	6
2.3.1	Restricted Boltzmann machine	8
2.4	Quantum computation	9
2.4.1	Qubits and pure states	9
2.4.2	Evolution of a quantum system	11
2.4.3	Density operators and mixed states	12
2.5	Quantum computing models	13
2.5.1	Quantum circuit	13
2.5.2	Adiabatic quantum computing	15
2.5.3	Quantum annealing	16
2.6	Quantum Boltzmann machine	16
2.6.1	Learning classical probability distribution	19
2.6.2	Learning quantum probability distribution	22
2.6.3	Learning using variational principle	24
3	Methods and algorithms	27
3.1	Variational quantum Boltzmann machine	27
3.2	Experimentation setup	27
3.3	Implementation	29
4	Results	30
4.1	Preparing thermal states using VarQITE	30
4.1.1	Preparing thermal states of Ising model	32
4.1.2	Preparing thermal states of Heisenberg model	34

4.2 Efficient approximation using stochasticity 38

4.3 Training quantum Boltzmann machine 39

5 Discussion 47

5.1 Analysis of the results 47

5.2 Limitations of the study 49

5.3 Future work 50

6 Conclusion 52

Bibliography 55

1 Introduction

Machine learning (ML) algorithms are essential ingredients in powering today's society [42]. Examples of ML applications include intelligent search engines, recommendation systems in e-commerce, self-driving cars, chatbots and industrial robots of various kinds. Furthermore, ML is being more and more applied for solving complex problems in engineering, science, medicine, management and finance, where it can provide good approximations for generally intractable problems. The common factor among all ML algorithms is that they do not require an exact and explicit specification of the problem at hand. Instead, they learn through some experience that is often data and evolve themselves on solving a particular problem [26].

Meanwhile, a totally new computing paradigm, quantum computing, is becoming more and more a reality [52]. Quantum computing is based on utilizing quantum mechanical phenomena like superposition and entanglement for computations. While there have been ideas for quantum computing already for decades, only recently, actual small-scale quantum computing hardware has become available for experimentation. Such hardware is referred to as noisy intermediate-scale quantum (NISQ) technology meaning noisy quantum computing hardware consisting of 50 to few hundreds of qubits [53].

Quantum computing is often promised to deliver exponential speedups by utilizing quantum mechanical phenomena [11, 50, 55]. While the arguments appear increasingly promising and optimistic, the promise of exponential speedups itself can be highly controversial. That is, some of the speedup opportunities so far discovered require some strict assumptions to hold [11, 63]. For example, that the necessary input could be provided in an already prepared superposition or the existence of a quantum RAM (QRAM) that can store and deliver copies of quantum data [24]. Since some of these assumed requirements may not be efficiently satisfied in the near future, the promised exponential speedups may not be immediately achievable [53].

Quantum computing has plenty of promising use cases, especially in solving complex problems and calculations in chemistry, physics and medicine. Furthermore, a new study field, quantum machine learning (QML), is also rising from the intersection of machine learning and quantum computing [11, 13, 58]. QML is an interdisciplinary study field focusing on enhancing ML methods using quantum computing in the hope of introducing

quantum speedups or inventing more expressive models. Like ML itself, QML is also challenging to define as there are many different ways to connect quantum computing to ML or vice versa. One traditional approach in QML is to consider two directions that could be quantum: the computing device and the data often originating from some problem [5]. Most of the currently studied models in QML are generalizations of their respective classical variants, where a quantum algorithm has replaced some element of the model [11]. For example, in a recent proposal on quantum support vector machine, quantum phase estimation and quantum matrix inversion algorithms are used to reduce the computational complexity to logarithmic [54]. Furthermore, some of these generic-level algorithms, like the Harrow, Hassidim and Lloyd (HHL) algorithm [29] for solving linear systems of equations, could be used as a subroutine in a wide range of ML models.

Another interesting area is to consider models whose classical computation is intractable and requires approximations. Quantum computing could potentially provide a better way for solving or approximating such models. Particularly with graphical models, the exact calculations tend to be intractable, and hence approximations are required [40]. For instance, Markov random fields (MRFs) are graphical models suitable for expressing undirected and cyclic dependencies between variables [10, 23, 38]. They are rather general-level models that can be applied to a wide range of ML tasks. However, training MRFs like Boltzmann machines (BMs) [2, 31, 33] with maximum likelihood is computationally expensive because it requires evaluating statistics whose calculation tends to scale exponentially.

The exponential nature of the BMs and the connections to statistical mechanics in physics suggest that the BMs could be superior models in quantum computing settings given the exponentially large Hilbert space where the computation is performed. Furthermore, as some of the studies have shown, the model scales naturally to quantum mechanics and can be even trained against quantum probability distributions [7, 37, 69]. Realizing a BM in a quantum setting (quantum Boltzmann machine) is a prime example of QML.

This thesis aims to investigate quantum Boltzmann machines (QBMs) and their training approaches. Primarily, the focus is on variational quantum Boltzmann machine (VarQBM) [73], a specific variant of QBM, which is examined in this thesis through a set of experiments. Furthermore, as VarQBM depends heavily on the variational quantum imaginary time evolution (VarQITE) [46, 72] algorithm, its efficiency and precision are of particular interest.

The key contributions of this thesis include the following.

- The survey of the state-of-the-art in QBMs, in Chapter 2, highlights the notable QBM variants and their training approaches from the literature.
- The implementation and experiments with VarQITE and VarQBM, in Chapter 4, show how the precision of the VarQITE algorithm can be adjusted and how it reflects towards the performance of VarQBM. Compared to previous studies on VarQBM [22, 73], we examine a broader collection of hyperparameters and their impact on performance. Furthermore, instead of the Bell states [52] and the Greenberger-Horne-Zeilinger (GHZ) states [27] experimented in the original VarQBM study [73], we use a toy problem of bars and stripes (BAS), representing more multimodal probability distribution.

The structure of the thesis will go as described in the following. First, we will briefly introduce some fundamental concepts from both ML and quantum computing in Chapter 2. Having introduced the preliminary concepts and background, we will continue reviewing the different formulations of QBMs and their training approaches. After the literature review, we will reimplement VarQITE and VarQBM for experimenting with QBMs. Particularly, we highlight the methods and algorithms we will be using for the experiments in Chapter 3. Following the implementation, in Chapter 4, we collect the results of the experiments aiming to answer how the precision of the VarQITE algorithm can be adjusted and how it reflects the performance of VarQBM. Furthermore, we also note how does VarQBM perform against the BAS dataset. We first experiment only with VarQITE due to its fundamental role in VarQBM. After which, we experiment with the generative training of VarQBM for the toy problem of BAS. In Chapter 5, we discuss and analyze the results and highlight notable limitations of the work. Finally, in Chapter 6, we conclude the thesis.

2 Background and literature review

This chapter introduces and discusses the background concepts and previous work related to quantum machine learning (QML) and particularly to quantum Boltzmann machines (QBMs). Specifically, we introduce models and algorithms in machine learning and quantum computation, covering the background of QBMs. Furthermore, we will highlight and discuss some of the notable previous work that has been conducted on QBMs and particularly on their training.

2.1 Machine learning

Machine learning (ML) is a study field focusing on systems improving themselves through some experience that is often data [35]. The exact definition has become somewhat vague over the years as there is an enormous amount of different models and algorithms that belong to ML. As an additional subfield in ML, deep learning (DL) is often used to characterize ML involving complex nested models usually applied to high-dimensional problems [42]. Additionally, ML is commonly coupled to artificial intelligence (AI), as it is powering some of today's most sophisticated AI solutions in our everyday life. This section will continue by introducing some essential ML and DL concepts using mainly the book [26] by Goodfellow et al. and the book [51] by Murphy. For a more thorough introduction on ML and DL, we refer the readers directly to particular books [26, 51].

There are different ways to categorize ML. Traditionally, ML is divided into supervised and unsupervised learning based on the training data used to train the model. In supervised learning, the data is a set of N labelled samples, $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_i^N$, in which $\mathbf{x}_i \in \mathbb{R}^D$ is a single sample consisting of D features and y_i is the respective label. Examples of supervised ML tasks are regression and classification.

In contrast to supervised learning, in unsupervised learning, the dataset is a set of N unlabelled samples, $\mathcal{D} = \{\mathbf{x}_i\}_i^N$, in which $\mathbf{x}_i \in \mathbb{R}^D$ is a single sample consisting of D features. Unsupervised learning is not as well-defined as supervised learning because it is used in many places for different purposes. Usually, unsupervised methods try to make sense of the data by discovering some latent structures, hierarchies or connections. That is, often, unsupervised learning requires the learning of the whole data-generating proba-

bility distribution. Examples of unsupervised ML tasks are clustering and dimensionality reduction.

In addition to this categorization, ML models can also be justified based on the probability distribution the model is learning. This way, models are often classified as either generative or discriminative. The distinction is that generative models learn the full joint distribution of data $p(x, y)$, while the discriminative models learn the conditional probability distribution $p(y|x)$. It is generally possible to use both kinds of models for tasks like classification, but only generative models are suitable for generating new samples (sampling).

The actual learning procedure varies a lot depending on the model and the problem. However, what is common is that some performance measure or objective function, often called loss or cost function, is designed, and the learning is about optimizing this measure. For instance, for model-based supervised learning, this is mathematically often defined as

$$\arg \min_{\theta} \sum_{\mathbf{x}_i, y_i \in \mathcal{D}} \mathcal{L}(f(\mathbf{x}_i, \theta), y_i), \quad (2.1)$$

where \mathcal{L} is an objective function and f is a θ parameterized function representing the model. Hence, the learning is about finding the parameters θ , which minimize the objective function \mathcal{L} . For finding the optimal parameters, there exists plenty of different ways. Often, the methods are based on a gradient that can determine the direction for the best parameter update.

Having laid out some of the concepts from general ML, we will continue in the following sections focusing on few specific models used in ML. Notably, we will focus on Markov Random fields in Section 2.2 and specifically on Boltzmann machines in Section 2.3 as they are our primary interest in this work.

2.2 Markov random fields

A Markov random field (MRF) [10, 23, 38] is an undirected graphical model described by an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ where the nodes \mathcal{V} (vertices) represent the random variables $\{X_v\}_{v \in \mathcal{V}}$ having a Markov property and the edges \mathcal{E} represent the dependencies among the variables. An example of a six-variable MRF is illustrated in Figure 2.1. The Markov property in a MRF is explained by the neighborhoods in the graph \mathcal{G} . Particularly, an open neighborhood $N(a)$ of node $a \in \mathcal{V}$ is defined as a set of nodes having an edge to

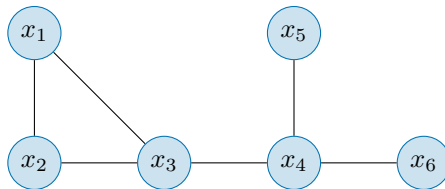


Figure 2.1: An example of a Markov random field including six variables. The nodes of the graph represent the variables having a Markov property, and the edges represent the dependencies between the variables.

node a

$$N(a) = \{b \in \mathcal{V} \mid (a, b) \in \mathcal{E}\}. \quad (2.2)$$

In a MRF all the nodes are independent of all the other nodes given the neighborhoods. Mathematically, the Markov property of a variable $X_a, a \in \mathcal{V}$ is defined by

$$X_a \perp\!\!\!\perp X_{\mathcal{V} \setminus (N(a) \cup a)} \mid X_{N(a)}. \quad (2.3)$$

This is also the Markov blanket of X_a , i.e. the minimal set of variables so that X_a is independent of the other variables.

Furthermore, the graph \mathcal{G} describing the MRF can be factorized into cliques over the variables X_1, \dots, X_n . Using the cliques, the joint probability distribution is defined by

$$p(x_1, \dots, x_n) = \frac{1}{Z} \prod_{c \in \mathcal{C}} \phi_c(x_c), \quad (2.4)$$

where \mathcal{C} is the set of cliques in the graph \mathcal{G} and $\phi_c(x_c)$ is a factor function of a clique c over the variables x_c associated to the nodes of c . Additionally, Z is the normalizing constant, the partition function, which is defined as a sum over all variable assignments

$$Z = \sum_{x_1, \dots, x_n} \prod_{c \in \mathcal{C}} \phi_c(x_c). \quad (2.5)$$

As it is visible from the definition of the normalization constant, it requires calculations over likely an exponential number of assignments. Hence, the exact calculations tend to be intractable with MRFs, at least in general cases, and approximations are required [40].

2.3 Boltzmann machine

A Boltzmann machine (BM) [2, 31, 33] is an energy-based model (EBM) and a specific type of an MRF. Specifically, it is a network of n symmetrically connected stochastic

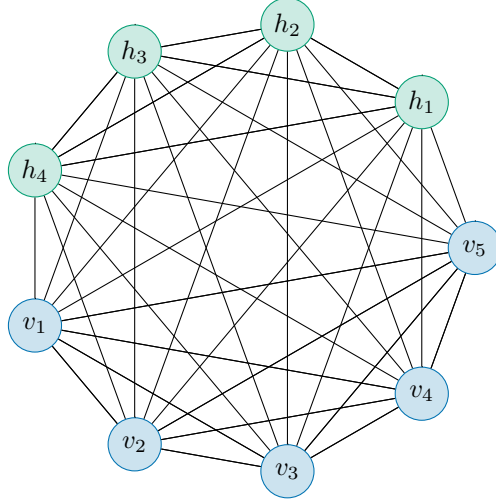


Figure 2.2: An example of a Boltzmann machine with both visible and hidden units.

binary units $\mathbf{z} \in \{0, 1\}^n$. Learning such a BM where all units are observed is a convex problem without global optima [31]. Therefore traditionally BMs are defined so that the units are divided into n_v visible units $\mathbf{v} \in \{0, 1\}^{n_v}$ and n_h hidden units $\mathbf{h} \in \{0, 1\}^{n_h}$. For convenience, \mathbf{z} denotes the set of both visible and hidden units $\mathbf{z} = (\mathbf{v}, \mathbf{h})$. In this case the visible units \mathbf{v} are specified by the observed data and hidden units \mathbf{h} define the latent space. An example of a BM consisting of both visible and hidden units is illustrated in Figure 2.2.

The probability distribution of BM inherits from the definition of total energy, Hamiltonian, in a network specified by the states of its binary units. Furthermore, BM's distribution will eventually converge to a Boltzmann distribution, given that the BM reaches its thermal equilibrium. The total energy of a BM in a state \mathbf{z} is specified by

$$E(\mathbf{z}) = - \sum_i b_i z_i - \sum_{i,j} w_{ij} z_i z_j, \quad (2.6)$$

where $z_i \in \{0, 1\}$ is the binary state of unit i , b_i is a bias associated to the unit i and w_{ij} is the interaction weight between units i and j . Furthermore, the probability of state \mathbf{z} in the equilibrium is specified completely by the state's energy $E(\mathbf{z})$ relative to all possible energies of binary states $\mathbf{z} \in \{0, 1\}^n$

$$P(\mathbf{z}) = \frac{e^{E(\mathbf{z})}}{Z} = \frac{e^{E(\mathbf{z})}}{\sum_{\mathbf{z}'} e^{E(\mathbf{z}')}}. \quad (2.7)$$

It is worth to note the close resemblance to physics, particularly to statistical mechanics, as Equation 2.6 is similar to the Ising model used for modelling ferromagnetism [8].

In the more traditional case where the units are divided into both visible and hidden units as defined earlier. The marginal probability of a visible state \mathbf{v} is calculated by marginalizing over the hidden units (and vice versa for the hidden state)

$$P(\mathbf{v}) = \frac{\sum_{\mathbf{h}} e^{E(\mathbf{v}, \mathbf{h})}}{\sum_{\mathbf{z}} e^{E(\mathbf{z})}}. \quad (2.8)$$

The training of a BM is based on adjusting the model’s parameters so that the probability distribution of the model would be as close as possible to the unknown true probability distribution, often determined by some data. In order to compare the distributions, an objective function is defined for measuring the difference (or similarity) between the distributions; then, the learning can be accomplished by minimizing (or maximizing) this function, for instance, with gradient-based methods. A standard measure for comparing such distributions is the Kullback–Leibler (KL) divergence [41]

$$D_{\text{KL}} = \sum_{\mathbf{v} \in \mathcal{D}} P(\mathbf{v})_{\text{data}} \log \left[\frac{P(\mathbf{v})_{\text{data}}}{P(\mathbf{v})_{\text{model}}} \right]. \quad (2.9)$$

From the learning perspective, this is the same as minimizing the negative log-likelihood (cross-entropy)

$$\mathcal{L} = - \sum_{\mathbf{v} \in \mathcal{D}} P(\mathbf{v})_{\text{data}} \log [P(\mathbf{v})_{\text{model}}]. \quad (2.10)$$

Finally, the parameter update rules for gradient-based optimization can be derived for both the weights and the biases

$$\Delta w_{ij} = \epsilon (\langle z_i z_j \rangle_{\text{data}} - \langle z_i z_j \rangle_{\text{model}}) \quad (2.11)$$

$$\Delta b_i = \epsilon (\langle z_i \rangle_{\text{data}} - \langle z_i \rangle_{\text{model}}), \quad (2.12)$$

where $\langle \cdot \rangle$ denotes the expectation value on the distribution specified in the subscript, and ϵ is a learning rate (or step size) [2]. For the derivations of the learning rules, we refer readers to the original study [2]. By dividing units into visible and hidden so that only visible units are observed and specified by data, the learning rules remain the same [31]. However, similar to MRFs in general, the training of BMs is intractable, as evaluating the expectations with the exact maximum likelihood method scales exponentially [56].

2.3.1 Restricted Boltzmann machine

A restricted Boltzmann machine (RBM) [60] is a special-kind of BM with two layers of binary-valued units, visible $\mathbf{v} \in \{0, 1\}^{n_v}$ and hidden $\mathbf{h} \in \{0, 1\}^{n_h}$, but no intra-layer

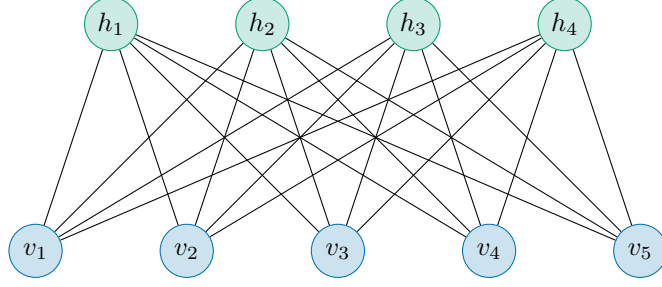


Figure 2.3: An example of a restricted Boltzmann machine

connections, i.e. no visible-to-visible nor hidden-to-hidden connections. An example of a RBM is shown in Figure 2.3.

In the case of RBM the total energy (Equation 2.6 in BM) can be defined with respect to the visible and hidden units by

$$E(\mathbf{v}, \mathbf{h}) = - \sum_i a_i v_i - \sum_j b_j h_j - \sum_{i,j} w_{ij} v_i h_j. \quad (2.13)$$

Meanwhile the marginal probabilities are defined similarly as in general BMs (Equation 2.8).

By introducing these restrictions, the hidden units become conditionally independent given the states of the visible units, making it possible to speed up the evaluation of data expectation [31]. Furthermore, having such a restriction allows the usage of more efficient training algorithms compared to general Boltzmann machines. Notably, a procedure, approximating gradient descent, called the contrastive divergence [32] is commonly used to train RBMs efficiently.

2.4 Quantum computation

This section introduces concepts from quantum mechanics and quantum computation at the level required for following the derivations on QML and QBMs. The introduction is based on the book [52] by Nielsen and Chuang and the book [71] by Wittek.

2.4.1 Qubits and pure states

The fundamental computational elements in quantum computing are qubits representing the state of a quantum system. A state is essentially an arbitrary unit vector in a Hilbert space, \mathcal{H} , that is often a finite-dimensional inner product complex vector space $\mathbb{C}^n, n \in \mathbb{N}$.

Regularly used notation for representing the states is the Dirac notation, in which a vector is notated by a ket, $|\psi\rangle$. For example, the single-qubit computational basis states are defined by the following kets

$$|0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \text{ and } |1\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix}. \quad (2.14)$$

Dual vector of a ket is represented by a bra $\langle\psi|$ which mathematically is a Hermitian conjugate of the respective ket so that $\langle\psi| = |\psi\rangle^\dagger$.

Unlike classical bits (0 and 1), quantum states can also be in the linear combination of computational basis states, traditionally referred to as superpositions

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle = \begin{bmatrix} \alpha \\ \beta \end{bmatrix}, \quad (2.15)$$

where the coefficients $\alpha, \beta \in \mathbb{C}$ are probability amplitudes so that the second axiom of probability is preserved $|\alpha|^2 + |\beta|^2 = 1$. The notation scales trivially to a multi-qubit case

$$|\psi\rangle = \sum_i \alpha_i |x_i\rangle = \alpha_1 |x_1\rangle + \alpha_2 |x_2\rangle + \cdots + \alpha_n |x_n\rangle = \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_n \end{bmatrix}, \quad (2.16)$$

where $\{|x_i\rangle\}$ are the computational basis states and $\alpha_i \in \mathbb{C}$ are the respective probability amplitudes so that $\sum_i |\alpha_i|^2 = 1$. It is worth noting that it is up to a measurement that the state is preserved in such a superposition. Once the state is measured, the state collapses to one of the computational basis states according to the probability amplitudes specified.

It is also possible that two or more distinct physical systems construct a composite quantum system. Suppose that the states of a component systems are defined by the set of state vectors $\{|\psi_i\rangle\}$ on the respective Hilbert spaces $\{\mathcal{H}_i\}$. The composite system's total state vector is then defined by a tensor product over the component systems' state vectors

$$\bigotimes_i |\psi_i\rangle = |\psi_1\rangle \otimes |\psi_2\rangle \otimes \cdots \otimes |\psi_n\rangle. \quad (2.17)$$

For example, two two-dimensional component systems construct a composite system's state mathematically by the following tensor product

$$\begin{bmatrix} \alpha_1 \\ \alpha_2 \end{bmatrix} \otimes \begin{bmatrix} \beta_1 \\ \beta_2 \end{bmatrix} = \begin{bmatrix} \alpha_1\beta_1 \\ \alpha_1\beta_2 \\ \alpha_2\beta_1 \\ \alpha_2\beta_2 \end{bmatrix}. \quad (2.18)$$

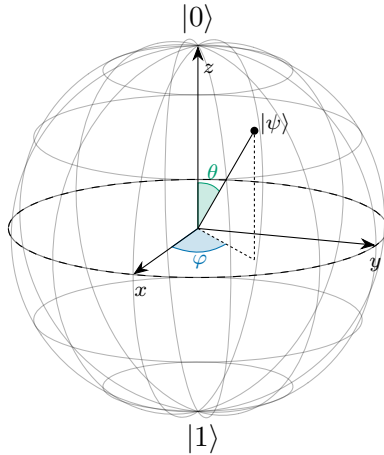


Figure 2.4: An arbitrary one qubit superposition of $|0\rangle$ and $|1\rangle$ illustrated on a Bloch sphere

It is possible to visualize qubit's state using a Bloch sphere. The visualization is based on rephrasing Equation 2.15 so that

$$|\psi\rangle = e^{i\gamma} \left(\cos \frac{\theta}{2} |0\rangle + e^{i\varphi} \sin \frac{\theta}{2} |1\rangle \right), \quad (2.19)$$

in which the variables γ , θ and φ can be interpreted as spherical coordinates. As a side note, γ is a global phase factor and has such no observable effects. Example of an arbitrary superposition of $|0\rangle$ and $|1\rangle$ illustrated using a Bloch sphere is shown in Figure 2.4.

2.4.2 Evolution of a quantum system

Evolution of a closed quantum system is determined by an unitary (Hermitian) operator

$$|\psi'\rangle = \hat{U}|\psi\rangle. \quad (2.20)$$

For instance, in quantum circuits, the gates are unitary operators acting on qubits. We will discuss the quantum circuits more in Section 2.5.1 including few example unitary operators (gates).

In the more general case, the evolution can also be defined in continuous time by the time-dependent Schrödinger equation

$$i\hbar \frac{d}{dt} |\psi\rangle = \hat{H} |\psi\rangle. \quad (2.21)$$

The unitary operator \hat{H} is commonly referred to as the Hamiltonian (total energy of the system). Particularly interesting features arise when the Hamiltonian is decomposed to

its eigenstates and eigenvalues

$$\hat{H} = \sum_i E_i |E_i\rangle\langle E_i|. \quad (2.22)$$

The eigenstates $|E_i\rangle$ are commonly referred to as excited states, and the eigenvalues E_i are the respective energies of the states. Especially, finding the state corresponding to the lowest energy, the ground state, is in the interest of many applications.

2.4.3 Density operators and mixed states

Density operators provide an alternative representation to the states. Particularly, a density operator (sometimes referred to as a density matrix) can be constructed from a state vector by an outer product

$$\rho = |\psi\rangle\langle\psi|. \quad (2.23)$$

Furthermore, a state that can be represented in such a form is a pure state. Additionally, it makes sense to highlight that pure state formalism does not consider whether the state is a superposition or not.

Density operators become useful when an ensemble of pure states describes the state of the system. That is a probabilistic mixture of pure states, called a mixed state

$$\rho = \sum_i p_i |\psi_i\rangle\langle\psi_i|. \quad (2.24)$$

While density operators introduce different semantics for quantum computing, all the quantum computing postulates can be defined using both state vectors and density matrices. For instance, the evolution of a system determined by a unitary operator can also be defined for density operators as

$$\rho \mapsto \hat{U}\rho\hat{U}^\dagger, \quad (2.25)$$

or similar to the state vector representation, component systems can also be considered as a composite system as

$$\rho_{AB} = \rho_A \otimes \rho_B. \quad (2.26)$$

To consider the component systems again, particularly convenient is tracing out parts of the density matrix. The operation is commonly referred to as partial trace, and it is very

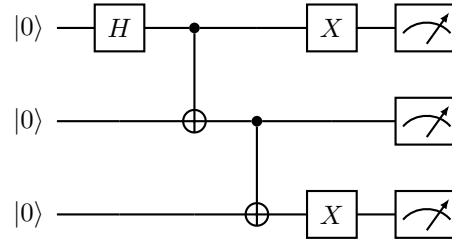


Figure 2.5: An example of a three-qubit circuit with measurements for each qubit in the end.

typical in quantum computing. For instance, with ρ_{AB} one could acquire the component system by doing a partial trace against the other systems

$$\rho_A = \text{Tr}_B(\rho_{AB}) \quad (2.27)$$

$$\rho_B = \text{Tr}_A(\rho_{AB}). \quad (2.28)$$

2.5 Quantum computing models

This section introduces two well-known quantum computing models: circuit-based quantum computing and adiabatic quantum computing. While there has been work on proving the equivalence of the models [4], the models work differently due to different kinds of hardware. Furthermore, these difference can affect the problem formulation and the used algorithms.

2.5.1 Quantum circuit

The quantum circuit model is a quantum computational model, a language, based on quantum logic gates acting on a set of qubits. The circuit language makes it possible to visualize and quantify quantum computation and algorithms easily with network-like graphs. We will introduce the model using the book [52] by Nielsen and Chuang again as the primary reference.

The quantum circuit model consists of wires (not necessarily physical) read from left to right. On the wires, the quantum logic gates represent unitary operators acting on qubits. The model is relatively similar to the classical Boolean logic gate model (consisting of gates like NOT, AND and OR). However, due to the nature of quantum computing, quantum logic gates are reversible and are represented by unitary matrices. The reversibility

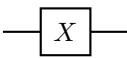
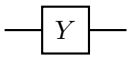
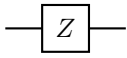
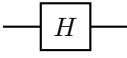
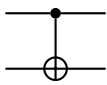
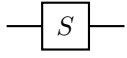
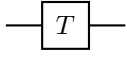
Name	Symbol	Matrix
Pauli-X (X)		$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$
Pauli-Y (Y)		$\begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}$
Pauli-Z (Z)		$\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$
Hadamard (H)		$\frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$
Controlled-NOT (CNOT)		$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$
Phase (S)		$\begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix}$
$\pi/8$ (T)		$\begin{bmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{bmatrix}$

Table 2.1: Common quantum logic gates with gate and matrix representation.

makes it impossible to have operations where wires would be joined, as such computation cannot be reversible. Furthermore, as quantum mechanics prevent the copying of qubits, a quantum circuit cannot have such copying operation, essentially splitting the wires. An example of a quantum circuit is illustrated in Figure 2.5.

There exists already plenty of different quantum logic gates for computation. Most gates are either single qubit or two-qubit gates, but there are also gates acting on more than two qubits. Notably, many of the multi-qubit gates are just “controlled” versions of some single-qubit gates. Some of the well-known gates have been summarized in Table 2.1. Many of the gates may seem first somewhat arbitrary, but they often have some interesting properties. For instance, the Hadamard gate is a quantum gate that prepares the equal superposition from the computation basis states.

In addition to the gates shown in Table 2.1, several other gates exist that are applied in different contexts. Interestingly, many of the more complex gates can be constructed from a set of simpler gates. Particularly interesting are the sets of gates that can be considered

to be universal for quantum computation. Those are gate sets capable of approximating any unitary operation to arbitrary accuracy without requiring any other gates. A standard gate set of such consists of CNOT, H, S and T -gates.

While the quantum logic gates act as the abstract building blocks of the circuit model, in practice, it is often relevant to consider which gates a particular hardware implements and how precisely. Considering the hardware is increasingly important with the small noisy devices we have today.

2.5.2 Adiabatic quantum computing

Adiabatic quantum computing is a computational model based on adiabatic evolution between Hamiltonians [4, 20]. The model relies on the (quantum) adiabatic theorem, which states that the system resides in the ground state given that the system's Hamiltonian evolves slowly enough [6]. The model makes it possible to solve ground states of potentially complicated Hamiltonians by adiabatically evolving from simple Hamiltonians. Thus if the ground state of particular Hamiltonian can encode the solution to a problem of interest, adiabatic quantum computing can solve it.

Mathematically, the quantum system is prepared to the ground state of some initial Hamiltonian, \hat{H}_{init} , which should be simple to prepare. Furthermore, the Hamiltonian of the system is gradually transformed from \hat{H}_{init} to the final target Hamiltonian \hat{H}_{final} whose ground state should encode the solution to a problem of interest. Given that the evolution is done slowly enough, the system remains in the ground state by the adiabatic theorem [6]. Thus in the end of evolution, the system represents the ground state of \hat{H}_{final} and hence solves the problem of interest.

The adiabatic evolution of the system can be described by the time-dependent Hamiltonian which is linear transformation between the initial and final Hamiltonian

$$\hat{H}(t) = \left(1 - \frac{t}{T}\right) \hat{H}_{\text{init}} + \left(\frac{t}{T}\right) \hat{H}_{\text{final}}, \quad (2.29)$$

where $t \in [0, T]$ is time and T is the total run time. As $t \rightarrow T$, by the adiabatic theorem, the state of the system, $|\psi(t)\rangle$, stays continuously very close to the ground state of $\hat{H}(t)$ and eventually approaches the ground state of \hat{H}_{final} thus solving the problem [20].

2.5.3 Quantum annealing

Quantum annealing [21, 36] is a method for approximating the global optimum of an objective function by interpreting the function as the energy of a system, Hamiltonian, and finding the ground state of it. Compared to the traditional simulated annealing [39] that uses thermal fluctuations, quantum annealing uses quantum fluctuations such as quantum tunnelling to transition between different states and eventually find the global optimum. Several studies, such as [17, 30, 36], have shown that quantum annealing can find the optimum faster and solve a broader range of problems than the traditional simulated annealing.

D-Wave systems are well-known for being suitable for quantum annealing as they can find the minimum of the given energy landscape [16]. Particularly, D-Wave systems can solve problems where the final target Hamiltonian can be expressed as an Ising model or a quadratic unconstrained binary optimization (QUBO) problem.

The quantum processing unit (QPU) of D-Wave system implements a time-dependent quantum Hamiltonian

$$\hat{H} = -\frac{A(s)}{2} \underbrace{\left(\sum_i \hat{\sigma}_i^x \right)}_{\hat{H}_{\text{init}}} + \frac{B(s)}{2} \underbrace{\left(\sum_i h_i \hat{\sigma}_i^z + \sum_{\langle i,j \rangle} J_{ij} \hat{\sigma}_i^z \hat{\sigma}_j^z \right)}_{\hat{H}_{\text{final}}}, \quad (2.30)$$

where $A(s)$ and $B(s)$ are QPU-specific annealing scheduling functions parameterized by $s = \frac{t}{T}$ in which $t \in [0, T]$ is time and T is the annealing time of single annealing run [16]. At the beginning of an annealing run $A(s) \gg B(s)$ and the time-dependent Hamiltonian is entirely defined by the \hat{H}_{init} . Thus, the lowest energy state is initially defined by the qubits being in the equal superposition state of 0 and 1. As the annealing run proceeds till the end $A(s) \ll B(s)$ and the time-dependent Hamiltonian is entirely defined by the \hat{H}_{final} which is the Ising model in the case of D-Wave. Thus, the lowest energy state in the end is the solution for the Ising model.

2.6 Quantum Boltzmann machine

Quantum Boltzmann machine (QBM) [7] is a quantum generalization of the (classical) Boltzmann machine that we introduced in Section 2.3. However, the definition is not so unambiguous as the subsequent work has shown different formulations and extensions for

the QBM, especially when it comes to the training routines. These have naturally inherited from the different quantum computing models and the diverse purposes that the models are applied. Furthermore, these ambiguities in the model formulations are a fundamental root cause of why the definition of QML is generally complex. In this section, we will introduce the QBM using primarily the model introduced in [7], that is to our knowledge the first proposal using quantum computing for both the training and the model itself. Furthermore, we will highlight the notable extensions following that work with additional references. It is worth to note, though, that there has been a lot of earlier work considering the Boltzmann machines in quantum setup, i.e. [3, 9]. However, they have not been so end-to-end complete and considered only a single aspect like training in quantum setup while the model has still been classical [7].

The QBM introduced in [7] is based on modelling the data using a transverse field Ising model [62] in thermal equilibrium. The Hamiltonian of the model is then given by

$$\hat{H} = - \sum_{\langle i,j \rangle} J_{ij} \hat{\sigma}_i^z \hat{\sigma}_j^z - \sum_i h_i \hat{\sigma}_i^z - \sum_i \Gamma_i \hat{\sigma}_i^x, \quad (2.31)$$

where $\hat{\sigma}_i^{\{x,z\}}$ are the Pauli-X and Z matrices acting on qubits, $J_{ij} \in \mathbb{R}$ is the interaction strength and $h_i, \Gamma_i \in \mathbb{R}$ are the external field strengths. Especially, Γ_i is the strength of the transverse field σ_i^x , which provides the quantum fluctuations to the model.

That is, the QBM model is essentially realized from the classical BM by replacing the (classical) Ising model, which BM is identical to, with a quantum version of the Ising model, transverse field Ising model. The particular Ising model then represents the system's total energy, the Hamiltonian. This energy formalization also illustrates the apparent relation to the family of energy-based models. The transverse field Ising model is parametrized by weights and biases similar to the (classical) Ising model or the classical BM we introduced earlier in Section 2.3. Particularly, it is parameterized by $\theta = \{J_{ij} = w_{ij}, h_i = b_i, \Gamma_i\}$ that are coupling strengths and qubit biases (w_{ij} and b_i added for notational connection to (classical) Equation 2.6). Effectively these parameters weigh the Pauli operators that are acting on the qubits in some state. The coupling strengths are essentially the weights of two-qubit interactions, while the qubit biases provide the effect that is applied trivially on single qubits. Intuitively, it could be thought that the qubits in QBM represent the units of BM when reflecting against the (classical) BM. An example of the transverse field Ising model for three qubits could be expanded as

$$\hat{H} = J_{12} \hat{\sigma}_1^z \hat{\sigma}_2^z + J_{13} \hat{\sigma}_1^z \hat{\sigma}_3^z + J_{23} \hat{\sigma}_2^z \hat{\sigma}_3^z + h_1 \hat{\sigma}_1^z + h_2 \hat{\sigma}_2^z + h_3 \hat{\sigma}_3^z + \Gamma_1 \hat{\sigma}_1^x + \Gamma_2 \hat{\sigma}_2^x + \Gamma_3 \hat{\sigma}_3^x, \quad (2.32)$$

in which the first three terms define the coupling strengths between two qubits and the rest of the terms define the qubit biases. Especially the Pauli-X operators are providing the quantum fluctuations to the QBM model. From a mathematical point of view, the Hamiltonian can be expanded further even to matrix representation, for example for the first term of Example 2.32

$$J_{12}\hat{\sigma}_1^z\hat{\sigma}_2^z = J_{12} \cdot \hat{\sigma}^z \otimes \hat{\sigma}^z \otimes I = J_{12} \cdot \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}. \quad (2.33)$$

These terms of Pauli operators are often referred to as Pauli terms or Pauli strings, and the whole sum consisting of Pauli strings weighted by coefficients is commonly referred to as Paulisum.

In some of the subsequent work on QBMs, i.e. [37, 73] more generalized parametrized Hamiltonians than the transverse field Ising model have been used for QBM models. For instance, the (Variational) QBM introduced in [73] is suitable with Hamiltonians with arbitrary Pauli terms weighted by real coefficients, which means that the interactions and biases can be specified by any combinations of Pauli operators ($\hat{\sigma}^x$, $\hat{\sigma}^y$ and $\hat{\sigma}^z$) acting on qubits. Additionally, in [37], it is denoted that using the transverse field Ising models for QBM do not introduce a clear quantum advantage because it is commonly assumed that quantum Monte Carlo methods can be used for simulating them efficiently.

Nevertheless, these parameters of the Hamiltonians are the ones that finally introduce the optimization aspect of the QBM model. That being to optimize the parameters so that the probability distribution determined by the Hamiltonian, such as the transverse field Ising model, in thermal equilibrium, would be as close as possible to the target probability distribution defined by the true data.

The QBM notation follows the density matrix representation that we introduced in Section 2.4.3. Specifically, the state of the QBM model in thermal equilibrium, commonly referred to as the Gibbs state, is represented by a density matrix

$$\rho = \frac{e^{-H}}{\text{Tr}(e^{-H})}, \quad (2.34)$$

in which the diagonal elements represent the Boltzmann probabilities of all 2^N possible basis states of N qubits and particularly H is the Hamiltonian, i.e. in [7] the transverse field Ising model. Similarly to the classical case, the normalization is guaranteed in the denominator by the tracing over the exponentiation, thus yielding a valid probability distribution. In the quantum setup, a useful definition for this was shown in [69] where the model consists of visible and hidden component systems of qubits. This way, the Gibbs state of the model is defined by tracing out the hidden subsystem in the numerator also. After applying the partial trace, the Gibbs state of the visible component system only is given by

$$\rho_v = \frac{\text{Tr}_h(e^{-H})}{\text{Tr}(e^{-H})}. \quad (2.35)$$

It is worth noting that the formulation is actually very close to the classical one shown in Equation 2.8.

Furthermore, the probability of measuring a certain state $|v\rangle$ is given by a projective measurement against the Gibbs state. That is the probability given by trace

$$P_v = \text{Tr}(\Lambda_v \rho) \quad (2.36)$$

using a projector of form

$$\Lambda_v = |v\rangle\langle v| \otimes \mathcal{I}_h, \quad (2.37)$$

which is essentially a diagonal matrix acting on the visible qubits in state v and all the qubits in hidden component system.

Up to the model definition, QBMs are defined very similarly across the literature. However, when it comes to the QBM model training, different setups with different characteristics and limitations exist. Similarly, as with most ML models, the training of QBM is based on minimizing an objective function which we already discussed in Section 2.1. In the following sections, we will introduce some of the well-known methods for training QBMs. Firstly we begin in Section 2.6.1 by introducing the methods for learning classical distributions with QBM. Then we continue in Section 2.6.2 by introducing the extensions for learning quantum probability distributions. Finally, in Section 2.6.3, we introduce and discuss a recent study on a variational learning procedure [73].

2.6.1 Learning classical probability distribution

In this section, we will continue introducing the QBM model training by primarily using [7] as it mainly focuses on learning classical probability distributions.

The QBM introduced in [7] uses average negative log-likelihood (NLL) as an objective function that is essentially the same as cross-entropy in this context

$$\mathcal{L} = - \sum_v P_v^{\text{data}} \log(P_v) \quad (2.38)$$

$$= - \sum_v P_v^{\text{data}} \log \left(\frac{\text{Tr}(\Lambda_v e^{-H})}{\text{Tr}(e^{-H})} \right). \quad (2.39)$$

Particularly, the gradient of NLL with respect to model parameters is given by

$$\partial_\theta \mathcal{L}(\theta) = \sum_v P_v^{\text{data}} \left[\frac{\text{Tr}[\Lambda_v \partial_\theta e^{-H(\theta)}]}{\text{Tr}[\Lambda_v e^{-H(\theta)}]} - \frac{\text{Tr}[\partial_\theta e^{-H(\theta)}]}{\text{Tr}[e^{-H(\theta)}]} \right] \quad (2.40)$$

$$= \sum_v P_v^{\text{data}} \left[\frac{\text{Tr}[\Lambda_v \partial_\theta e^{-H(\theta)}]}{\text{Tr}[\Lambda_v e^{-H(\theta)}]} - \text{Tr}[\rho \partial_\theta H(\theta)] \right] \quad (2.41)$$

in which the second term inside the main brackets is a parametrized exponential operator; hence it is possible to derive it further as shown in [70]. Furthermore, it is possible to evaluate it efficiently by sampling from the Gibbs distribution ρ defined in Equation 2.34. However, the gradient's first term is particularly troublesome because it requires a separate calculation for each training sample, meaning, for instance, exact diagonalization or quantum Monte Carlo sampling [7]. Therefore it is concluded in [7] that the training of QBM for a large dataset is computationally expensive when using such a gradient.

To overcome the inefficiency in gradient evaluation, two methods are proposed in [7]. The first one is based on defining an alternative objective function that acts as an upper bound for the exact NLL objective. Hence, by minimizing this upper bound, the exact NLL is minimized to some extent and yields a reasonable solution. The upper bound used in [7] is effectively based on the Golden-Thompson inequality [25, 65]

$$\text{Tr}(e^{A+B}) \leq \text{Tr}(e^A e^B), \quad (2.42)$$

where A and B are Hermitian matrices. By first defining the projector in the exponential form $\Lambda_v = e^{\ln(\Lambda_v)}$, and then utilizing the Golden-Thompson Inequality 2.42, it is effectively possible to combine the projector $e^{\ln(\Lambda_v)}$ into the full Hamiltonian exponential e^H , hence introducing a new bound by the inequality. More precisely, in [7], a new clamped Hamiltonian H_v is defined that can be used for further defining the upper bound for the exact loss

$$\mathcal{L} \leq \tilde{\mathcal{L}} = - \sum_v P_v^{\text{data}} \log \left[\frac{\text{Tr}(e^{-H_v})}{\text{Tr}(e^{-H})} \right]. \quad (2.43)$$

As there is no more projector inside the trace operation, instead, it is induced into the clamped Hamiltonian. Hence, it is possible to derive both of the terms inside the gradient's sum (Equation 2.40) further on to

$$\partial_\theta \tilde{\mathcal{L}}(\theta) = \sum_v P_v^{\text{data}} \left[\frac{\text{Tr} [\partial_\theta e^{-H_v(\theta)}]}{\text{Tr} [e^{-H_v(\theta)}]} - \frac{\text{Tr} [\partial_\theta e^{-H(\theta)}]}{\text{Tr} [e^{-H(\theta)}]} \right] \quad (2.44)$$

$$= \sum_v P_v^{\text{data}} [\text{Tr}(\rho_v \partial_\theta H_v(\theta)) - \text{Tr}(\rho \partial_\theta H(\theta))]. \quad (2.45)$$

Therefore, both of the terms inside the sum can be evaluated efficiently by sampling from Gibbs distribution. That is, the first one using the clamped Hamiltonian H_v and the second one using the full Hamiltonian H .

However, while the bounded loss introduces a gradient that is more efficient to evaluate than the exact objective, it also introduces problems, particularly following two problems related to the transverse field were highlighted in [7]. Firstly, as the sampling is done on σ_i^z -basis, one cannot calculate the expectations for σ_i^x without doing measurements on the same basis. Hence, measurements are needed on σ_i^x -basis. Secondly, the transverse field's coefficients suffer from vanishing gradient, hence effectively yielding them to be zero. Furthermore, as pointed in [37], it is impossible to learn the transverse field from classical data without utilizing brute-force techniques that make it hard to find the full Hamiltonian.

The second solution proposed in [7] is somewhat inherited from classical BM. That is, introducing restrictions to the connections inside the model. In classical BM typically the restricted variant is the RBM model that we introduced in Section 2.3.1. However, in [7] the lateral connection restriction is only applied for the hidden units (commonly referred as semi-restricted), while in (classical) RBM the restriction is applied to both hidden and visible units.

Notably, utilizing the bounded loss requires sampling using the clamped Hamiltonian H_v that needs to be done per data sample; therefore, as highlighted previously for exact gradient (Equation 2.40), thus setup is computationally expensive for large datasets. In [7], by applying the restriction, this challenge is overcome as the clamped Hamiltonian H_v reduces to a simple form of which σ_i^z expectations can be calculated exactly, thus making learning much more efficient.

2.6.2 Learning quantum probability distribution

Recently, there has been work, i.e. [37, 69], on generalizing the QBM training to quantum probability distributions. That is essentially learning the non-diagonal entries of the density matrix. Furthermore, the extensions provide a way to learn the transverse field from the data, which was not possible in the method proposed in [7] that we introduced in the previous section. This section will introduce the methods for learning quantum probability distributions by primarily using [37, 69].

Particularly, in [37], two new approaches for QBM learning are proposed. The first one is based on using a completely different objective function from the NLL introduced in previous section. That is the quantum relative entropy (QRE)

$$S(\rho^{\text{data}}\|\rho) = \text{Tr} \left[\rho^{\text{data}} \log(\rho^{\text{data}}) \right] - \text{Tr} \left[\rho^{\text{data}} \log(\rho) \right], \quad (2.46)$$

which is a quantum generalization of the (classical) Kullback–Leibler (KL) divergence. In fact, QRE reduces to KL-divergence in the case of diagonal matrices, and holds similar properties to KL-divergence; for example, $S(\rho\|\sigma) = 0 \Leftrightarrow \rho = \sigma$. Furthermore, as essentially the first term is constant and defined by the data only, the actual objective comes from the second term only. Thus, the gradient with respect to the model parameters is defined by

$$\partial_{\theta} S(\rho^{\text{data}}\|\rho(\theta)) = \partial_{\theta} \text{Tr} \left[\rho^{\text{data}} \log \frac{e^{-H(\theta)}}{\text{Tr}[e^{-H(\theta)}]} \right] \quad (2.47)$$

$$= -\text{Tr} \left[\rho^{\text{data}} \partial_{\theta} H(\theta) \right] + \frac{\text{Tr} \left[e^{-H(\theta)} \partial_{\theta} H(\theta) \right]}{\text{Tr} \left[e^{-H(\theta)} \right]} \quad (2.48)$$

$$= -\text{Tr} \left[\rho^{\text{data}} \partial_{\theta} H(\theta) \right] + \text{Tr} \left[\rho \partial_{\theta} H(\theta) \right]. \quad (2.49)$$

That is, the gradient is effectively determined by the difference between expectations of Hamiltonian terms in data distribution ρ^{data} and the model distribution ρ . Furthermore, as pointed out in [37] and is commonly applied practice, it is possible to add a regularization term to the objective function to penalize large coefficients, for instance, an l2-norm of the coefficients.

However, the introduced QRE so far is only applicable for an all-visible model without hidden units. The extension including the hidden units is proposed in [69], and essentially means replacing the whole state ρ (defined in Equation 2.34) with only the visible state ρ_v (defined in Equation 2.35) where hidden component system is traced out. The problem, however, is that once e^{-H} is replaced by partial trace $\text{Tr}_h(e^{-H})$ inside logarithm, it does

not simplify anymore when evaluating the gradient. In [69], two approaches are proposed for overcoming this challenge and introducing the hidden units:

1. By assuming a specific form for the Hamiltonian, specifically, that Hamiltonian's non-commuting terms only act on visible units and, respectively, commuting terms act on hidden units, it is possible to define a variational bound for the QRE for which the derivatives are simple to evaluate.
2. Generic method for approximating the exact gradient by high-order divided difference estimates and Fourier-like approximation.

Both methods are efficient, given that Gibbs state can be prepared efficiently, and Hamiltonian's matrix elements can be calculated efficiently. We refer the readers to [69] and especially to the supplementary material in the study for details and the derivations.

The second method proposed in [37] is positive operator-valued measure (POVM)-based Golden-Thompson training. That is essentially an extension to the bound-based method proposed in [7]. The realization in [37] is that as the gradients could not be calculated for Equation 2.44 in [7] in the case when $\text{Tr} [\partial_\theta e^{-H_v(\theta)}] = 0$, POVMs could provide a better-suited way to express the dataset so that such issues are prevented. It is possible to circumvent the problem by introducing POVM elements that are non-diagonal [37].

Formally, in [37], the POVM-based training set is defined in Hilbert space $\mathcal{H} = \mathcal{H}_V \otimes \mathcal{H}_L$ consisting of both visible and hidden component systems by probability distribution P_v and POVM $\Lambda = \{\Lambda_v\}$ if (i) there exists a bijection between the domain of P_v and Λ , and (ii) the domain of each Λ_v is \mathcal{H} and it acts nontrivially only on visible component system \mathcal{H}_V . The clarifying example presented in [37] is considering the problem of learning to generate even numbers between 1 and 16. It is possible to define the training set either by

$$\Lambda_n = |2n\rangle\langle 2n|, \quad 1 \leq n \leq 8 \quad (2.50)$$

$$\Lambda_0 = \mathcal{I} - \sum_{n=1}^8 \Lambda_n, \quad P_v = \frac{1 - \delta_{v,0}}{8} \quad (2.51)$$

or

$$\Lambda_1 = \frac{1}{8} (|2\rangle + \dots + |16\rangle) (\langle 2| + \dots + \langle 16|) \quad (2.52)$$

$$\Lambda_0 = \mathcal{I} - \Lambda_1, \quad P_v = \delta_{v,1}. \quad (2.53)$$

Both of them are essentially similar in terms of the dataset they are representing. However, as shown in [37], only the second one circumvents the problem of learning the coefficients of transverse terms.

Additionally, an exceptional finding was made in [37] that increasing the number of hidden units in the QBM setup did not benefit similarly as in the classical setup. While the training set’s simplicity initially explained this observation in [37], a more recent study [45] investigated this further and showed that the QBMs do not benefit from multiple hidden units.

2.6.3 Learning using variational principle

In a recent study [73], a completely new variant of QBM is proposed. That is a variational QBM (VarQBM) based on variational quantum algorithms. VarQBM is compatible with generic Hamiltonians, including arbitrary Pauli terms parametrized by real coefficients. Additionally, VarQBM is effectively compatible with the NISQ era quantum hardware, meaning it should be possible to run QBM calculations with small noisy circuits.

The VarQBM model is defined similarly as QBMs we have discussed so far. Particularly, projective measurements are done in VarQBM using the similar projectors as defined in Equation 2.37. In [73], the VarQBM model is mainly being experimented on training Gibbs states that reflect classical probability distributions in their sampling behaviours. Hence, the cross-entropy is utilized as a loss function for the model. However, the authors of VarQBM suggest that their method could be adapted for a quantum probability distribution using an objective function based on QRE that was introduced in Equation 2.46 [73].

In the core of VarQBM is the variational quantum imaginary time evolution (VarQITE) procedure used for preparing approximations of Hamiltonians’ thermal (Gibbs) states [46, 72]. The VarQITE routine begins with an expressive variational quantum circuit V parameterized by ω . Notably, the circuit needs to be initialized to prepare a maximally mixed state. In [72, 73], this is achieved by dividing the system to two separate component systems, a and b , where the pairs of qubits between the systems are initially entangled (Bell states) $|\Phi^+\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$, and the problem Hamiltonian acts only to component system a . Then tracing out the second component system b from the composite system produces a maximally mixed state

$$\text{Tr}_b \left[|\Phi^+\rangle^{\otimes n} \right] = \frac{1}{2^n} I. \quad (2.54)$$

The essential factor for preparing the thermal states is to evolve the state only for a finite time of $\tau_{\text{total}} = \tau = (2k_b T)^{-1}$ based on system temperature T . Otherwise, doing it for $\tau \rightarrow \infty$, the evolution would converge to the ground state, i.e. the state corresponding

to the lowest energy (sought in many applications). Furthermore, the evolution is split into a number of timesteps, N_τ , which essentially defines how many iterations VarQITE includes and the size of parameter updates $\delta\tau = \tau_{\text{total}}/N_\tau$ per iteration (timestep).

For actually evolving the maximally mixed state to a thermal state with VarQITE, the parameter update rules can be derived from McLachlan’s variational principle [48]. For the derivations, we refer readers to the studies [46, 48, 72]. The derivations lead to a system of linear equations (adapted to the notation we have introduced so far)

$$\sum_j A_{ij}\omega_j = C_i, \quad (2.55)$$

in which

$$A_{ij} = \Re \left[\frac{\partial \langle \psi(\omega(\tau)) | \partial \psi(\omega(\tau)) \rangle}{\partial \omega_i} \frac{\partial \langle \psi(\omega(\tau)) | \partial \psi(\omega(\tau)) \rangle}{\partial \omega_j} \right], \quad (2.56)$$

$$C_i = \Re \left[- \sum_p \theta_p \frac{\partial \langle \psi(\omega(\tau)) | h_p | \psi(\omega(\tau)) \rangle}{\partial \omega_i} \right]. \quad (2.57)$$

Here, $|\psi(\omega(\tau))\rangle$ is the state parameterized by ω at time τ and h_p is a Pauli term of the problem Hamiltonian having θ_p as a coefficient. The elements of A and C can be evaluated using quantum circuits of a specific form and calculating the expectations. However, the evaluation of matrix A is costly, scaling quadratically to the number of circuit parameters ω [22]. A recent proposal [22] investigates the possibility of using a stochastic approximation for evaluating it.

Solving Equation 2.55 for the ω gives the update for the circuit parameters, which can be applied for instance with Euler method

$$\omega(\tau + \delta\tau) = \omega(\tau) + A^{-1}(\tau)C(\tau)\delta\tau. \quad (2.58)$$

Furthermore, for solving Equation 2.55, which is potentially ill-posed, regularization methods such as Tikhonov regularization [66] can be applied.

After reaching the end of the evolution, the thermal state can be retrieved again by tracing out the second subsystem b . One of the benefits of VarQITE is that the thermal state it prepares is inherently normalized; hence the partition function is not needed to be evaluated explicitly for training VarQBM [73].

Having introduced and discussed VarQITE, the core of the VarQBM model, the following three steps will outline the three-stage VarQBM training routine proposed in the particular study [73] (adapted to the notation we have introduced so far).

1. VarQITE is used to prepare the approximation of the Gibbs state ρ_ω by using a variational circuit $V(\omega)$

$$\rho_\omega = \text{Tr}_b [|\psi(\omega(\tau))\rangle\langle\psi(\omega(\tau))|] \approx \rho, \quad (2.59)$$

in which $|\psi(\omega(\tau))\rangle$ is the trial state propagated by VarQITE using circuit $V(\omega)$ for time τ and Tr_b the trace over the second component system b . As pointed out in [73], the approximation is profoundly determined by the representation capabilities of the parametrized state.

2. For updating the Hamiltonian parameters θ , the gradient of the loss functions with respect to θ needs to be evaluated

$$\partial_\theta \mathcal{L} = \partial_\theta \left[- \sum_v P_v^{\text{data}} \log(P_v) \right] \quad (2.60)$$

$$= - \sum_v P_v^{\text{data}} \frac{\partial_\theta P_v}{P_v}. \quad (2.61)$$

By using the chain rule, $\partial_\theta P_v$ can be expanded furthermore, so that

$$\partial_\theta P_v = \frac{\partial P_v}{\partial \omega(\tau)} \frac{\partial \omega(\tau)}{\partial \theta} \quad (2.62)$$

$$= \frac{\partial \text{Tr}(\Lambda_v \rho_\omega)}{\partial \omega(\tau)} \frac{\partial \omega(\tau)}{\partial \theta}, \quad (2.63)$$

in which the first factor can be calculated by quantum gradient methods [43, 49, 57]. Meanwhile, the second factor can be factorized furthermore depending on the structure of the quantum circuit V , and it can be evaluated analytically by VarQITE. Furthermore, for the loss function, P_v can be calculated from the approximated Gibbs state ρ_ω .

3. After calculating the gradient $\partial_\theta \mathcal{L}$, the Hamiltonian parameters θ can be updated using a classical optimizer. After which, the learning continues again from step 1 with updated parameters.

One of the benefits highlighted in [73] is the possibility of using analytic gradients even through VarQITE and further enable the usage of an automatic differentiation scheme, which is especially useful in step 2, where chain rule is utilized. However, depending on the form of Hamiltonian and the number of parameters, it might be more beneficial in terms of the computational cost to use finite difference methods for evaluating the gradient [73].

3 Methods and algorithms

3.1 Variational quantum Boltzmann machine

In order to do experiments with QBM, we implemented it from scratch. Particularly, we decided to implement variational quantum Boltzmann machine (VarQBM) [73], which we introduced in Section 2.6.3. VarQBM seemed to be the most end-to-end complete for practical usage, and the study claimed it to be suitable for NISQ era devices, which made it ideal for us to focus.

The core algorithm on which VarQBM relies is the variational quantum imaginary time evolution (VarQITE) which we introduced in Section 2.6.3 jointly with VarQBM. In VarQBM, VarQITE is used for preparing the thermal states of problem Hamiltonians. So more accurate and efficient the VarQITE, the better the results should be for VarQBM itself. For proceeding with the VarQBM implementation and experiments, we focused initially on experimenting with VarQITE only as it is a crucial element of VarQBM. After experimenting with VarQITE and finding the optimal setup, we only began the experiments on VarQBM.

3.2 Experimentation setup

For experimentation setup, we began our study by entirely focusing on VarQITE. Particularly, we focused on identifying first an efficient and well-performing variational circuit for VarQITE using two different problem Hamiltonians. Those were, firstly, the transverse field Ising model, which we introduced earlier, as well as the Heisenberg model used for studying magnetic systems [8]. Using different models provided us with a way to analyze the VarQITE performance across different Hamiltonians and understand whether different circuit structures could perform differently with diverse Hamiltonians. In addition to using two different problem Hamiltonians, we repeated the experiments 40 times with resampled random coefficients for Hamiltonians. For experimenting with various circuit structures, we considered three directions for variations: layer count, two-qubit gates and the topologies in which the two-qubit gates act on qubits. Similar to the original VarQBM

study [73], we fixed our effective temperature $k_b T = 1$ for all the experiments. Furthermore, as we were using VarQITE in VarQBM, where it is run numerous times, we wanted to achieve relatively precise solutions with as few as possible timesteps. Hence for the VarQITE experiments, we fixed the number of timesteps to 20. However, in the subsequent experiments, this was also varied by experimenting with even fewer timesteps. For all the VarQITE experiments, we used general mixed state fidelity

$$F(\rho, \sigma) = \left[\text{Tr} \sqrt{\sqrt{\rho} \sigma \sqrt{\rho}} \right]^2, \quad (3.1)$$

quantum relative entropy

$$S(\rho \parallel \sigma) = \text{Tr} [\rho \log(\rho)] - \text{Tr} [\rho \log(\sigma)] \quad (3.2)$$

and trace distance

$$T(\rho, \sigma) = \frac{1}{2} \text{Tr} \left[\sqrt{(\rho - \sigma)^\dagger (\rho - \sigma)} \right] \quad (3.3)$$

as our performance measures, which we calculated against the exact solution obtained from solving the matrix exponential. Mainly, fidelity was used as the primary measure for comparing the VarQITE models.

After identifying an efficient and well-performing variational circuit for VarQITE, we focused on making the algorithm more efficient, making it run faster. Specifically, we applied a stochastic approximation method, simultaneous perturbation stochastic approximation (SPSA) [61], for speeding up the evaluation of the quantum Fisher information (QFI) matrix (matrix A in Equation 2.55), which is done on every timestep of the VarQITE process [22]. To evaluate the usage of SPSA for VarQITE, we considered three directions for variations to evaluate how much we can approximate without sacrificing precision significantly. Particularly, we varied the number of SPSA approximations for the QFI matrix to average over, the number of timesteps to divide the evolution time in VarQITE and the amount of perturbation used in SPSA.

Finally, given the efficient and well-performing variational circuit accompanied with SPSA for providing additional speedups with stochastic approximations, we began the experiments with VarQBM itself. For VarQBM, we decided to use the Ising model as the problem Hamiltonian because it is a rather expressive model and fairly natural generalization from the classical Boltzmann machine to quantum setup in terms of formulation. As a dataset, we used the toy dataset of blocks and stripes (BAS) on four qubits (two by two), which meant we had to perform simulation on eight qubits. As the data is classical, we decided to reduce the Hamiltonian only to include the Pauli-Z operators for one-qubit and two-qubit

interactions. We also tried initially running the same problem on the dataset with three by three blocks and stripes, but that meant we had to run circuits of 18 qubits, which became increasingly slow to iterate over with simulators.

Even though VarQBM should be compatible with analytic gradients and automatic differentiation, we decided to use the finite difference approximation to evaluate the gradients. Based on the original study [73] on VarQBM, using finite difference was mentioned to be computationally more efficient given that there are more Hamiltonian parameters than trial state parameters. For updating the Hamiltonian coefficients, we applied standard vanilla gradient descent. The training process was run a maximum of 60 epochs. However, we used early stopping criteria for terminating non-converging runs.

For experimenting with VarQBM, we chose to experiment with few directions again. Firstly, we chose to freeze the model in terms of topology and entanglement gates and try with different layer counts; secondly, we tried with and without SPSA for VarQITE. Otherwise, we used hyperparameters which worked the best in the VarQITE experiments.

3.3 Implementation

For implementing VarQITE and VarQBM, we decided to use Python with Tensorflow Quantum (TFQ) [12] and Google Cirq [15] -libraries as the core of our implementation. We chose TFQ as it seemed superior in performance given the highly optimized and somewhat parallelized C++ backend code. Furthermore, we considered TFQ ideal because of its tight connections to Tensorflow [1] (core) itself, which we could use further for any custom tensor calculations, including evaluating Jacobians or Hessians. Additionally, we used Tensorflow probability [18], Numpy [28], SciPy [67] and Pandas [47, 64] on a few occasions, and Seaborn [68] together with Matplotlib [14, 34] to produce meaningful figures on the experiments. In the implementation, we focused heavily on modularity to provide means for easy experimentation and upgradeability. We also note that the studies related to evaluating gradients in quantum circuits [43, 49, 57] and higher-order derivatives [44] were critical from the implementation point of view, even though some of the functionalities are provided out of the box by TFQ.

We carried out all the experiments mainly on Google Cloud virtual machines where we had allocated compute-optimized C2 virtual machines for running the experiments. Particularly, we used virtual machines with eight virtual CPUs of the Intel Cascade Lake -platform and 32GB of RAM.

4 Results

In this chapter, we report the results of the experiments which we described in previous sections. We will begin with the experiments on VarQITE and continue further with the VarQBM experiments.

4.1 Preparing thermal states using VarQITE

In our first experiments, we used the VarQITE algorithm for preparing thermal states of problem Hamiltonians. We began from a similar quantum circuit that was used in the original VarQBM study [73]. That is, a parameterized quantum circuit having parameterized Y and Z -rotation gates (RY, RZ) applied on each qubit and CNOT gates in chain topology introducing entanglement between qubits. We shall also once highlight again that the circuit consists of two sets of qubits: the first half of qubits is for the actual problem Hamiltonian, while the second half is used for preparing the maximally mixed state. Particularly, we used a total of eight qubits ($4 + 4$) in our experiments. The maximally mixed state is prepared using the last rotation gates and the CNOT gates applied between the two sets of qubits. In our setup, we also introduced additional parameters for CNOT gates' exponents in case they would provide additional degrees of freedom to the model. An example of the described quantum circuit (scaled down to six qubits) is shown in Figure 4.1.

For experimenting with different quantum circuits, we considered three directions to extend or modify the baseline setup. These were, firstly, increasing the layer count; secondly, attempting different two-qubit gates for introducing entanglements between qubits; and finally, applying the two-qubit gates in different topologies. To be more precise, we tried circuits with one, two, and three layers for layer counts; for two-qubit gates, we experimented with the gates shown in Table 4.1; and for topologies, we applied the topologies illustrated in Figure 4.2.

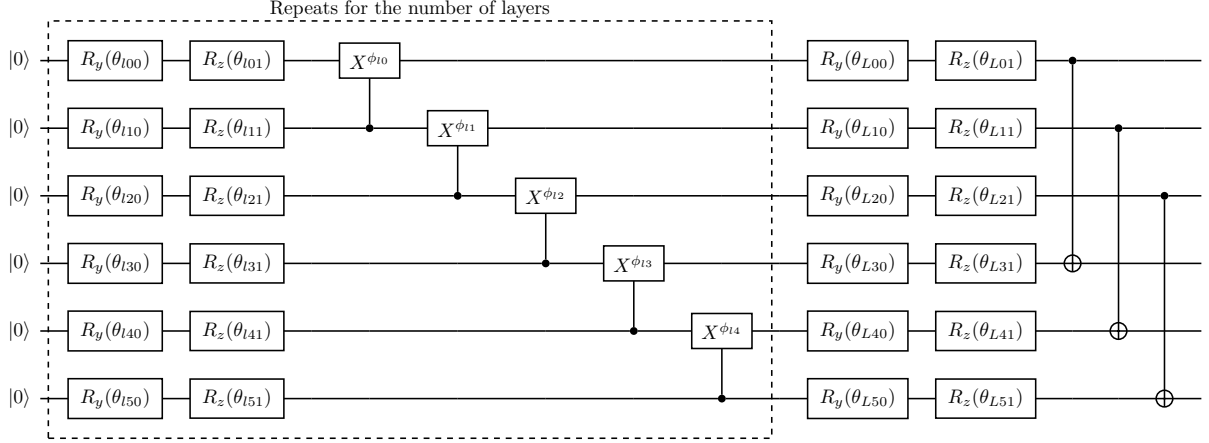


Figure 4.1: An example of a quantum circuit (three qubits for the problem itself and another three for the ancillary system) used for VarQITE. The circuit is scaled down to six qubits for convenience.

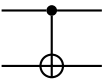
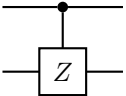
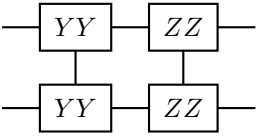
Gate	Description
	The controlled-NOT gate (CNOT) is often applied for introducing entanglement. The CNOT is commonly also referred to as the controlled-X gate (CX).
	The controlled-Z gate (CZ) is another controlled Pauli -gate. CZ can be made from CX by wrapping it with Hadamard -gates (H).
	Two qubit interactions (YY and ZZ) are made from tensor products of particular Pauli matrices ($\hat{\sigma}^y \otimes \hat{\sigma}^y$, $\hat{\sigma}^z \otimes \hat{\sigma}^z$). These are commonly referred to as the parity gates (Y-parity and Z-parity).

Table 4.1: Different gate sets that we applied for two-qubit interactions and for introducing entanglements to the model. Parameters (exponents) are omitted for clarity.

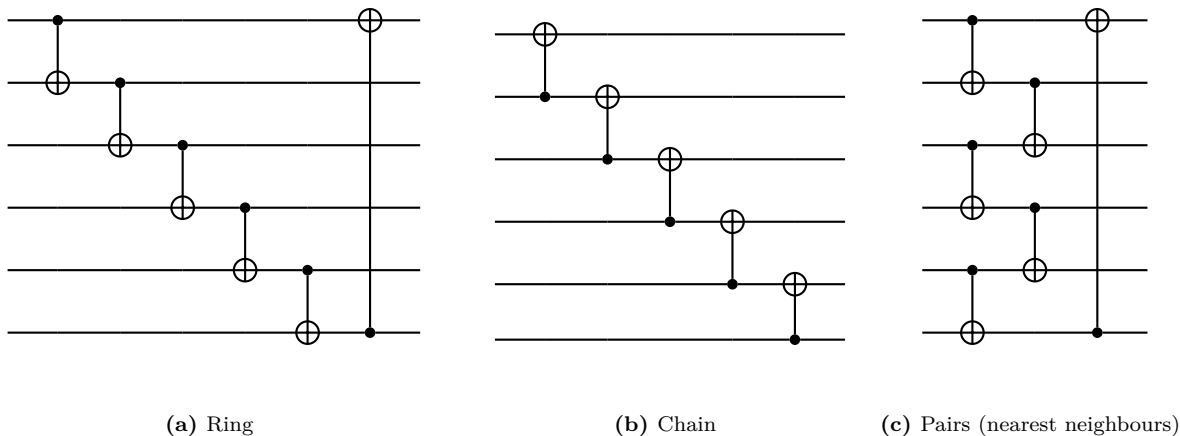


Figure 4.2: Different topologies we experimented with for two-qubit interactions included (a) ring topology, (b) chain topology and (c) pairs (nearest neighbours) topology. Parameters (exponents) are omitted for clarity.

4.1.1 Preparing thermal states of Ising model

As a primary problem Hamiltonian, we considered the transverse field Ising model, which we introduced previously. For convenience, the exact Hamiltonian is defined as

$$\hat{H} = - \sum_{\langle i,j \rangle} J_{ij} \hat{\sigma}_i^z \hat{\sigma}_j^z - \sum_i h_i \hat{\sigma}_i^z - \sum_i \Gamma_i \hat{\sigma}_i^x. \quad (4.1)$$

For the experiments, we sampled the coefficients from uniform distribution $J_{ij}, h_i, \Gamma_i \sim \mathcal{U}(-1, 1)$ (half-open interval) to acquire generalized results.

The results of preparing thermal states of the transverse field Ising models with VarQITE using various model setups are summarized in Figure 4.3 with a grid of plots showing the fidelity against the exact state (solution) through the evolution. Particularly, the grid is constructed so that the rows represent the number of layers used, while the different columns represent the different entanglement topologies used. Finally, the different coloured lines showcase the use of different entanglement gates. From the plots, we extracted the following results:

- None of the models failed to converge or performed poorly as all the model variants evolved to some peak fidelity value, and the lowest mean value on the last timestep was 0.905 while the highest was 0.986. Particularly, models using CNOT gates for entanglement in chain topology seemed to perform the best. Specifically, the three-layer variant using CNOT gates shown in 4.3h was the best model out of all.
- Generally, models did not benefit much from increasing the number of layers, except

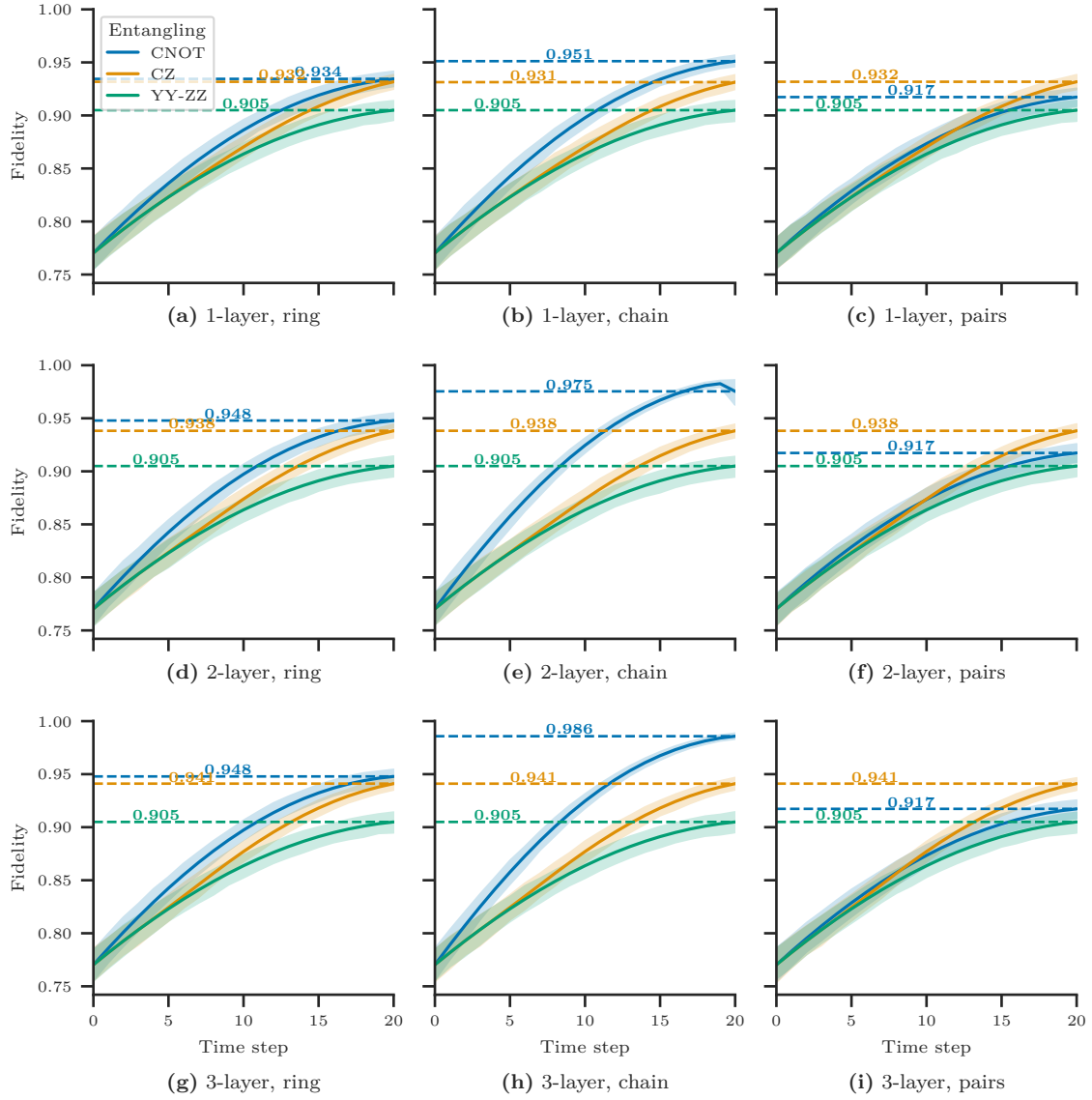


Figure 4.3: Fidelity against the exact thermal states of the transverse field Ising models through VarQITE across all the different circuit setups that we experimented with. The number of layers is varied on the rows (1, 2 and 3), while entanglement topology is varied on the columns (ring, chain and pairs). Finally, the three different coloured lines represent the different gates used for entanglement (CNOT, CZ and YY+ZZ). Furthermore, the solid coloured lines represent the means across all the experiment runs, and the shaded areas represent the 90% confidence intervals. Additionally, the last values of the means are highlighted with horizontal lines. The figures indicate that models using CNOT gates for entanglement in chain topology seemed to perform the best. Specifically, the three-layer variant (h) using CNOT gates was the best model out of all.

for the models using CNOT gates and either ring or chain topology for entanglement. Especially when increasing the number of layers from two to three, the increases in the fidelities are substantial. For instance, the model using CNOT gates and ring topology for entanglement had exactly same mean fidelity value in the last timestep when using three layers compared to two layers.

- The YY-ZZ parity gates for entanglement seemed to perform the same across different amounts of layers and entanglement topologies. At the same time, the particular model had the lowest fidelity value, 0.905, yet it included most parameters out of all the models.
- Using the pairs topology for entanglement caused reduced performance for the models using CNOT gates and effectively left the models using CZ gates as the best models. Also, increasing the number of layers had a minimal effect with pairs topology.
- All the models were relatively stable as the confidence interval remained close to the mean throughout the evolution process. However, the model using CNOT gates in the chain topology for entanglement with two layers seemed to have some stability issues during the last timesteps, as the confidence interval began to expand during the last steps across the different problem Hamiltonians. Even though this model was one of the best ones based on fidelity, stability could be a concern when training VarQBM.

For completeness, we highlight all the metrics of the model having CNOT gates in the chain topology for entanglement with three layers in Figure 4.4, as it seemed to perform the best out of all the models. Furthermore, we show an example thermal state using a heatmap in Figure 4.5 prepared by VarQITE with the particular model.

4.1.2 Preparing thermal states of Heisenberg model

In addition to applying variational quantum imaginary time evolution on preparing the thermal states of the transverse field Ising model, we experimented with another well-known model, that is, the Heisenberg model used for studying magnetic systems. Particularly, we used the XYZ extension, which we consider precisely as

$$\hat{H} = \sum_{\langle i,j \rangle} J_x \hat{\sigma}_i^x \hat{\sigma}_j^x + J_y \hat{\sigma}_i^y \hat{\sigma}_j^y + J_z \hat{\sigma}_i^z \hat{\sigma}_j^z + \sum_i h \hat{\sigma}_i^z. \quad (4.2)$$

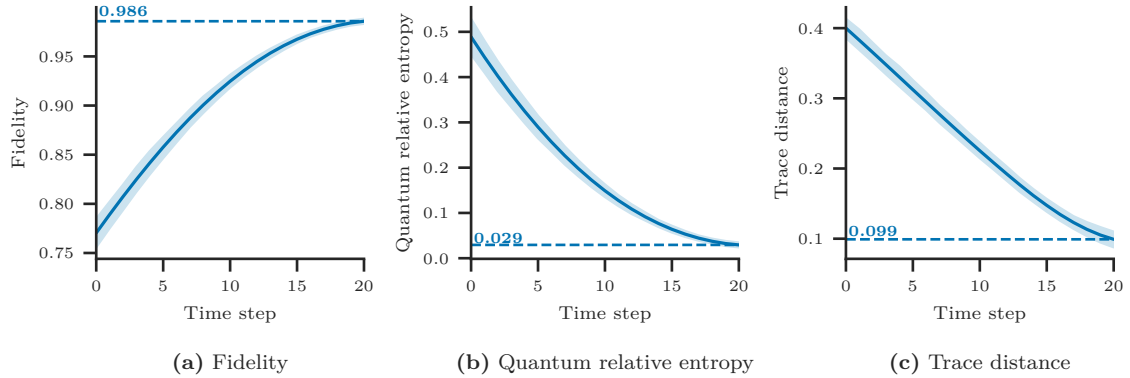


Figure 4.4: (a) fidelity, (b) quantum relative entropy and (c) trace distance against the exact thermal states of the transverse field Ising models through VarQITE with a quantum circuit having three parameterized layers with CNOT gates in the chain topology for entanglement. Furthermore, the solid coloured lines represent the means across all the experiment runs, and the shaded areas represent the 90% confidence intervals. Additionally, the last values of the means are highlighted with horizontal lines. Examining all the metrics confirms the convergence and the successful VarQITE runs for the model.

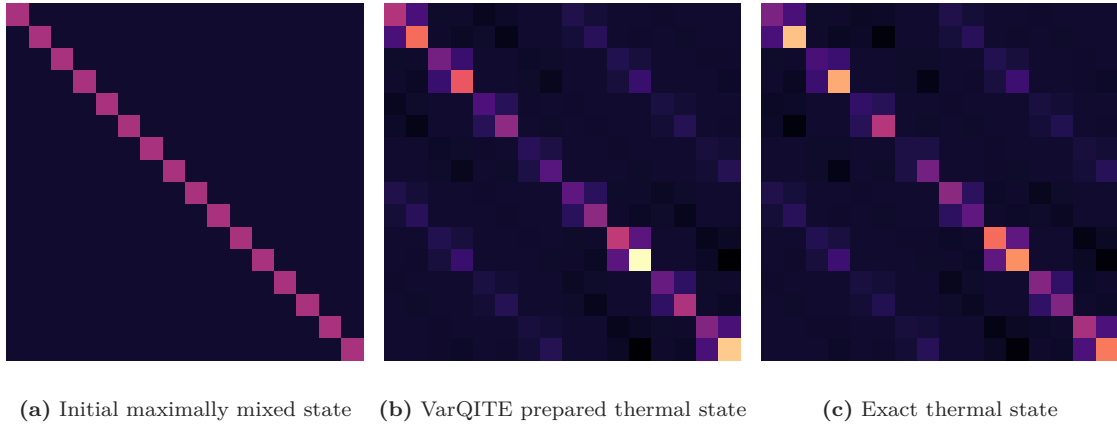


Figure 4.5: The VarQITE process visualized using an example from a quantum circuit having three parameterized layers with CNOT -gates in the chain topology for entanglement. (a) the initial maximally mixed state from which VarQITE begins, and (b) the result of VarQITE with the particular circuit after the evolution. Furthermore, as a comparison, (c) the exact solution calculated by solving the matrix exponential. Particularly, for this example, the fidelity between (b) and (c) is 0.973.

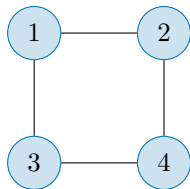


Figure 4.6: The square (2D lattice) connectivity was used to experiment with the thermal state preparation of the Heisenberg model.

In the XYZ variant, the coefficients are free from each other so that $J_x \neq J_y \neq J_z$, while in the XXX variant the coefficients would be as $J_x = J_y = J_z$, or in the XXZ as $J_x = J_y \neq J_z$ [19]. Furthermore, we defined the interaction on a square (2D lattice) of four qubits illustrated in Figure 4.6. Similarly, as we did for the experiments with the Ising model, for each experiment run, we sampled the coefficients from uniform distribution $J_x, J_y, J_z \sim \mathcal{U}(-1, 1)$ (half-open interval) to acquire generalized results.

The results of preparing thermal states of the Heisenberg (XYZ) models with VarQITE using various model setups are summarized in Figure 4.7 with a grid of plots showing the fidelity through the evolution similarly as we previously showed for the Ising model in Figure 4.3. From the plots, we again extracted the following results and connected some of the results with the previous notions:

- Generally, the Heisenberg model turned out to be more complicated than the Ising model as problem Hamiltonian, as the mean fidelity values in the last timesteps were generally worse than in the Ising model across the models. The lowest mean value on the last timestep was 0.837, while the highest was 0.932. Furthermore, there seemed to be much more deviation or instabilities as the confidence intervals were also wider.
- Models using pairs as the entanglement topology seemed to perform generally poorly. Especially, the models using CZ or YY-ZZ as the entangling gates performed unsuccessfully, as the mean fidelity values were only reaching 0.837 in the last timesteps.
- The stability issues which we noted earlier for the model using CNOT gates in the chain topology for entanglement with two or three layers were more evident in the Heisenberg model than in the Ising model, as in the Heisenberg model, even the mean line collapses before reaching the last timesteps.
- Similarly to the Ising model, increasing the layer count provided again only substantial advantage, with few exceptions being again the models using CNOT as

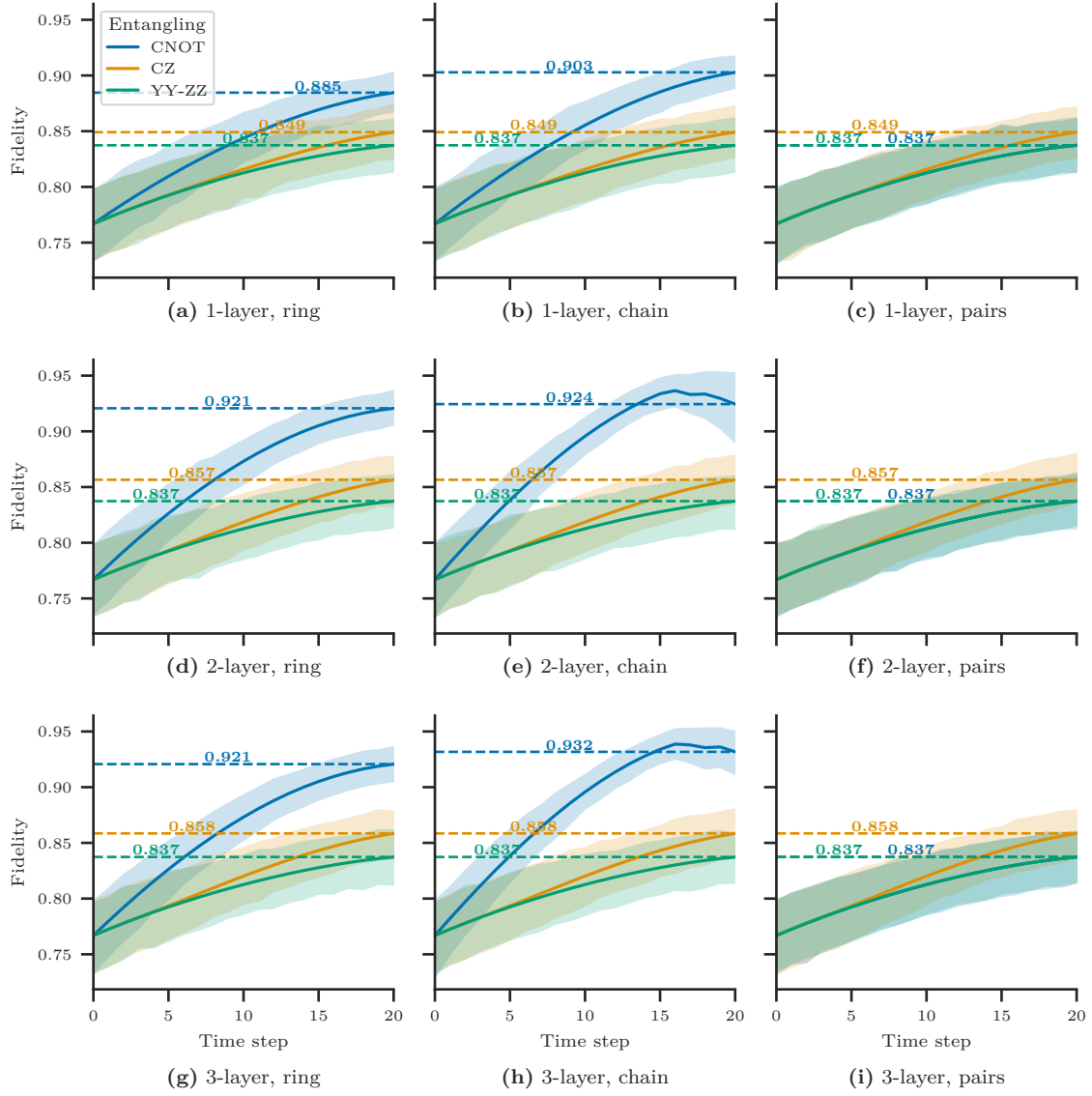


Figure 4.7: Fidelity against the exact thermal states of the Heisenberg (XYZ) models through VarQITE across all the different circuit setups that we experimented with. The number of layers is varied on the rows (1, 2 and 3), while entanglement topology is varied on the columns (ring, chain and pairs). Finally, the three different coloured lines represent the different gates used for entanglement (CNOT, CZ and YY+ZZ). Furthermore, the solid coloured lines represent the means across all the experiment runs, and the shaded areas represent the 90% confidence intervals. Additionally, the last values of the means are highlighted with horizontal lines. The figures indicate that models using CNOT gates for entanglement in chain topology continued to perform the best. Specifically, the three-layer variant shown in Figure (h) was again the best model out of all. Compared to the transverse field Ising model, Heisenberg (XYZ) model turned out to be more difficult as the fidelities are overall smaller than in Figure 4.3.

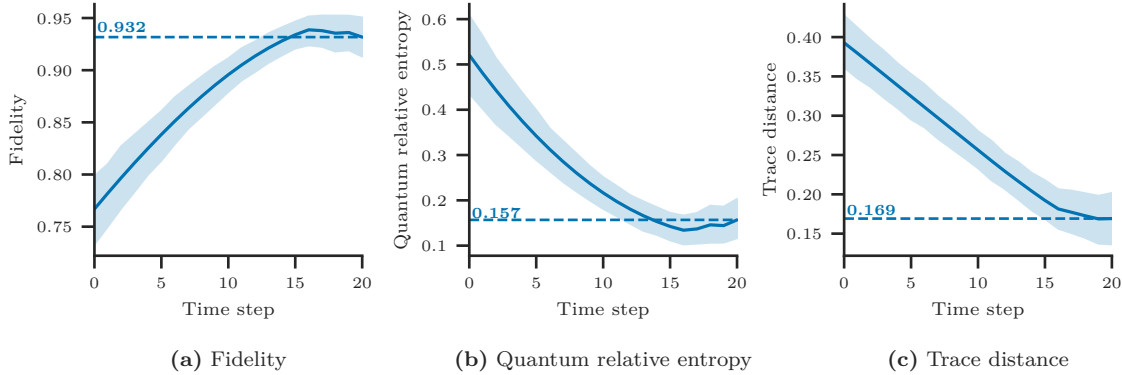


Figure 4.8: (a) fidelity, (b) quantum relative entropy and (c) trace distance against the exact thermal states of the Heisenberg (XYZ) models through VarQITE with a quantum circuit having three parameterized layers with CNOT gates in the chain topology for entanglement. Furthermore, the solid coloured lines represent the means across all the experiment runs, and the shaded areas represent the 90% confidence intervals. Additionally, the last values of the means are highlighted with horizontal lines. Examining all the metrics confirms the convergence and the moderately successful VarQITE runs for the model.

the entangling gates. Furthermore, some minor increases were also visible with the models using CZ as the entangling gates when layer count was increased.

We again, for completeness, highlight all the metrics of the model having CNOT gates in the chain topology for entanglement with three layers in Figure 4.8, as it seemed to perform the best out of all the models for the Heisenberg model.

4.2 Efficient approximation using stochasticity

As noted earlier in the literature review, VarQITE requires evaluating the quantum Fisher information (QFI) matrix (matrix A in Equation 2.55), which is a computationally expensive task scaling at least quadratically with respect to circuit parameters [22]. A recent proposal [22], experimented with simultaneous perturbation stochastic approximation (SPSA) [61] for speeding up the evaluation of QFI matrix. We decided to employ similar kind of experiments for our setup if we could have achieved speedups from the stochastic approximation.

Mainly, we experimented with the number of SPSA approximations for the QFI matrix to average over (50, 200 and 800), the number of timesteps to divide the evolution time in VarQITE (10, 20 and 40) and the amount of perturbation used in SPSA ($\frac{\pi}{20}$, $\frac{\pi}{40}$ and $\frac{\pi}{80}$). The results of employing SPSA for evaluating the QFI in VarQITE are summarized

in Figure 4.9 with a grid of plots showing the fidelity against the exact state (solution) through the evolution. Particularly, the grid is constructed so that the rows represent the number of timesteps to which the evolution was split, while the different columns represent the different perturbation amounts used. Finally, the different coloured lines showcase the different amounts of SPSA approximations for the QFI matrix. In contrast to the previous plots, we highlight the peak values rather than the last values with horizontal lines. Furthermore, the “None” variant plotted with a dashed line is the reference for the model that was not using the stochastic approximation.

The results on employing SPSA were well aligned with the expectations. Notably, using more SPSA resamplings for building the approximations of the QFIs provided more precise results, and higher fidelities were achieved at the end of the evolution. Furthermore, more timesteps yielded generally more precise evolution, and hence higher fidelities were achieved. Then again, using larger values for perturbation caused the evaluation to peak before the end of the evolution, even though the perturbation amount should have normalized it. This early peaking caused the evolution to fail and likely continue to the next excited state yielding bad results at the end of VarQITE. For completeness, we highlight all the metrics of the SPSA setup using 40 timesteps, 800 SPSA resamplings for QFI approximation and minimal perturbation of $\frac{\pi}{80}$ in Figure 4.10, as it seemed to perform the best out of all the setups.

4.3 Training quantum Boltzmann machine

Having done thorough experimentation with the VarQITE procedure, we proceeded with experimenting with VarQBM. As already noted, our goal was to do generative training with VarQBM using the blocks and stripes (BAS) dataset illustrated in Figure 4.11. Particularly, the BAS dataset represents a state with sampling probabilities as

$$p_{\text{BAS}} = \left[\frac{1}{6}, 0, 0, \frac{1}{6}, 0, \frac{1}{6}, 0, 0, 0, 0, \frac{1}{6}, 0, \frac{1}{6}, 0, 0, \frac{1}{6} \right]. \quad (4.3)$$

This probability distribution is multimodal as there is a total of 6 peaks (out of 16 possible values) with probabilities of $\frac{1}{6}$ for each.

After conducting the experiments on VarQITE, we already noticed that the runtime would be a bottleneck for experimenting VarQBM through the enormous hyperparameter space. Hence, we decided to use the best performing model variants from the VarQITE experiments, which were the models using CNOT gates in the chain topology, and only vary the

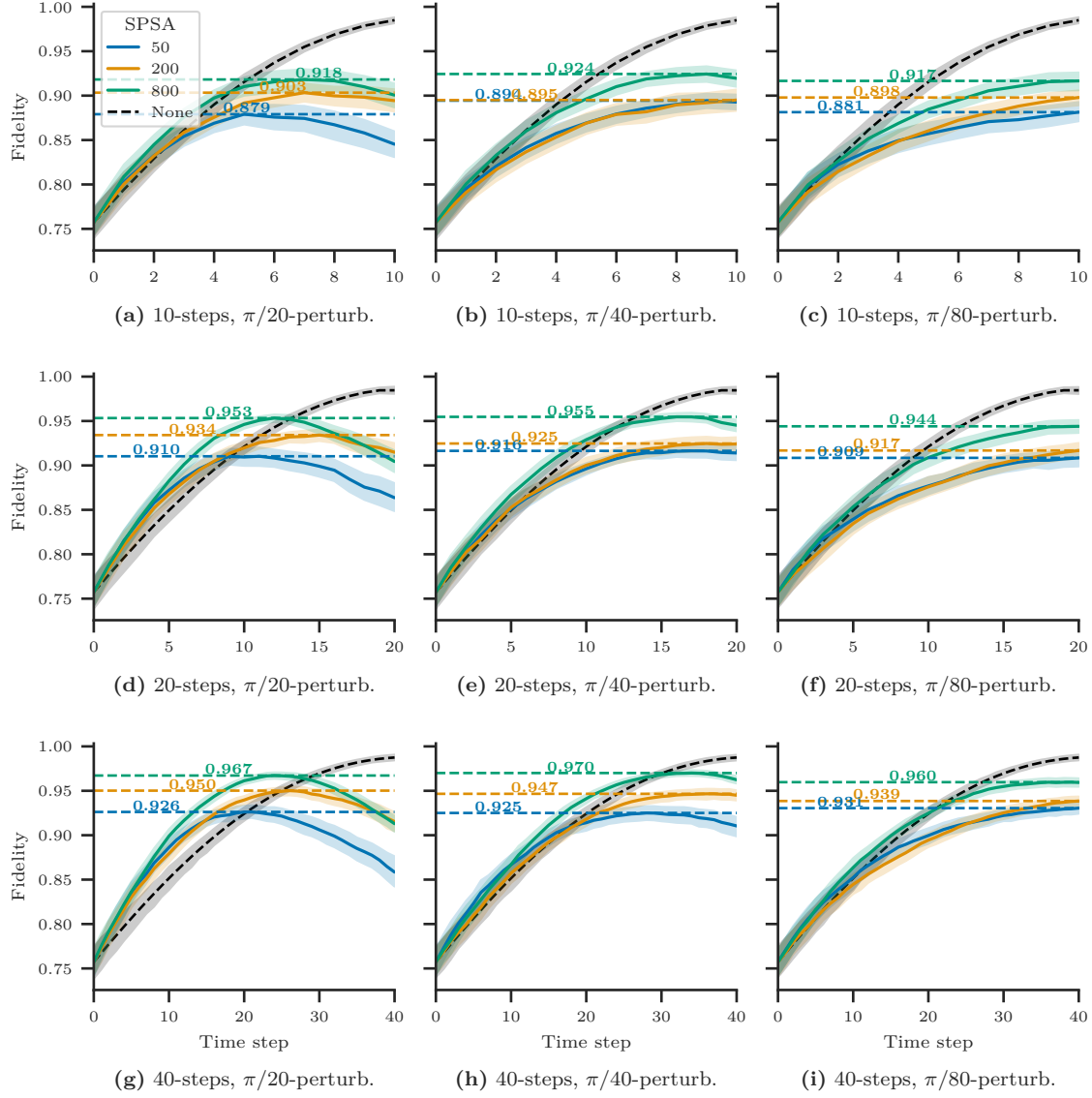


Figure 4.9: Fidelity against the exact thermal state through VarQITE across all the different SPSA hyperparameter setups that we experimented with. The amount of timesteps is varied on the rows (10, 20 and 40), while the perturbation amount is varied on the columns ($\frac{\pi}{20}$, $\frac{\pi}{40}$ and $\frac{\pi}{80}$). Finally, the three different coloured lines represent the different amounts of SPSA approximations for the QFI matrix (50, 200 and 800), and the black dashed line is a reference for comparing the same model without using SPSA. Furthermore, the solid coloured lines represent the means across all the experiment runs, and the shaded areas represent the 90% confidence intervals. Additionally, the peak values of the means are highlighted with horizontal lines. The figure validates that increasing the number of SPSA approximations yields better results.

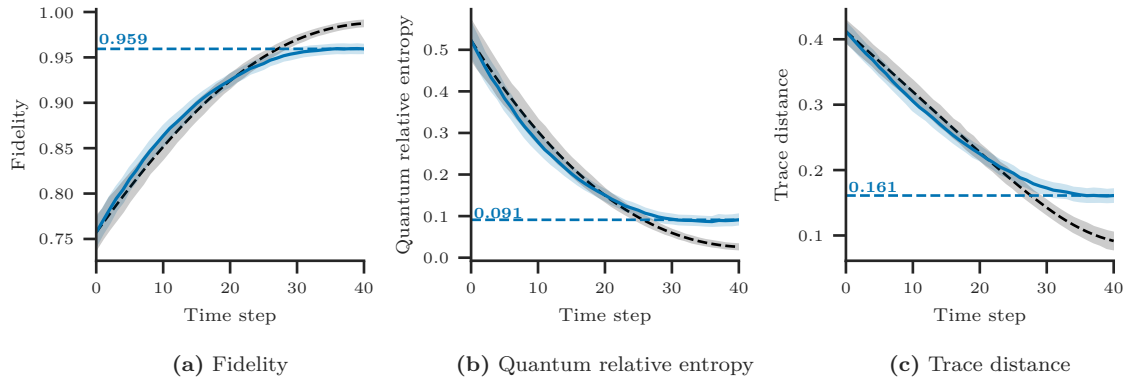


Figure 4.10: (a) fidelity, (b) quantum relative entropy and (c) trace distance against the exact thermal states of the transverse field Ising models through VarQITE with 40 timesteps and SPSA with perturbation of $\frac{\pi}{80}$ and 800 SPSA resamplings for QFI approximation. The black dashed line is a reference for comparing the same model without using SPSA. Furthermore, the solid coloured lines represent the means across all the experiment runs, and the shaded areas represent the 90% confidence intervals. Additionally, the last values of the means are highlighted with horizontal lines. Examining all the metrics confirms the model convergence with SPSA. However, the results with SPSA are not as good as without SPSA, which is to be expected.

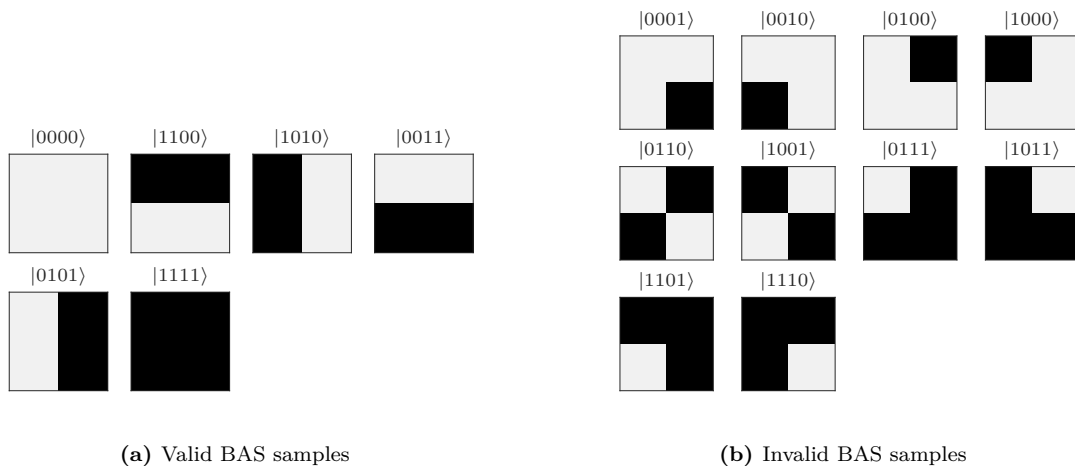


Figure 4.11: (a) valid values of blocks and stripes (BAS) dataset (two by two), and (b) invalid values of BAS dataset. The goal of generative training of the BAS dataset is to train the model to sample the valid values.

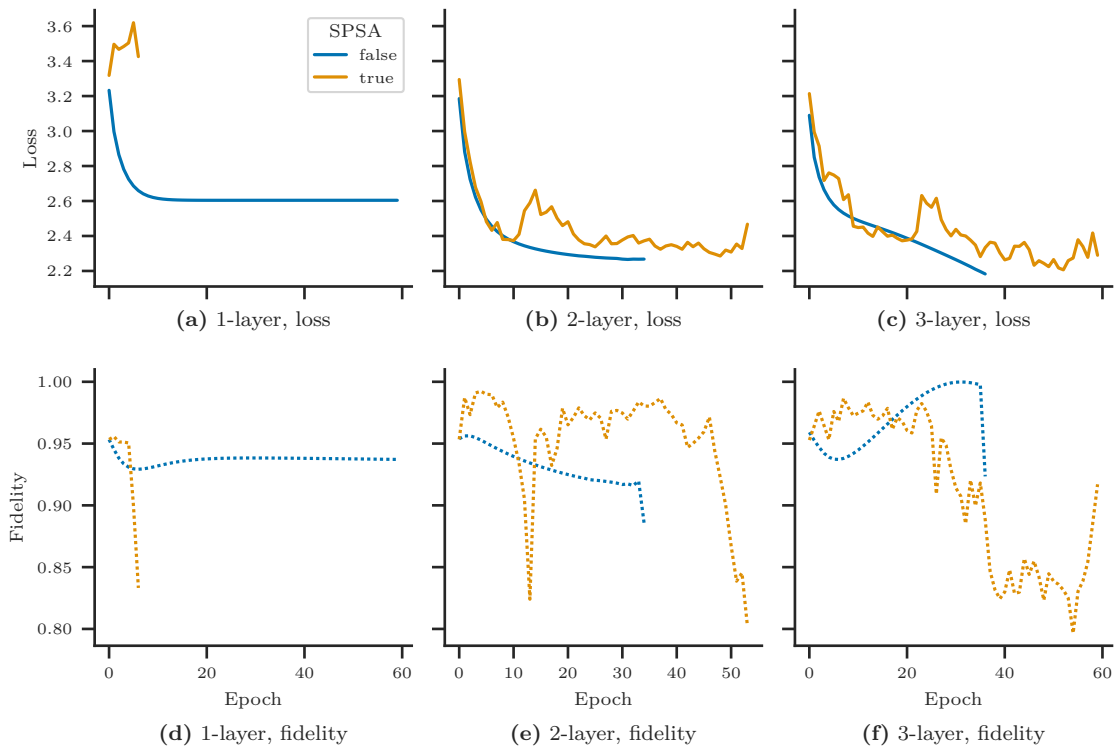


Figure 4.12: The cross-entropy loss between the probability distribution represented by VarQBM’s state and the probability distribution represented by the BAS dataset with (a) the 1-layer variational circuit, (b) the 2-layer variational circuit and (c) the 3-layer variational circuit. Below the loss, the mean of fidelities against the exact thermal states in the last timesteps of VarQITE. Particularly, the fidelities are for the same VarQBM runs with (d) the 1-layer variational circuit, (e) the 2-layer variational circuit and (f) the 3-layer variational circuit. Furthermore, different colours show whether SPSA was used for VarQITE. None of the models performed considerably well, given that a perfect model would get a loss of approximately 1.79. Nevertheless, convergence is visible, especially with models having two or three layers in the quantum circuit. Particularly, the model using the three-layer variational circuit was converging very well, but the sudden collapse in VarQITE terminated the training after epoch 37.

number of layers. Furthermore, we decided to try with and without employing SPSA for VarQITE. Particularly for the SPSA variants, we used the best performing variant having $\frac{\pi}{80}$ perturbation and 800 resamplings for QFI. After conducting a few initial rounds of experimenting using SPSA, we quickly confirmed that utilizing SPSA would require additional averaging for thermal state preparation to be stable enough. The same condition was also noted in the numerical tests of the original study [22]. Similarly, as in [22], we decide to use ten resamplings for the thermal state.

The results on training VarQBM against the BAS dataset using the described parameters are summarized in Figure 4.12. From the particular figure, we extracted the following

results and enhanced some of the previous notions:

- None of the models ended up being particularly successful in terms of loss, given that a perfect model would get a cross-entropy loss of approximately 1.79. Nevertheless, the model using a three-layer variational circuit ended up being the best, with the final loss of 2.18. Based on the curvature, we predict that this model would have achieved better results, given that we could have accounted for the sudden collapse in VarQITE (shown in the fidelity).
- All of the models eventually failed, as VarQITE suddenly produced poor approximations of the thermal states (collapsing). As we were using the finite-difference method, this failure most often meant that the gradient exploded in some direction, causing increasingly high-magnitude Hamiltonian parameters.
- Related to the previous point, we noticed that the higher the magnitude was for the Hamiltonian parameters, the bigger the chance was for failures in VarQITE. This fragility was troublesome because it is natural for the training process that the magnitude of specific parameters increases quite large, and that should be manageable. Outside the primary experiments, we verified that this fragility could be accounted for by increasing the number of timesteps used for the VarQITE.
- SPSA was able to provide reasonably good approximations for models having a two- or three-layer circuit. However, we note that we had to use a considerable amount of resamplings for both the QFI and thermal state to account for the precision, so the actual benefit of using SPSA may be substantial. For the model having a one-layer variational circuit, the experiment was inconclusive with SPSA.
- The one-layer model without SPSA could converge very stably, but it lacked representation capabilities and was left relatively high in terms of loss.

For the sake of completeness, we show the visualization between the exact probability distribution that the BAS dataset represents and the probability distribution that the state of the best performing VarQBM represents in Figure 4.13. Furthermore, to enable comparison against less multimodal probability distributions, we show the final results of training VarQBM against simpler (less multimodal) probability distributions in Figure 4.14. Those are a simpler BAS dataset with no blank or fully filled samples and the probability distribution represented by the 4-qubit GHZ state.

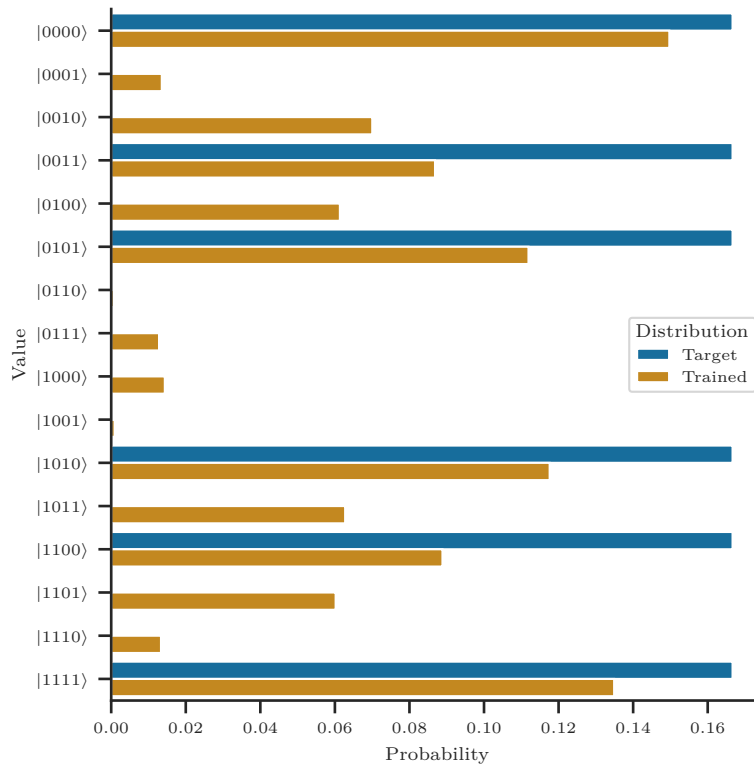


Figure 4.13: The final probability distribution represented by the state of the VarQBM model compared to the probability distribution represented by the BAS dataset. Particular VarQBM model has the three-layer variational quantum circuit and was trained without the usage of SPSA for VarQITE. The comparison shows that the model could not capture the probability distribution thoroughly and is rather noisy. However, considering only the (six) peak values of valid BAS samples, the model was able to capture them correctly as they are also the peaks of the probability distribution represented by the state of the VarQBM.

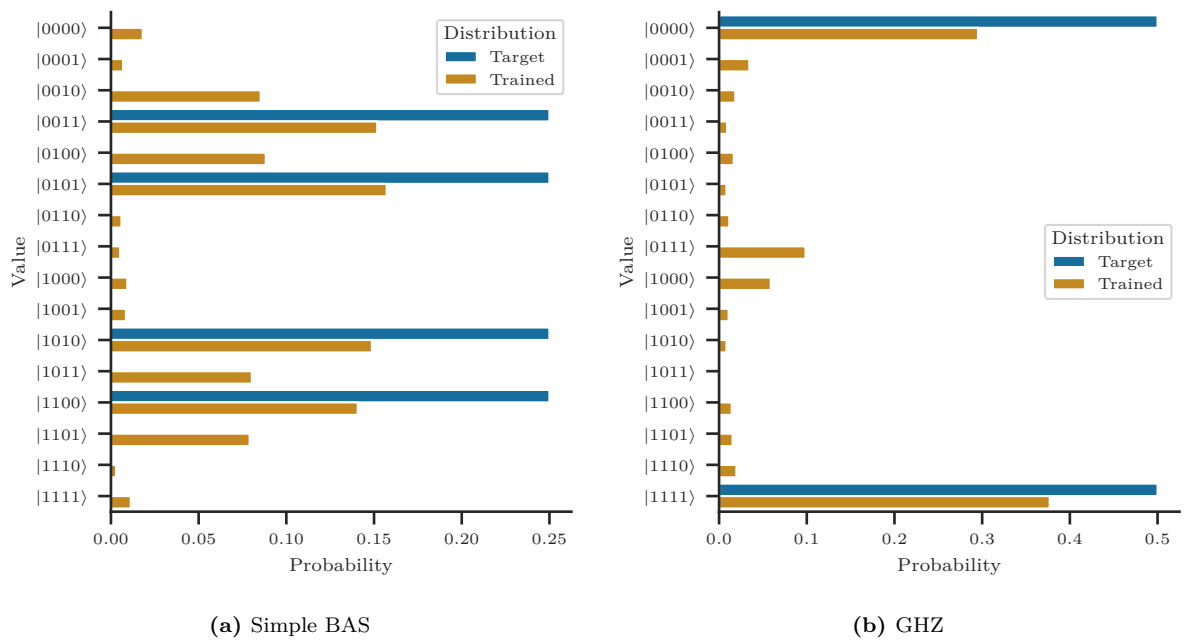


Figure 4.14: The data distribution and the distribution represented by VarQBM’s state after training the model against (a) the simple BAS dataset where blank and completely filled samples are considered invalid, and (b) the probability distribution represented by the 4-qubit GHZ state. Particular VarQBM model has the three-layer variational quantum circuit and was trained without the usage of SPSA for VarQITE. The comparisons show that the fewer there are peaks in the data distribution (less multimodal), the better VarQBM can capture those and learn them.

The comparisons in Figure 4.13 and Figure 4.14 confirm that multimodal datasets are more challenging to learn for all-visible VarQBMs. Particularly, training against the probability distribution of the 4-qubit GHZ state (2 peaks) performs rather successfully with few minimal outliers in Figure 4.14b. In contrast, training against the BAS datasets, VarQBM performed less successfully and there are clear outliers in Figure 4.13 and Figure 4.14a. Especially, the default BAS dataset with six valid values was more complicated than the simplified BAS dataset with only four valid values.

5 Discussion

In this chapter, we will primarily discuss the results of our experiments and highlight the findings of our study. Furthermore, we will reflect the results against the original VarQBM study [73], which was the primary method that we experimented with for the QBM. Moreover, we will highlight any limitations that we faced during the study and discuss future directions to which this work could be extended.

5.1 Analysis of the results

Considering the results of thermal state preparation in Section 4.1, we were able to confirm a set of factors that affect the performance of VarQITE. Those were the number of layers in the variational circuit, the entangling gates and the two-qubit gate placement topologies. However, within the variation that we experimented with, some of the results came as a surprise. For instance, increasing the number of layers did not always benefit, or the benefit was only substantial. Furthermore, the models having YY-ZZ gates as two-qubit gates performed the same across all the other variations.

The difference in performance between the transverse field Ising model and Heisenberg (XYZ) model also came somewhat as a surprise. Specifically, the Heisenberg model turned out to be much more difficult based on the measures we collected. We suspect that the additional two-qubit operators in the Hamiltonian made it more complex even though the parameters were partly shared between the operators.

Overall, we concluded that the models having CNOT gates in the chain topology worked out the best. Furthermore, particular models did benefit from increasing the number of layers, while the benefits were rather substantial when increasing the number of layers from two to three. With the two-layer model having CNOT gates in the chain topology, we also noted a sudden drop or collapse during the last timesteps of VarQITE. We could not fully understand the reason behind this collapsing. However, we suspect that it is related to the limited number of timesteps used for evolution or that VarQITE was overshooting the evolution. We also noticed similar behaviour occasionally with other models. To rule out the possibility of exceeding the thermal state in the evolution, we decided to tweak the parameter update rule as there would be an extra timestep that is not run at all. This

way, we could minimize the possibility that the evolution would go beyond the thermal state.

The failures in VarQITE turned out to be fatal in the VarQBM training process, particularly when evaluating the gradient for the Hamiltonian parameters. As we were using the finite difference methods to evaluate the gradient, if any of the finite difference runs failed, it often caused the gradient to explode in some direction. Furthermore, exploding gradients potentially lead to numerical instabilities and completely failing the training process. We also noticed in the VarQBM training that as the training proceeded, Hamiltonian parameters' magnitude tended to increase. Furthermore, as the magnitude of the parameters increased, we saw more failures in VarQITE and generally worse results in our measures, which indicated that the thermal state preparation became difficult. By increasing the number of timesteps, we were able to overcome this challenge to some extent.

We did not find that this kind of failures in VarQITE would have been noted in the original VarQBM study [73]. Also, the studies focusing on VarQITE [46, 72] focused primarily on ground state preparation, so we did not have much to compare against our experiments of thermal state preparation.

For the results of stochastic approximation experiments in Section 4.2, we were able to acquire results somewhat similar to those reported in the original study [22]. However, one unusual thing which we could not explain was noticed by experimenting with the perturbation amount in SPSA. By increasing the perturbation amount, VarQITE overshot the evolution and reached the thermal state prematurely. While this initially seemed like a defect in implementation and some normalization against perturbation amount would be missing, we could not find such deficiency in the implementation. Thus this finding remained unresolved. Furthermore, the conclusion was that decreasing the perturbation amount made the evolution match more towards the VarQITE evolution without SPSA. Additionally, we noticed that increasing the number of resamplings yield better approximations and better final measures, as is to be expected.

In the VarQBM training results, in Section 4.3, the obvious observation was that the dataset of blocks and stripes (BAS) turned out to be more difficult than the Bell states and the GHZ states experimented in the original studies [22, 73]. Hence, we could not achieve results as good as we hoped for and were in previous work. We see that the multimodality of the BAS dataset made it more challenging to learn. To be more precise, the GHZ states and the Bell states are bimodal distributions having two distinctive peaks with probabilities of 0.5. Meanwhile, the BAS dataset (two by two) has six valid values,

giving a probability of $\frac{1}{6}$ for each of the valid values (or four valid values with $\frac{1}{4}$ probability for each, if blank and filled are not considered to be valid). We think these smaller and less emphasized peaks in the probability distribution made it a more difficult task for VarQBM to learn. Furthermore, we think that the noise introduced by NISQ hardware could make the learning much more difficult or even impossible if the peaks would end up lost in the noise of the hardware. We already see some hints of such behaviour once we applied SPSA, which naturally introduces noise to the optimization process.

While we noticed many challenges in learning the BAS dataset, especially with SPSA and simpler models, we could still achieve relatively successful runs. Using enough timesteps for evolution, we were able to generate precise approximations of thermal states without the fear of failures. Furthermore, the three-layer model having CNOTs in chain topology for entanglement turned out to have good representation capabilities for VarQBM. Potentially by increasing the number of timesteps and designing a more powerful variational circuit, even better results could be achieved. However, as highlighted, the simulation times tend to scale with more timesteps and more powerful models.

5.2 Limitations of the study

After conducting the experiments, we noticed two significant limitations. Firstly, the size of hyperparameter space was already large for VarQITE and became even larger for VarQBM. Therefore, we were forced to limit our experiments only to a set of hyperparameters. Those were the number of layers on the quantum circuit model and usage of SPSA as a stochastic approximation. Furthermore, in the VarQBM experiments, we decided to use only the VarQITE setups, which performed the best. However, we see that exploring more combinations in the hyperparameter space would be relevant to study for analyzing the performance of VarQBM. Also, the VarQBM models which we experimented with were fully visible with no hidden units. Hence, it would be crucial to determine which kind of effects the hidden units introduce to the VarQBM model.

The second significant limitation was related to the runtime of the model training. Simulating a model with eight qubits and especially with many layers quickly took hours. For instance, for VarQITE, we experimented with 54 different model setups, each for 40 runs, yielding a total of 2160 VarQITE experiments without even considering the SPSA experiments. Moreover, simulating a single VarQITE run took some minutes quickly, especially on models with many layers. Furthermore, for training a single VarQBM, VarQITE had

to be run several times during the training process, which quickly increased the training time to some hours. Spending a tremendous amount of time on a single model made it impossible for us to repeat the experiments with different random initial parameter setups. Thus the generalisability is potentially compromised if the results are not repeatable with different initial parameter assignments. The same is also one reason for the first point and why we could try only a limited set of models in the large hyperparameter space.

5.3 Future work

As already mentioned in previous sections, many directions for future investigation rose during the experiments. Firstly, we see that trying out more different kinds of variational circuits would be an essential factor in fine-tuning VarQITE and VarQBM. For instance, the different models experimented in [59] could be tried in the context of VarQBM. Furthermore, it should be considered together by introducing the hidden (latent) state to the model to analyze how the representation capability of the model changes. Also, we used only a vanilla gradient descent with no decaying or other adjustments, so trying out different optimizers (both gradient and gradient-free) is a direction that should be examined in future, especially with larger datasets.

Secondly, as the hardware in NISQ era is rather noisy, it would be crucial to try out the VarQBM training process and apply a noise model to it. Especially with multimodal distributions where the probability distributions' peaks are not so distinctive, it could very well be so that the model fails to learn such distributions as the peaks could end up lost into the noise. We already saw some hints on this with models having only a single layer and using the noisy SPSA for approximation.

Considering only VarQITE for thermal state preparation, we also noted few future directions to consider. First of all, as we noticed, the thermal state preparation got difficult as the magnitude of the Hamiltonian parameters ended up larger during the training process. However, this problem was somewhat mitigated by splitting the evolution into more timesteps. A future study could consider this relation more carefully and especially identify how to minimize the number of time steps required for a successful VarQITE run on thermal state preparation with sufficient precision. Especially, this should be considered with Hamiltonians having high-magnitude coefficients as those tended to be more difficult. Furthermore, it would be ideal to recognize when VarQITE is failing roughly. As we noticed, in some cases, the evolution was collapsing likely to some other exciting state,

thus yielding an incorrect approximation for the thermal state. It would be vital to identify such failing VarQITE runs and drop them out of the VarQBM training process, as they could potentially lead to exploding gradients or other fatal issues.

6 Conclusion

Quantum computing has plenty of use cases across the industries and is particularly suitable for problems that are intractable for classical computation. Especially in machine learning, specific models have suffered from exponentially scaling computational cost and have settled mostly for approximating methods. Boltzmann machines belong to the family of energy-based graphical models, which are well known to suffer from exponentially scaling computation. Generalizing Boltzmann machines to quantum computing has already been investigated rather extensively with various studies.

In this thesis, we examined quantum Boltzmann machines (QBMs) and their training routines through a literature review and experimented with variational quantum Boltzmann machine (VarQBM), a variant of QBM. Particular focus was on variational quantum imaginary time evolution (VarQITE), an algorithm in the core of VarQBM, used for thermal state preparation. The experiments were considered with a more extensive set of hyperparameters compared to previous studies [22, 73]. Furthermore, a dataset of blocks and stripes (BAS) was used for training the VarQBM. Compared to the Bell states and the Greenberger-Horne-Zeilinger (GHZ) states used in earlier studies, the BAS dataset is more multimodal and turned up being a difficult task for VarQBM.

We observe that the multimodality of data makes the probability distribution peaks less distinctive, hence harder to learn. Furthermore, by combining the noise of the near-term quantum hardware, these peaks of the distribution could potentially end up completely lost in the noise. Nevertheless, we see that including the hidden (latent) state could be a crucial element in learning the BAS dataset, given the patterns and structures that the latent state could potentially capture.

Before conducting the experiments with VarQBM, we examined profoundly how the performance of VarQITE can be altered by adjusting various hyperparameters and potentially speed up with a stochastic approximation. Especially different circuit topologies and gate combinations were examined. We confirmed that variational quantum circuits having CNOT gates in the chain topology perform well in preparing the thermal states of different Hamiltonians. However, we noticed that as the magnitude of the Hamiltonians' parameters increases, VarQITE could become fragile, and the precision suffers. The fragility could be mitigated by splitting the evolution into more timesteps, causing an

increase in the overall runtime. Nevertheless, poorly performing VarQITE has fatal effects on VarQBM, especially if it leads to exploding gradients in training.

As for applying the stochastic approximations for VarQITE using simultaneous perturbation stochastic approximation (SPSA), we were able to verify similar results as in the original study [22]. However, the benefits of using SPSA for approximation in VarQITE may be substantial if it requires an increasing amount of resamplings to produce precise enough approximations.

Generally, we conclude VarQBM to be a particularly appealing model for NISQ era with much potential. However, more studies are required for revealing its true power with more significant and more complex problems. Especially, optimizing the VarQITE routine through the whole training process is an essential factor when scaling the modal to real-world tasks.

Whether it is VarQBM or some other QBM model variant, QBMs continue to be a promising model in quantum machine learning. However, it is up to the future to show what kind of innovations there can be in quantum computing and quantum machine learning. Especially, realizing more stable hardware with more qubits and concepts like quantum RAM should enable further studies with more significant problems.

Bibliography

- [1] M. Abadi et al. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. Software available from tensorflow.org. 2015. URL: <https://www.tensorflow.org/>.
- [2] D. H. Ackley, G. E. Hinton, and T. J. Sejnowski. “A learning algorithm for Boltzmann machines”. In: *Cognitive Science* 9.1 (1985), pp. 147–169. ISSN: 0364-0213. DOI: [10.1016/S0364-0213\(85\)80012-4](https://doi.org/10.1016/S0364-0213(85)80012-4).
- [3] S. H. Adachi and M. P. Henderson. *Application of Quantum Annealing to Training of Deep Neural Networks*. 2015. arXiv: [1510.06356 \[quant-ph\]](https://arxiv.org/abs/1510.06356).
- [4] D. Aharonov, W. van Dam, J. Kempe, Z. Landau, S. Lloyd, and O. Regev. “Adiabatic Quantum Computation is Equivalent to Standard Quantum Computation”. In: *SIAM Journal on Computing* 37.1 (2007), pp. 166–194. DOI: [10.1137/S0097539705447323](https://doi.org/10.1137/S0097539705447323).
- [5] E. Aïmeur, G. Brassard, and S. Gambs. “Machine Learning in a Quantum World”. In: *Advances in Artificial Intelligence*. Ed. by L. Lamontagne and M. Marchand. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 431–442. ISBN: 978-3-540-34630-2. DOI: [10.1007/11766247_37](https://doi.org/10.1007/11766247_37).
- [6] T. Albash and D. A. Lidar. “Adiabatic quantum computation”. In: *Rev. Mod. Phys.* 90 (1 Jan. 2018), p. 015002. DOI: [10.1103/RevModPhys.90.015002](https://doi.org/10.1103/RevModPhys.90.015002).
- [7] M. H. Amin, E. Andriyash, J. Rolfe, B. Kulchytskyy, and R. Melko. “Quantum Boltzmann Machine”. In: *Phys. Rev. X* 8 (2 May 2018), p. 021050. DOI: [10.1103/PhysRevX.8.021050](https://doi.org/10.1103/PhysRevX.8.021050).
- [8] R. J. Baxter. *Exactly solved models in statistical mechanics*. eng. London: Academic Press, 1982. ISBN: 0-12-083180-5.
- [9] M. Benedetti, J. Realpe-Gómez, R. Biswas, and A. Perdomo-Ortiz. “Estimation of effective temperatures in quantum annealers for sampling applications: A case study with possible applications in deep learning”. In: *Phys. Rev. A* 94 (2 Aug. 2016), p. 022308. DOI: [10.1103/PhysRevA.94.022308](https://doi.org/10.1103/PhysRevA.94.022308).

- [10] J. Besag. “Spatial Interaction and the Statistical Analysis of Lattice Systems”. In: *Journal of the Royal Statistical Society: Series B (Methodological)* 36.2 (1974), pp. 192–225. DOI: [10.1111/j.2517-6161.1974.tb00999.x](https://doi.org/10.1111/j.2517-6161.1974.tb00999.x).
- [11] J. Biamonte, P. Wittek, N. Pancotti, P. Rebentrost, N. Wiebe, and S. Lloyd. “Quantum machine learning”. In: *Nature* 549.7671 (Sept. 2017), pp. 195–202. ISSN: 1476-4687. DOI: [10.1038/nature23474](https://doi.org/10.1038/nature23474).
- [12] M. Broughton et al. *TensorFlow Quantum: A Software Framework for Quantum Machine Learning*. 2020. arXiv: [2003.02989](https://arxiv.org/abs/2003.02989) [quant-ph].
- [13] N. H. Bshouty and J. C. Jackson. “Learning DNF over the Uniform Distribution Using a Quantum Example Oracle”. In: *SIAM Journal on Computing* 28.3 (1998), pp. 1136–1153. DOI: [10.1137/S0097539795293123](https://doi.org/10.1137/S0097539795293123).
- [14] T. A. Caswell et al. *matplotlib/matplotlib: REL: v3.4.1*. Version v3.4.1. Mar. 2021. DOI: [10.5281/zenodo.4649959](https://doi.org/10.5281/zenodo.4649959).
- [15] Cirq Developers. *Cirq*. Version v0.10.0. Mar. 2021. DOI: [10.5281/zenodo.4586899](https://doi.org/10.5281/zenodo.4586899).
- [16] D-Wave Systems Inc. *D-Wave System Documentation*. 2021. URL: https://docs.dwavesys.com/docs/latest/c_gs_2.html.
- [17] V. S. Denchev, S. Boixo, S. V. Isakov, N. Ding, R. Babbush, V. Smelyanskiy, J. Martinis, and H. Neven. “What is the Computational Value of Finite-Range Tunneling?” In: *Phys. Rev. X* 6 (3 Aug. 2016), p. 031015. DOI: [10.1103/PhysRevX.6.031015](https://doi.org/10.1103/PhysRevX.6.031015).
- [18] J. V. Dillon, I. Langmore, D. Tran, E. Brevdo, S. Vasudevan, D. Moore, B. Patton, A. Alemi, M. Hoffman, and R. A. Saurous. *TensorFlow Distributions*. 2017. arXiv: [1711.10604](https://arxiv.org/abs/1711.10604) [cs.LG].
- [19] L. D. Faddeev. *How algebraic Bethe ansatz works for integrable model*. May 1996. arXiv: [hep-th/9605187](https://arxiv.org/abs/hep-th/9605187).
- [20] E. Farhi, J. Goldstone, S. Gutmann, and M. Sipser. *Quantum Computation by Adiabatic Evolution*. 2000. arXiv: [quant-ph/0001106](https://arxiv.org/abs/quant-ph/0001106).
- [21] A. Finnila, M. Gomez, C. Sebenik, C. Stenson, and J. Doll. “Quantum annealing: A new method for minimizing multidimensional functions”. In: *Chemical Physics Letters* 219.5 (1994), pp. 343–348. ISSN: 0009-2614. DOI: [10.1016/0009-2614\(94\)00117-0](https://doi.org/10.1016/0009-2614(94)00117-0).

- [22] J. Gacon, C. Zoufal, G. Carleo, and S. Woerner. *Simultaneous Perturbation Stochastic Approximation of the Quantum Fisher Information*. 2021. arXiv: [2103.09232](https://arxiv.org/abs/2103.09232) [quant-ph].
- [23] S. Geman and D. Geman. “Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* PAMI-6.6 (Nov. 1984), pp. 721–741. DOI: [10.1109/TPAMI.1984.4767596](https://doi.org/10.1109/TPAMI.1984.4767596).
- [24] V. Giovannetti, S. Lloyd, and L. Maccone. “Quantum Random Access Memory”. In: *Phys. Rev. Lett.* 100 (16 Apr. 2008), p. 160501. DOI: [10.1103/PhysRevLett.100.160501](https://doi.org/10.1103/PhysRevLett.100.160501).
- [25] S. Golden. “Lower Bounds for the Helmholtz Function”. In: *Phys. Rev.* 137 (4B Feb. 1965), B1127–B1128. DOI: [10.1103/PhysRev.137.B1127](https://doi.org/10.1103/PhysRev.137.B1127).
- [26] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016. URL: <http://www.deeplearningbook.org>.
- [27] D. M. Greenberger, M. A. Horne, and A. Zeilinger. “Going Beyond Bell’s Theorem”. In: *Bell’s Theorem, Quantum Theory and Conceptions of the Universe*. Ed. by M. Kafatos. Dordrecht: Springer Netherlands, 1989, pp. 69–72. ISBN: 978-94-017-0849-4. DOI: [10.1007/978-94-017-0849-4_10](https://doi.org/10.1007/978-94-017-0849-4_10).
- [28] C. R. Harris et al. “Array programming with NumPy”. In: *Nature* 585.7825 (Sept. 2020), pp. 357–362. DOI: [10.1038/s41586-020-2649-2](https://doi.org/10.1038/s41586-020-2649-2).
- [29] A. W. Harrow, A. Hassidim, and S. Lloyd. “Quantum Algorithm for Linear Systems of Equations”. In: *Phys. Rev. Lett.* 103 (15 Oct. 2009), p. 150502. DOI: [10.1103/PhysRevLett.103.150502](https://doi.org/10.1103/PhysRevLett.103.150502).
- [30] B. Heim, T. F. Rønnow, S. V. Isakov, and M. Troyer. “Quantum versus classical annealing of Ising spin glasses”. In: *Science* 348.6231 (2015), pp. 215–217. ISSN: 0036-8075. DOI: [10.1126/science.aaa4170](https://doi.org/10.1126/science.aaa4170).
- [31] G. E. Hinton. “Boltzmann machine”. In: *Scholarpedia* 2.5 (2007). revision #91076, p. 1668. DOI: [10.4249/scholarpedia.1668](https://doi.org/10.4249/scholarpedia.1668).
- [32] G. E. Hinton. “Training Products of Experts by Minimizing Contrastive Divergence”. In: *Neural Computation* 14.8 (2002), pp. 1771–1800. DOI: [10.1162/089976602760128018](https://doi.org/10.1162/089976602760128018).

- [33] G. E. Hinton and T. J. Sejnowski. “Optimal perceptual inference”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. Vol. 448. IEEE Computer Society, June 1983, pp. 448–453.
- [34] J. D. Hunter. “Matplotlib: A 2D graphics environment”. In: *Computing in Science & Engineering* 9.3 (2007), pp. 90–95. DOI: [10.1109/MCSE.2007.55](https://doi.org/10.1109/MCSE.2007.55).
- [35] M. I. Jordan and T. M. Mitchell. “Machine learning: Trends, perspectives, and prospects”. In: *Science* 349.6245 (2015), pp. 255–260. ISSN: 0036-8075. DOI: [10.1126/science.aaa8415](https://doi.org/10.1126/science.aaa8415).
- [36] T. Kadowaki and H. Nishimori. “Quantum annealing in the transverse Ising model”. In: *Phys. Rev. E* 58 (5 Nov. 1998), pp. 5355–5363. DOI: [10.1103/PhysRevE.58.5355](https://doi.org/10.1103/PhysRevE.58.5355).
- [37] M. Kieferová and N. Wiebe. “Tomography and generative training with quantum Boltzmann machines”. In: *Phys. Rev. A* 96 (6 Dec. 2017), p. 062327. DOI: [10.1103/PhysRevA.96.062327](https://doi.org/10.1103/PhysRevA.96.062327).
- [38] R. Kindermann and J. L. Snell. *Markov Random Fields and Their Applications*. Vol. 1. Contemporary Mathematics. Providence, Rhode Island: American Mathematical Society, 1980. ISBN: 0-8218-5001-6. DOI: [10.1090/conm/001](https://doi.org/10.1090/conm/001).
- [39] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. “Optimization by Simulated Annealing”. In: *Science* 220.4598 (1983), pp. 671–680. ISSN: 0036-8075. DOI: [10.1126/science.220.4598.671](https://doi.org/10.1126/science.220.4598.671).
- [40] D. Koller and N. Friedman. *Probabilistic graphical models: principles and techniques*. MIT press, 2009.
- [41] S. Kullback and R. A. Leibler. “On Information and Sufficiency”. In: *The Annals of Mathematical Statistics* 22.1 (1951), pp. 79–86. ISSN: 00034851. DOI: [10.1214/aoms/1177729694](https://doi.org/10.1214/aoms/1177729694).
- [42] Y. LeCun, Y. Bengio, and G. E. Hinton. “Deep learning”. In: *Nature* 521.7553 (May 2015), pp. 436–444. ISSN: 1476-4687. DOI: [10.1038/nature14539](https://doi.org/10.1038/nature14539).
- [43] J. Li, X. Yang, X. Peng, and C.-P. Sun. “Hybrid Quantum-Classical Approach to Quantum Optimal Control”. In: *Phys. Rev. Lett.* 118 (15 Apr. 2017), p. 150503. DOI: [10.1103/PhysRevLett.118.150503](https://doi.org/10.1103/PhysRevLett.118.150503).
- [44] A. Mari, T. R. Bromley, and N. Killoran. “Estimating the gradient and higher-order derivatives on quantum hardware”. In: *Phys. Rev. A* 103 (1 Jan. 2021), p. 012405. DOI: [10.1103/PhysRevA.103.012405](https://doi.org/10.1103/PhysRevA.103.012405).

- [45] C. O. Marrero, M. Kieferová, and N. Wiebe. *Entanglement Induced Barren Plateaus*. 2020. arXiv: [2010.15968](https://arxiv.org/abs/2010.15968) [[quant-ph](#)].
- [46] S. McArdle, T. Jones, S. Endo, Y. Li, S. C. Benjamin, and X. Yuan. “Variational ansatz-based quantum simulation of imaginary time evolution”. In: *npj Quantum Information* 5.1 (Sept. 2019), p. 75. ISSN: 2056-6387. DOI: [10.1038/s41534-019-0187-2](https://doi.org/10.1038/s41534-019-0187-2).
- [47] W. McKinney. “Data Structures for Statistical Computing in Python”. In: *Proceedings of the 9th Python in Science Conference*. Ed. by S. van der Walt and J. Millman. 2010, pp. 56–61. DOI: [10.25080/Majora-92bf1922-00a](https://doi.org/10.25080/Majora-92bf1922-00a).
- [48] A. McLachlan. “A variational solution of the time-dependent Schrodinger equation”. In: *Molecular Physics* 8.1 (1964), pp. 39–44. DOI: [10.1080/00268976400100041](https://doi.org/10.1080/00268976400100041).
- [49] K. Mitarai, M. Negoro, M. Kitagawa, and K. Fujii. “Quantum circuit learning”. In: *Phys. Rev. A* 98 (3 Sept. 2018), p. 032309. DOI: [10.1103/PhysRevA.98.032309](https://doi.org/10.1103/PhysRevA.98.032309).
- [50] A. Montanaro. “Quantum algorithms: an overview”. In: *npj Quantum Information* 2.1 (Jan. 2016), p. 15023. ISSN: 2056-6387. DOI: [10.1038/npjqi.2015.23](https://doi.org/10.1038/npjqi.2015.23).
- [51] K. P. Murphy. *Machine learning: a probabilistic perspective*. Cambridge, MA: MIT Press, 2012. ISBN: 9780262018029.
- [52] M. A. Nielsen and I. L. Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press, 2010. ISBN: 978-1107002173.
- [53] J. Preskill. “Quantum Computing in the NISQ era and beyond”. In: *Quantum* 2 (Aug. 2018), p. 79. ISSN: 2521-327X. DOI: [10.22331/q-2018-08-06-79](https://doi.org/10.22331/q-2018-08-06-79).
- [54] P. Rebentrost, M. Mohseni, and S. Lloyd. “Quantum Support Vector Machine for Big Data Classification”. In: *Phys. Rev. Lett.* 113 (13 Sept. 2014), p. 130503. DOI: [10.1103/PhysRevLett.113.130503](https://doi.org/10.1103/PhysRevLett.113.130503).
- [55] T. F. Rønnow, Z. Wang, J. Job, S. Boixo, S. V. Isakov, D. Wecker, J. M. Martinis, D. A. Lidar, and M. Troyer. “Defining and detecting quantum speedup”. In: *Science* 345.6195 (2014), pp. 420–424. ISSN: 0036-8075. DOI: [10.1126/science.1252319](https://doi.org/10.1126/science.1252319).
- [56] R. Salakhutdinov and G. Hinton. “Deep Boltzmann Machines”. In: *Proceedings of the Twelfth International Conference on Artificial Intelligence and Statistics*. Ed. by D. van Dyk and M. Welling. Vol. 5. Proceedings of Machine Learning Research. Hilton Clearwater Beach Resort, Clearwater Beach, Florida USA: PMLR, Apr. 2009, pp. 448–455. URL: <http://proceedings.mlr.press/v5/salakhutdinov09a.html>.

- [57] M. Schuld, V. Bergholm, C. Gogolin, J. Izaac, and N. Killoran. “Evaluating analytic gradients on quantum hardware”. In: *Phys. Rev. A* 99 (3 Mar. 2019), p. 032331. DOI: [10.1103/PhysRevA.99.032331](https://doi.org/10.1103/PhysRevA.99.032331).
- [58] M. Schuld, I. Sinayskiy, and F. Petruccione. “An introduction to quantum machine learning”. In: *Contemporary Physics* 56.2 (2015), pp. 172–185. DOI: [10.1080/00107514.2014.964942](https://doi.org/10.1080/00107514.2014.964942).
- [59] S. Sim, P. D. Johnson, and A. Aspuru-Guzik. “Expressibility and Entangling Capability of Parameterized Quantum Circuits for Hybrid Quantum-Classical Algorithms”. In: *Advanced Quantum Technologies* 2.12 (2019), p. 1900070. DOI: [10.1002/qute.201900070](https://doi.org/10.1002/qute.201900070).
- [60] P. Smolensky. “Information processing in dynamical systems: Foundations of harmony theory”. In: *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Vol. 1: Foundations*. Ed. by D. E. Rumelhart and J. L. McClelland. Cambridge, MA, USA: MIT Press, Jan. 1986, pp. 194–281. ISBN: 0-262-68053-X.
- [61] J. Spall. “Multivariate stochastic approximation using a simultaneous perturbation gradient approximation”. In: *IEEE Transactions on Automatic Control* 37.3 (Mar. 1992), pp. 332–341. DOI: [10.1109/9.119632](https://doi.org/10.1109/9.119632).
- [62] R. B. Stinchcombe. “Ising model in a transverse field. I. Basic theory”. In: *Journal of Physics C: Solid State Physics* 6.15 (Aug. 1973), pp. 2459–2483. DOI: [10.1088/0022-3719/6/15/009](https://doi.org/10.1088/0022-3719/6/15/009).
- [63] E. Tang. “A Quantum-Inspired Classical Algorithm for Recommendation Systems”. In: *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*. STOC 2019. Phoenix, AZ, USA: Association for Computing Machinery, 2019, pp. 217–228. ISBN: 9781450367059. DOI: [10.1145/3313276.3316310](https://doi.org/10.1145/3313276.3316310).
- [64] The pandas development team. *pandas-dev/pandas: Pandas 1.2.3*. Version v1.2.3. Mar. 2021. DOI: [10.5281/zenodo.4572994](https://doi.org/10.5281/zenodo.4572994).
- [65] C. J. Thompson. “Inequality with Applications in Statistical Mechanics”. In: *Journal of Mathematical Physics* 6.11 (1965), pp. 1812–1813. DOI: [10.1063/1.1704727](https://doi.org/10.1063/1.1704727).
- [66] A. Tikhonov, A. Goncharsky, V. Stepanov, and A. G. Yagola. *Numerical Methods for the Solution of Ill-Posed Problems*. Vol. 328. Mathematics and Its Applications. Springer Science & Business Media, 1995. DOI: [10.1007/978-94-015-8480-7](https://doi.org/10.1007/978-94-015-8480-7).

- [67] P. Virtanen et al. “SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python”. In: *Nature Methods* 17 (2020), pp. 261–272. DOI: [10.1038/s41592-019-0686-2](https://doi.org/10.1038/s41592-019-0686-2).
- [68] M. L. Waskom. “seaborn: statistical data visualization”. In: *Journal of Open Source Software* 6.60 (2021), p. 3021. DOI: [10.21105/joss.03021](https://doi.org/10.21105/joss.03021).
- [69] N. Wiebe and L. Wossnig. *Generative training of quantum Boltzmann machines with hidden units*. 2019. arXiv: [1905.09902 \[quant-ph\]](https://arxiv.org/abs/1905.09902).
- [70] R. M. Wilcox. “Exponential Operators and Parameter Differentiation in Quantum Physics”. In: *Journal of Mathematical Physics* 8.4 (1967), pp. 962–982. DOI: [10.1063/1.1705306](https://doi.org/10.1063/1.1705306).
- [71] P. Wittek. *Quantum machine learning: what quantum computing means to data mining*. Academic Press, 2014.
- [72] X. Yuan, S. Endo, Q. Zhao, Y. Li, and S. C. Benjamin. “Theory of variational quantum simulation”. In: *Quantum* 3 (Oct. 2019), p. 191. ISSN: 2521-327X. DOI: [10.22331/q-2019-10-07-191](https://doi.org/10.22331/q-2019-10-07-191).
- [73] C. Zoufal, A. Lucchi, and S. Woerner. “Variational quantum Boltzmann machines”. In: *Quantum Machine Intelligence* 3.1 (Feb. 2021), p. 7. ISSN: 2524-4914. DOI: [10.1007/s42484-020-00033-7](https://doi.org/10.1007/s42484-020-00033-7).

