Thesis for The Degree of Licentiate of Engineering

Learning properties of parametrized quantum states

Practical algorithms with theoretical guarantees

Marc Wanner

Department of Computer Science and Engineering
Chalmers University of Technology | University of Gothenburg
Gothenburg, Sweden, 2026

Learning properties of parametrized quantum states
Practical algorithms with theoretical guarantees

Marc Wanner

ISSN 1652-876X

Department of Computer Science and Engineering
Division of Data Science and AI
Chalmers University of Technology | University of Gothenburg
SE-412 96 Göteborg,
Sweden
Phone: +46(0)31 772 1000

Printed by Chalmers Digitaltryck,
Gothenburg, Sweden 2026.

"Computers, as any programmer will tell you, are giant morons, not giant brains."
- Arthur Samuel

Learning properties of parametrized quantum states
Practical algorithms with theoretical guarantees

Marc Wanner

Department of Computer Science and Engineering
Chalmers University of Technology | University of Gothenburg

Abstract

The analysis of quantum state properties is of fundamental importance in quantum physics and serves as a primary motivation for the development of quantum computing. In particular, low-temperature and ground states are of significant interest, as they most prominently exhibit phenomena unique to quantum mechanics. However, extracting such properties typically requires costly numerical simulations or laboratory experiments, while simulation on quantum hardware remains limited by the capabilities of current devices. This thesis proposes two complementary approaches to address these challenges.

The first contribution presents an algorithm for predicting ground-state properties across an entire family of quantum states within the same phase of matter, based on training data. Under slightly stronger assumptions, we improve upon the sample complexity of existing methods, prove the same rigorous guarantees for a deep neural network-based approach, and demonstrate its practical advantages through extensive numerical experiments.

The second contribution addresses the challenge of preparing quantum states of interest using Variational Quantum Algorithms (VQAs), a framework based on the classical optimization of parametrized quantum circuits. We introduce multi-armed stochastic bandits into this setting, propose an instance-optimal algorithm for continuous single-parameter optimization, and and motivate further exploration of the framework we introduced with an array of experiments.

Keywords

Quantum Computing, Quantum Information Processing, Machine Learning, Learning theory, Deep Learning, Stochastic Bandits

List of Publications

Appended publications

This thesis is based on the following publications:

[**Paper I**] **M. Wanner**, L. L. Lewis, C. Bhattacharyya, D. Dubhashi, A. Gheorghiu, *Predicting Ground State Properties: Constant Sample Complexity and Deep Learning Algorithms*
Advances in Neural Information Processing Systems 37 (NeurIPS 2024).

[**Paper II**] **M. Wanner**,J. Jonasson, Emil Carlsson, D. Dubhashi, *Variational Quantum Optimization with Continuous Bandits*
Submitted, under review.

Acknowledgment

I would like to express my sincere gratitude to my main supervisor, Devdatt Dubhashi, for his guidance, support, and insightful research directions, as well as for connecting me with valuable collaborators. I am also deeply grateful to my co-supervisor, Fredrik Johansson, for his constructive feedback, for providing access to computational resources; I have also greatly appreciated the opportunity to engage with his group and benefit from their insights. I further thank my examiner, Dag Wedelin, for his continued and valuable feedback.

I am grateful to my co-authors for their collaboration and support. In particular, I thank Chiru for many stimulating discussions, and Laura and Andru for their guidance in writing and for sharing their expertise in scientific communication. I also thank Johan and Emil for their excellent collaboration and insightful discussions. I further acknowledge the current and former members of the Quantum Algorithms group, as well as the members of Quantum Stack, for their helpful feedback. I am also grateful to the Healthy AI Lab for the many enriching interactions, including lunch meetings, feedback sessions, stand-ups, and social activities. Finally, I thank my office mates and the members of DSAI for creating a pleasant and productive working environment.

I would also like to thank my friends and family for their support. I am especially grateful to Nicolas, my former Master's thesis supervisor, for his encouragement and support in applying for this position. Finally, I would like to thank you, Linnea, for your constant support and encouragement. You bring great joy to my life, and I am also deeply grateful to your family for making me feel so welcome in Sweden.

This project was funded by the Swedish Foundation for Strategic Research (SSF), grant number FUS21-0063. Computations were enabled by resources provided by the National Academic Infrastructure for Supercomputing in Sweden (NAISS), funded in part by the Swedish Research Council through grant 2022-06725.

Contents

Abstract iii

List of Publications v

Acknowledgement vii

I Introductory Chapters 1

1 Introduction 3

2 Background 5
2.1 Properties of local ground states 5
2.1.1 Parametrized quantum states and observables 5
2.1.2 Geometrically local ground states 6
2.1.3 Exponential decay of correlations 6
2.2 Learning Theory 7
2.2.1 The PAC framework 7
2.2.2 Guarantees for linear regression 8
2.2.3 Methods for deep neural networks 9
2.3 Bandits 9
2.3.1 The bandit model 9
2.3.2 Regret vs best arm 10
2.3.3 Track and stop 11
2.4 Variational Quantum Algorithms 11
2.4.1 Variational Quantum Optimization 11
2.4.2 Barren Plateaus 12

3 Summary of Included Papers 13
3.1 Predicting Ground State Properties: Constant Sample Complexity and Deep Learning Algorithms 13
3.2 Variational Quantum Optimization with Continuous Bandits . 17

4 Discussion and Future Work 21

Bibliography 23

II Appended Papers **27**

Paper I - Predicting Ground State Properties: Constant Sample Complexity and Deep Learning Algorithms

Paper II - Variational Quantum Optimization with Continuous Bandits

Part I

Introductory Chapters

Chapter 1

Introduction

The field of quantum computing has attracted significant attention in recent years [1], [2]. Quantum computers leverage fundamental principles from quantum mechanics, such as superposition and entanglement. These phenomena are generally intractable to simulate efficiently on classical hardware, and it is widely believed that computations native to quantum hardware enable advantages beyond quantum-mechanical simulations, one of the most prominent being Shor's algorithm for prime factorization [3]. Further potential applications exist in various fields, ranging from molecular sciences including quantum physics, chemistry and materials science [4], [5] to industrial domains such as finance and advanced manufacturing [6].
Quantum-physical properties are mostly observed in low-temperature regimes, where systems are in or near their ground states. Such states are of paramount interest in various applications. In quantum many-body physics, the ground state of electronic systems often provides an accurate description their behaviour also at room temperature. In industrial contexts, ground states can be associated with optimal solutions to corresponding optimization problems.
However, obtaining ground states by experiment or simulation is expensive; in particular, the latter is QMA-hard in general [7]. Although efficient quantum algorithms have been developed for ground-state preparation of certain classes of Hamiltonians, such as those with constant spectral gaps, which are classically intractable [8], the practical realization of these algorithms remains challenging: The currently available so-called *Noisy Intermediate-Scale Quantum (NISQ)* devices are limited in both the number of qubits and the depth of coherent gate operations, which are often affected by noise. Consequently, exact simulation methods like Adiabatic Quantum Computation [9] are not well-suited to present-day hardware, and one typically resorts to approximate approaches instead.
One of the most prominent frameworks for this are *Variational Quantum Algorithms (VQAs)* [10], which iteratively apply Parametrized Quantum Circuits (PQCs) in conjunction with classical optimization over the circuit parameters. Their advantage lies in their compatibility with NISQ-era hardware, because they limit the number of gates and harness classical computational resources,

while remaining, in principle, applicable in a noisy setting. However, a major challenge is optimizing the parameters. PQCs do not, in general, permit exact readout of gradients; instead, gradients must be estimated using techniques such as the parameter-shift rule [11], which can be costly. Furthermore, many circuit architectures suffer from the *Barren Plateaus (BP)* phenomenon [12], wherein gradients concentrate around zero as the number of qubits increases, thereby hindering the effectiveness of optimization methods commonly employed in machine learning. So far, all *ansätze*, i.e. models for parametrized circuits, have either shown to exhibit BPs or be classically simulable [13].
Motivated by the difficulty of preparing certain quantum states, recent work has adopted an alternative perspective: if ground states belong to the same family—for instance, the same phase of matter—then information obtained from one representative state may generalize to other members of the family. Accordingly, by acquiring observables from a fixed set of states, it becomes possible to predict properties of other members of the family to a desired level of accuracy, avoiding the need for costly state preparation in each instance. This approach is complementary to the classical shadows framework [14], where properties are inferred from measurements of the same quantum state, and is provably efficient on various geometrically local systems [8]. A series of works improved the sample complexity exponentially with respect to the number of system size across various settings, including ground states [15], thermal states [16], Lindbladian phases with local rapid mixing [17], and even ground states of systems with long-range interactions [18]. However, most of these works do not establish corresponding lower-bounds, leaving open whether the achieved scaling can be improved. Moreover, the results are primarily of complexity theoretical nature. While they show existence of efficient schemes, the proposed algorithms are relatively simple and differ significantly from state-of-the-art methods in classical machine learning. The existence of an algorithm does not guarantee that more advanced methods can obtain the same asymptotic scaling.

Paper I addresses these limitations by proposing a neural network-based learning algorithm to predict geometrically local ground state properties. Under slightly stronger assumptions, we prove rigorous performance guarantees for this model, which improve upon the previous ones from [15]. In our experiments, conducted on a dataset constructed for this work that substantially exceeds prior datasets in size, we demonstrate that the proposed model exhibits significant practical advantages and that the underlying assumptions are satisfied in the respective setting.

In **Paper II**, we address the challenge of preparing ground states by introducing the bandit framework into the landscape of classical optimization algorithms for VQAs. In particular, we take initial theoretical steps toward extending the methods of [19] to a continuous setting. We establish an information-theoretic lower bound, derive an optimal algorithm in the one-dimensional case, and show that, in this setting, best-arm identification is asymptotically equivalent to regret minimization.

Chapter 2

Background

This chapter provides the necessary background for Chapter 3. The notation in these parts may slightly diverge from the one used in **Paper I** and **II** to retain this part of the thesis simple and consistent. **Paper I** and **II** reintroduce all notation to avoid any confusion.

2.1 Properties of local ground states

This section introduces the formalism from quantum mechanics relevant to **Papers I** and **II**. Section 2.1.2 and Section 2.1.3 are only relevant for **Paper I**.

2.1.1 Parametrized quantum states and observables

In this work, we use the density operator formalism, which describes n-qubit quantum states as a symmetric, positive definite matrix $\rho \in \mathbb{C}^{2^n \times 2^n}$ with unit trace. By parametrized quantum state, we refer to a density matrix depending on a set of real parameters $x \in \Omega \subseteq \mathbb{R}^d$, where d is typically of polynomial order in n. In **Paper II**, the parameters arise from the unitary $U(x)$ defining the quantum circuit which constructs $\rho(x) = U(x)\rho_0 U^\dagger(x)$, and typically $x \in [0, 2\pi]^d$.
An *observable* is a Hermitian matrix $H \in \mathbb{C}^{2^n \times 2^n}$. By *measuring* a property or observable, we denote a measurement performed in the basis spanned by the eigenvectors of O. This measurement projects the state ρ onto one of the basis elements of O and the respective *measurement outcome* is the associated eigenvalue. Formally, the outcome of measuring property O of a (parametrized) state $\rho(x)$ can be modelled with a random variable $M(x)$, where by the Born rule, the respective probabilities are

$$P(M(x) = y) = \begin{cases} \langle z| \rho(x) |z\rangle & \text{if } y = o_z, \\ 0 & \text{otherwise.} \end{cases} \tag{2.1}$$

Furthermore, the expected outcome for a fixed x is $\mathbb{E}_z[M(x)] = \mathrm{Tr}[O\rho(x)]$. When the parametrization is with respect to a quantum circuit $U(x)$, it could

equivalently be attributed to the observable as $O(x) = U^\dagger(x)OU(x)$. In **Paper I**, the parameters arise from the governing Hamiltonian $H(x)$, which we discuss in the next section.

2.1.2 Geometrically local ground states

Consider a quantum many-body system composed of n particles with Hamiltonian $H(x)$. Then, its ground (or thermal) states $\rho(x)$ define a parametrized family.[1]

To introduce the setting of **Paper I**, we first consider the $2D$ antiferromagnetic Heisenberg model [20] where particles are fixed on a two-dimensional lattice and only interact with their nearest neighbours. Consequently, the Hamiltonian can be written as

$$H(x) = \sum_{j=1}^{L} h_j(\vec{x}_j) \qquad \text{for any } x \in \mathbb{R}^d, \tag{2.2}$$

where h_j only act non-trivially on neighbouring sites and the parameters x are the concatenated, constant-dimensional vectors $\vec{x}_1, \ldots, \vec{x}_L$. $H(x)$ is an example for a *geometrically local* Hamiltonian. More generally, such Hamiltonians are defined for many-body systems on a D-dimensional lattice and can be decomposed into a sum of L operators h_j only being supported (i.e. acting non-trivially) on sites, which are contained in a D-dimensional ball with constant radius. Importantly, this property pertains to the physical space in which the lattice is embedded, rather than to the parameter space. Similarly, a geometrically local observable denotes a Hermitian $O = \sum_{j=1}^{m} O_j$ where each O_j satisfies the same condition as h_j. This describes the setting of **Paper I** and is adapted from [15], [21].

2.1.3 Exponential decay of correlations

Consider the previously described setup with $f(x) = \mathrm{Tr}[O_i\rho(x)]$, where O_i is the component of a local observable and $\rho(x)$ the ground state of local Hamiltonian $H(x)$. It turns out that $f(x)$ can be well approximated by a function with lower-dimensional input domain, i.e. $f(x) \approx f(x_{\mathrm{loc}})$, where $x \in \mathbb{R}^{d_{\mathrm{loc}}}$ and $d_{\mathrm{loc}} \ll d$.

This is established by proving that most parameters x_j have negligible influence on the value of $f(x)$, due to the respective partial derivative being negligibly small. By the spectral flow formalism [22], [23], [24], the partial derivatives of a ground state of $H(x)$ are given by

$$\frac{\partial}{\partial x_j}\rho(x) = \hat{u} \cdot \nabla_x \rho(x) = i[D_j(x), \rho(x)] \tag{2.3}$$

where $D_j(x)$ depends on $H(x)$ and its derivative with respect to x_j. Applying *Lieb-Robinson bounds* [25] to Equation (2.3) results in a bound stating that

[1] The parametrization could also result from a unitary $U(x)$ which prepares the ground state.

the partial derivatives with respect to x_j decay exponentially with the distance between the support of h_j and O_j, i.e.

$$\frac{\partial}{\partial x_j} f(x) \lesssim e^{-cd_{\text{lattice}}(O_i, h_j)}. \tag{2.4}$$

This strongly impacts the sample complexity, a concept which is introduced in the upcoming section.

2.2 Learning Theory

In this section we introduce some basic and advanced concept of learning theory relevant to this thesis. Section 2.2.1 introduces basic concepts of PAC learning, Section 2.2.2 corresponding results for linear regression and Section 2.2.3 respective counterparts for deep neural networks. The former section is relevant for **Paper I** and **II**, the latter two only for **Paper I**.

2.2.1 The PAC framework

The central objective of learning theory is to rigorously quantify the performance of machine learning algorithms. The first question that arises in this context is what it means to "learn" a function to a sufficient extent. It immediately clear that exact learning of a continuous function is hopeless. Instead, a reasonable objective is to *approximately* learn the function up to error ϵ. Similarly, when the training data is random, it is adequate not to require the learner to always succeed, but allow failure with probability at most δ. The Probably Approximately Correct (PAC) framework formalizes this argument and provides a quantitative characterization of learning performance in terms of the parameters ϵ and δ.

Definition 2.2.1 (PAC learnability [26], adapted to regression)**.** *A hypothesis class $\mathcal{H}$ is PAC learnable if there exist a function $m_{\mathcal{H}} : (0,1)^2 \mapsto \mathbb{N}$ amd a learning algorithm with the following property: For every $\epsilon, \delta \in (0,1)$, for every distribution $\mathcal{D}$ over $\mathcal{X}$, and for every labeling function $f : \mathcal{X} \mapsto \mathbb{R}$, if the realizable assumption holds with respect to $\mathcal{H}, \mathcal{D}, f$, then when running the learning algorithm on $N \geq N_{\mathcal{H}}(\epsilon, \delta)$ i.i.d. examples generated by $\mathcal{D}$ and labeled by f, the algorithm returns a hypothesis h such that, with probability of at least $1 - \delta$ (over the choice of the examples), the generalization error satisfies $L_{(\mathcal{D},f)}(h) \leq \epsilon$.*

In this section, we stick to the L_2-error as our notion of generalization error (expected risk), implying that

$$L_{(\mathcal{D},f)}(h)^2 = \mathop{\mathbb{E}}_{x \sim \mathcal{D}}[(f(x) - h(x))^2], \tag{2.5}$$

but many more choices exist. The component most crucial to the PAC bounds of this work is the *sample complexity* $N_{\mathcal{H}}(\epsilon, \delta)$, the quantity denoting the number of training data needed to learn a function to some required accuracy

ϵ with success probability at least $1-\delta$. Obtaining a favourable scaling of $N_{\mathcal{H}}$ in terms of ϵ, δ and the number of qubits n is the main goal of **Paper I**.
Finally, the PAC framework relies on the *realizability assumption*, namely that the target function belongs to the hypothesis class $\mathcal{H}$. An alternative framework that relaxes this assumption is agnostic learning, which we do not consider here.

2.2.2 Guarantees for linear regression

A common approach to deriving PAC bounds is to analyze the convergence of the training error to the generalization error in probability, i.e.

$$\Pr_{S\sim\mathcal{D}}[|L_{(\mathcal{D},f)}(h) - \hat{L}_S(h)| \leq \epsilon(N)], \tag{2.6}$$

where $\hat{L}_S(h) = \frac{1}{N}\sum_{\ell=1}^{N} l(x_\ell, y_\ell)$ is the training error, $S = \{(x_\ell, y_\ell)\}_{\ell=1}^{N}$ a training set i.i.d. sampled from $\mathcal{D}$ and $\epsilon(N)$ denotes the convergence rate. When the hypothesis class is given by linear regression, i.e. $h(x) = w^T x$ for some $w \in \mathbb{R}^d$, in the setting of Theorem 2.2.2, the convergence rate can be shown to satisfy

$$\Pr_{S\sim\mathcal{D}}\left[|L_{(\mathcal{D},f)}(h) - \hat{L}_S(h)| \leq \frac{2L\Lambda R + c\sqrt{2\log(2/\delta)}}{\sqrt{N}}\right] \geq 1-\delta. \tag{2.7}$$

Applying the reverse triangle inequality to the left-hand side yields an upper-bound for the generalization error.

Theorem 2.2.2 (Informal statement of Theorem 26.12 in [26])**.** *Suppose that $\mathcal{D}$ is the data distribution with $\|\boldsymbol{x}\|_2 \leq R$. Let $\mathcal{H} = \{\boldsymbol{x} \mapsto \boldsymbol{w}\cdot\boldsymbol{x} : \|\boldsymbol{w}\|_2 \leq \Lambda\}$ and l a loss function of the form $\ell(\boldsymbol{w}, (\boldsymbol{x}, y)) = \phi(\boldsymbol{w}\cdot\boldsymbol{x}, y)$, where ϕ is L-Lipschitz in the first argument and its magnitude is upper-bounded by c. Then, for any $\delta \in (0,1)$, with probability of at least $1-\delta$ over the choice of an i.i.d. sample of size N,*

$$\forall h \in \mathcal{H}, \qquad L_{(\mathcal{D},f)}(h) \leq \hat{L}_S(h) + \frac{2L\Lambda R}{\sqrt{N}} + c\sqrt{\frac{2\log(2/\delta)}{N}}. \tag{2.8}$$

Note that Theorem 2.2.2 does not rely on the *realizable assumption* and therefore cannot, by itself, guarantee a small generalization error. It must further be shown that the employed training (optimization) identifies $h \in \mathcal{H}$ for which the training error $\hat{L}_S(h)$ converges to zero with $N \to \infty$; ideally at least as quickly as $\epsilon(N)$. Therefore, one typically shows that there is a model $h \in \mathcal{H}$ (in this case $w \in \mathbb{R}^d$) which approximates f well, and that the training algorithm recovers the empirical risk minimizer $h^* = \arg\min_{h\in\mathcal{H}} \hat{L}_S(h)$. Substituting h^* into Theorem 2.2.2 yields a meaningful upper-bound.
Note that there is often a trade-off between training error and $\epsilon(N)$. This trade-off is closely related to the phenomenon of *overfitting* and can be mitigated, for instance, by regularizing the loss function, which is also reflected in the corresponding generalization bound.

2.2.3 Methods for deep neural networks

Similar tools have recently been developed for settings where both the target function and the hypothesis class belong to Sobolev spaces $H^k(\mathcal{X})$, i.e., spaces of functions whose mixed derivatives up to order k are square-integrable on $\mathcal{X}$. In [27], the *Koksma-Hlawka* inequality was employed in a manner analogous to Equation (2.7) to obtain a respective bound for the generalization error.

Theorem 2.2.3 (Koksma-Hlawka inequality)**.** *Let $f : [0,1]^d \to \mathbb{R}$ be a function whose mixed derivatives are absolutely integrable over its domain with bounded Hardy-Krause variation $V_{HK}(f) < \infty$. Let $x = \{x_\ell\}_{\ell=1}^N$ be a sequence of N d-dimensional points in $[0,1]^d$ with star-discrepancy $D_N^*(d)$. Then*

$$\left| \int_{[0,1]^d} f(x)\, dx - \frac{1}{N} \sum_{\ell=1}^{N} f(x_\ell) \right| \leq V_{HK}(f) D_N^*(d). \tag{2.9}$$

The generality of Theorem 2.2.3 allows to consider deep neural networks and can be combined with the main result from [27] for a useful upper-bound on the expected risk.

Theorem 2.2.4 (Informal special case of Theorem 5.1 in [28], informal)**.** *Let $d, k \in \mathbb{N}_0$ and let f be k times continuously differentiable on $[0,1]^d$. For every $\epsilon > 0$, there exists a* tanh *neural network $\hat{f}^\epsilon$ with two hidden layers and width $\mathcal{O}(\epsilon^{-d/k})$ such that*

$$|f(x) - \hat{f}^\epsilon| < \epsilon \qquad \forall x \in [0,1]^d. \tag{2.10}$$

This result establishes the *realizable assumption* for the fairly general class functions in $H^k(\mathcal{X})$, under the hypothesis class $\mathcal{H}$ consisting of sufficiently wide two layer tanh neural networks. The formal statement further provides estimates for the weights of the neural network, which approximates f well. These in turn affect the Hardy-Krause variation of the loss, again showing a relation to overfitting.

2.3 Bandits

This section provides preliminary background from the bandit literature relevant to **Paper II**. Section 2.3.1 introduces the bandit model, Section 2.3.2 objectives for bandit algorithms and Section 2.3.3 the algorithmic framework **Paper II** builds on.

2.3.1 The bandit model

The multi-armed stochastic bandit model [29] is inspired by the slot machines in casinos, often called one-armed bandits. In the finite arm setting, the bandit defines a collection of m random variables (arms) with expected values $\mu = [\mu_1, \mu_2, \ldots, \mu_m]$. In the following, we refer to a bandit by μ and to its arms j by μ_j. The only way to interact with a bandit is by playing its arms, i.e.

sampling from one of the random variables. As the casino analogy suggests, the related tasks involve maximizing the expected reward, after T rounds of playing an arm. There are various notions of "maximizing the expected reward", of which we introduce two in the upcoming section. What differentiates this class of learning tasks from the one in Section 2.2 is the data access model: the learner can select which element of the domain to sample, with each choice potentially depending on the outcomes observed in previous rounds.
More generally, the set of arms is represented by a pair $(\mathcal{X}, M)$ of a measurable space $\mathcal{X}$ and a set of random variables M. This formulation allows for a continuous set of arms, a setting that is explored in **Paper II**. For simplicity, we stick to the finite arm setting in the remainder of this section.

2.3.2 Regret vs best arm

From now on, let (without loss of generality) $\mu_1 > \mu_j$ for all $2 \leq j \leq m$. In the bandit literature, the loss or risk is called *regret*, and is defined for each arm as $r_j = \mu_1 - \mu_j$. With slight abuse of notation, it is common to refer to the regret in round t by $r_t = r_{a(t)}$ where $a(t)$ is the arm the learner plays in round t. The literature distinguishes *cumulative* and *simple regret*. Cumulative regret $R(T)$ is the regret accumulated after T rounds, i.e.

$$R(T) = \sum_{t=1}^{T} r_t. \tag{2.11}$$

Simple regret is the learner's guess for the best arm μ_1 after T rounds. These notion give rise to two different learning tasks.

Definition 2.3.1 (Regret minimization)**.** *The learner aims to sample in a way minimizes $R(T)$, or formally solves*

$$\text{minimize} \sum_{t=1}^{T} r_t. \tag{2.12}$$

Definition 2.3.2 (Best arm identification (BAI))**.** *The learner optimizes for simple regret, i.e. aims to identify the arm with largest mean after as few samples as possible. Formally, this solves*

$$\text{minimize } T_\delta \quad \text{s.t.} \quad P(\mu_T \neq \mu_1) \leq \delta. \tag{2.13}$$

Note that although these two objectives seem superficially similar, the respective optimal algorithms are fundamentally different. In regret minimization, the learner aims to avoid sampling suboptimal arms, whereas in BAI, it may draw a large number of samples from them if doing so optimally improves the confidence interval around the estimate of the best arm. Nevertheless, any regret minimization algorithm can be used for BAI by returning a uniformly random arm from the played arms $\{a(t)\}_{t=1}^{T}$, allowing for a simple connection between the worst-case guarantees. In general, however, this approach does not yield optimal BAI strategies. The next section introduces a widely used class of such strategies.

2.3.3 Track and stop

A notable result in the literature establishes the information-theoretically minimal number of samples required to achieve the best-arm identification (BAI) objective.

Theorem 2.3.3 (adapted from [19]). *Consider the bandit $\mu = [\mu_1, \mu_2, \ldots, \mu_m]$. For any (ϵ, δ)-PAC learner,*

$$\mathbb{E}[T_\delta^\epsilon] \geq \frac{\log(1/\delta)}{c^*(\mu)}, \tag{2.14}$$

where

$$c^*(\mu) = \sup_{w \in \mathcal{W}} \inf_{\lambda \in \mathrm{Alt}(\mu)} \int_{\mathcal{X}} w_j \mathrm{kl}(\mu_j, \lambda_j) dx. \tag{2.15}$$

The set $\mathcal{W}$ denotes the probability simplex, i.e. the set of vectors w with positive elements whose sum equals one and $\mathrm{kl}(\cdot, \cdot)$ the *Kullback-Leibler* divergence. The set $\mathrm{Alt}(\mu)$ refers to the set of *alternate instances*, i.e. all bandits whose optimal arm is not μ_1. Note that Theorem 2.3.3 is instance-specific, meaning that it depends explicitly on the bandit μ, in contrast to lower bounds that hold uniformly over all worst-case instances. Intuitively, it describes an interplay between finding the alternate instance which is hardest to distinguish from μ versus determining the optimal sampling strategy to discriminate between them. Remarkably, the quantity $c^*(\mu)$ therefore gives rise to a sampling strategy. The *Track-and-stop* algorithm [19] estimates μ from the previously collected samples, solves the optimization problem in Equation (2.15) and samples according to the respective sampling ratios w. Equipped with an appropriate stopping criterion, this algorithm was shown to be optimal for $\delta \to 0$. Over the years, numerous variations of this algorithm and corresponding settings have been investigated [30], [31], [32]. In **Paper II**, we investigate the one where μ is a Lipschitz function on a continuous set of arms. This has direct applications to variational quantum algorithms, which we introduce in the next section.

2.4 Variational Quantum Algorithms

This section provides a brief introduction to variational quantum optimization and Barren Plateaus as a challenge, concepts which are relevant for **Paper II**.

2.4.1 Variational Quantum Optimization

In variational quantum optimization one considers a Parametrized Quantum Circuit (PQC) as hinted to in Section 2.1.1. It can be shown that each PQC can be expressed as a sequence of p parametrized unitaries $U(x) = \prod_{i=1}^{p} U_i(x_i)$, where $x = (x_1, ..., x_L)$ is a set of trainable parameters [33]. In variational quantum optimization, these parameters are trained towards a given objective, typically formulated as maximizing the expected outcome for measuring some property O. The optimization is performed by alternately evaluating the circuit

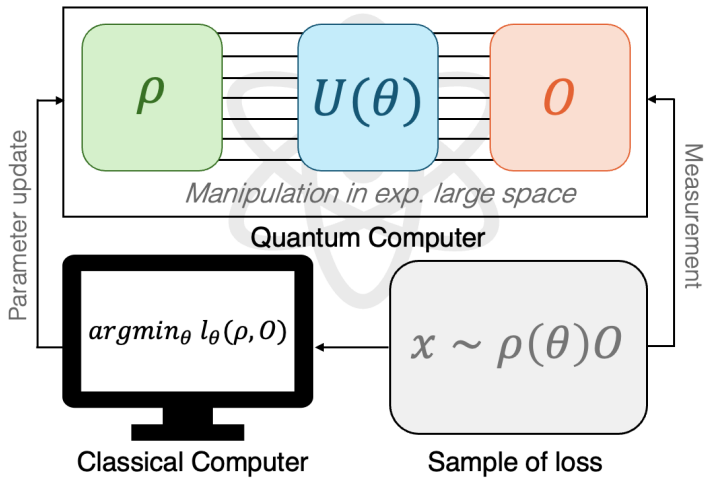

Figure 2.1: Illustration of VQA inspired by [12]

and updating its parameters, as illustrated in Figure 5.6. It was shown that the exact gradients are given by the parameter-shift rule

$$\nabla_x f(x) = \frac{1}{2}\left(f(x+\frac{\pi}{2}) - f(x-\frac{\pi}{2})\right), \tag{2.16}$$

where $f(x) = \mathrm{Tr}[U(x)\rho_0 U(x)^\dagger O]$, if the gates are parametrized as $U(x_i) = e^{-i\frac{x}{2}P}$ and P denotes some Pauli operator [11]. Therefore, descendants of gradient descent belong, among other finite-difference-based, zero-order methods, to the most popular optimization algorithms.

The optimization is, however, NP-hard in general [34] and is complicated by a phenomenon called Barren Plateaus which we introduce in the upcoming section.

2.4.2 Barren Plateaus

The *Barren Plateau*-phenomenon [12] is the property of quantum circuits that the gradients of PQCs concentrate around zero with increasing number of qubits. Formally, there exists a constant $b > 1$, such that

$$\Pr_x[|\partial_x \mathrm{Tr}[O\rho(x)] - \mathbb{E}_x[\partial_x \mathrm{Tr}[O\rho(x)]]| \geq \delta] = \mathcal{O}\left(\frac{1}{b^n}\right), \tag{2.17}$$

where $\mathbb{E}_x[\partial_x \mathrm{Tr}[O\rho(x)] = 0$ for a generic ansatz [33], and it turns out that there is a trade-off between trainability and expressivity. Approaches to mitigate BPs have been proposed [35], [36], however, recent work suggests that the absence of BPs implies classical simulability [13].

Chapter 3

Summary of Included Papers

3.1 Predicting Ground State Properties: Constant Sample Complexity and Deep Learning Algorithms

In this paper we present two algorithms with sample complexity *constant* with respect to the number of qubits for predicting geometrically local ground state properties of geometrically local quantum systems, as introduced in Section 2.1.2. The first algorithm constitutes a small extension to [15]. The second leverages deep neural networks, providing, to the best of our knowledge, the first rigorous guarantees for neural network models of this kind in this setting.

The task we study in this work is to obtain a classical representation of ground state properties of an entire parametrized family of states, where Hamiltonian and observable are geometrically local, as introduced in Section 2.1.2. We are given a labelled training dataset $\{(x_\ell, y_\ell)\}_{\ell=1}^N$, where $y_\ell \approx \mathrm{Tr}[O\rho(x_\ell)]$, with the eigenvalues of O bounded in absolute value by

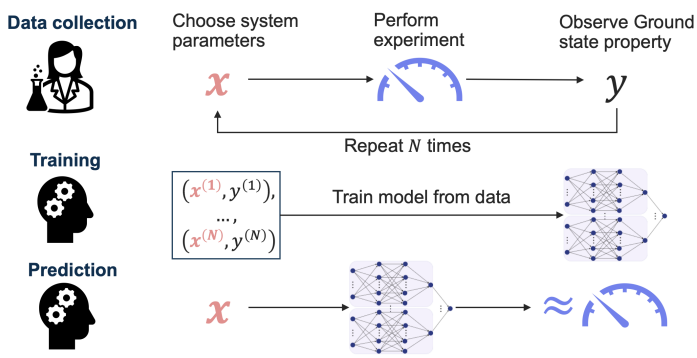

Figure 3.1: Machine Learning protocol for predicting ground state properties.

1. The objective is to train a machine learning model $h^*(x)$, which predicts

$$f(x) = \mathrm{Tr}[O\rho(x)] \text{ for all } x \in [-1,1]^d.$$

The central question of this work concerns the dependence of the generalization error ϵ on the number of training data N, in terms of a PAC bound, as introduced in Section 2.2.1. Direct application of the bounds from Section 2.2.2 results in unfavourable scaling with respect to system size; therefore, one on aims to exploit the geometric locality of the Hamiltonian and the observable. Prior work [15] leverages the fact that any geometrically local variable can be decomposed into local Pauli operators $\alpha_P P$, allowing one to write

$$f(x) = \sum_i \mathrm{Tr}[O_i\rho(x)] = \sum_P \alpha_P \mathrm{Tr}[P\rho(x)] = \sum_P \alpha_P f_P.$$

Furthermore, it is shown that the coefficient vector α satisfies $|\alpha|_1 = \mathcal{O}(1)$. The corresponding model then exploits the exponential decay of correlations of each individual component f_P, which implies that each term can be approximated with an accuracy that depends on $\epsilon(N)$. The resulting feature map is given by

$$\phi(x)_{x',P} \triangleq \mathbb{1}[x \in T_{x',P}], \tag{3.1}$$

where $T_{x',P}$ is the (cube-shaped) cell with centre x' on a uniform grid on $[-1,1]^{d_{\mathrm{loc}}}$ with coarseness depending on N.
Our first contribution proposes a slight modification of this algorithm, under the assumption that the target observable is known, yielding a sample complexity that is independent of the number of qubits n.

Theorem 3.1.1 (Informal)**.** *Let $H(x)$ be an n-qubit gapped, geometrically local Hamiltonian with ground state $\rho(x)$. Given an observable O, with a known decomposition as a sum of local Pauli operators and given training data $\{(x_\ell, y_\ell)\}_{\ell=1}^N$ sampled from an arbitrary distribution, with $y_\ell \approx \mathrm{Tr}(O\rho(x_\ell))$, there is an ML algorithm for predicting ground state properties $\mathrm{Tr}(O\rho(x))$ to within precision $\epsilon > 0$ using $N = \mathcal{O}\left(2^{\mathrm{polylog}(1/\epsilon)}\right)$ training samples.*

The new map is defined as

$$\tilde{\phi}(x)_{x',P} \triangleq \mathrm{sign}(\alpha_P)\sqrt{|\alpha_P|}\mathbb{1}\{x \in T_{x',P}\}. \tag{3.2}$$

We prove the result by using that $\|\alpha\|_1$ is constant, together with Theorem 2.2.2.

Our second result improves on this approach by dropping the assumption that the observable is known. Instead, we assume that the partial derivatives of the probability density function of the data distribution and of $H(x)$ are sufficiently well-behaved, and that the data distribution does not exhibit correlations across parameters associated with different h_j, consistent with the geometric locality of the system. This enables the application of the methods from Section 2.2.3 to the prediction task, yielding a bound that is sufficiently general to encompass even neural networks. However, we do not prove convergence of training, so that in this case, the bound only is effective after successful training. Nonetheless, the constant sample complexity is retained for other suitable models which admit guarantees for training.

Theorem 3.1.2 (Informal)**.** *Let $H(x)$ be an n-qubit gapped, geometrically local Hamiltonian with ground state $\rho(x)$. For any observable O, expressible as a sum of local Pauli operators and given training data $\{(x_\ell, y_\ell)\}_{\ell=1}^N$, sampled from a distribution satisfying certain assumptions with $y_\ell \approx \mathrm{Tr}(O\rho(x_\ell))$, there is a neural network ML algorithm for predicting ground state properties $\mathrm{Tr}(O\rho(x))$, for uniform x, to within precision $\epsilon > 0$ using $N = \mathcal{O}\left(2^{\mathrm{polylog}(1/\epsilon)}\right)$ training samples under mild assumptions on training.*

The employed deep neural network is constructed as a linear combination of local approximations of f_P, with weights intended to approximate the coefficients α_P. Accordingly, the network's connectivity, as depicted in Figure 3.2, mirrors the local approximation used in [15].

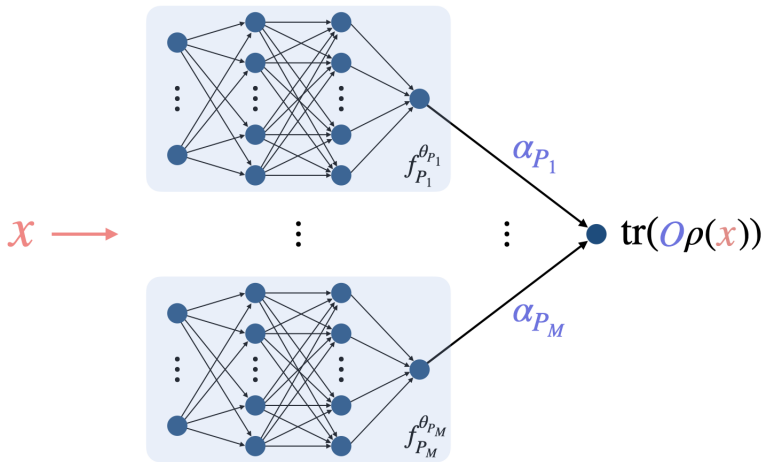

Figure 3.2: Sketch of the deep network model for predicting ground state properties.

To prove Theorem 3.1.2, we iteratively apply the derivative rule given by spectral flow formalism Section 2.1.3 to explicitly upper-bound the Hardy-Krause variation from Theorem 2.2.3 for a single local property f_P. Combining this insight with the bound $\|\alpha\|_1 \leq C$, we show that constant sample complexity extends to the sum of local predictors represented by our model.

Finally, we simulate a considerable amount of additional data for the numerical experiments, where the properties are Pauli correlators of the $2d$ Heisenberg spin-1/2 model whose size exceeds the one from previous work. As can be seen in Figure 3.3, the deep learning-based algorithm demonstrates clear practical advantage and also gives evidence that the trainability of the deep neural network is not affected by the number of qubits.

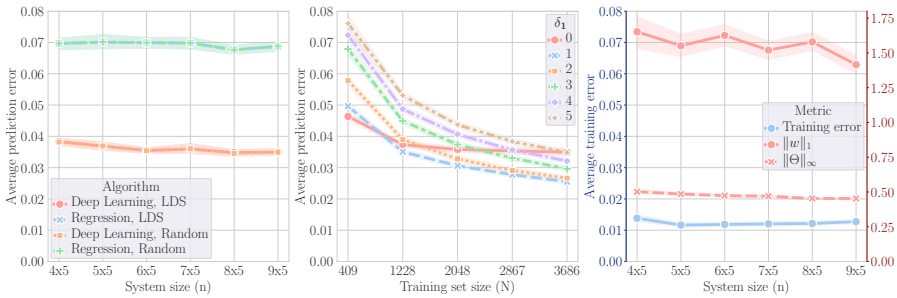

Figure 3.3: **Numerical experiments. (Left)** Comparison with previous methods. Each point indicates the prediction error (RMSE) of our deep learning model or the regression model of [15]. **(Center)** Scaling with training size for various cutoff radii δ_1. **(Right)** Neural network weights and training error. Blue points correspond to the training error of the neural network model. Red points correspond to the ℓ_1 norm of parameters in the last layer or the largest absolute value of the parameters of the neural network.

3.2 Variational Quantum Optimization with Continuous Bandits

In this paper, we address the task of optimizing VQAs by introducing the multi-armed bandit framework to the problem. The inherently global perspective of this optimization framework mitigates the problem of the optimizer being "stuck" in regions where the gradients concentrate around zero. Although the intrinsic computational hardness of the optimization problem remains, this work is a step toward an instance-specific, information-theoretically optimal method.

We consider a version of bandit model discussed in Section 2.3.1, with arms defined over the space $([0,1]^d, M(x))$. The PQCs introduced in Section 2.4.1 naturally correspond to continuous bandits, since the only way to interact with them is to sample outcomes, given a set of parameters. Furthermore, the respective optimization problem corresponds to identifying the best arm x^* of the bandit $\mu(x) = \mathrm{Tr}[O\rho(x)]$ (see Section 2.1.1 for a proper introduction).
To solve the BAI problem more efficiently, we exploit the inherent structure of PQCs. It can be shown that PQCs satisfy the following structural assumptions:

(i) The rewards $M(x)$ are 1-sub-Gaussian.

(ii) The expected reward $\mu(x)$ is L-Lipschitz.

Paper II makes some additional technical assumptions, mainly for convenience. These insights lead to the first part of our theoretical results, which extends the ideas from Section 2.3.3 to the Lipschitz bandit model with continuous arms. The first result is an extension of Theorem 2.3.3 to the continuous setting.

Theorem 3.2.1. *Let μ be a bandit continuous bandit on some set $\mathcal{X}$. For any (ϵ, δ)-PAC learner,*

$$\mathbb{E}[\tau_\delta^\epsilon] \geq \frac{\log(1/\delta)}{c^*(\mu)}, \tag{3.3}$$

where

$$c^*(\mu) = \sup_{w \in \mathcal{W}} \inf_{\lambda \in \mathrm{Alt}^\epsilon(\mu)} \int_{\mathcal{X}} w(x)(\mu(x) - \lambda(x))^2 dx. \tag{3.4}$$

Since an algorithm analogous to Track-and-stop from Theorem 3.2.1 would hardly be tractable, we break up the space the arms are defined over into level-sets, which is illustrated in Figure 3.4. This leads to a simpler, more tractable lower-bound for the special case that the domain the arms are given on is one-dimensional, approximating $c^*(\mu)$ from above.

Theorem 3.2.2 (Informal)**.** *Let $v : [0,1] \to \mathbb{R}$ satisfy assumptions (i) and (ii). Then,*

$$\mathbb{E}[\tau_\delta^\epsilon] = \mathcal{O}\left(\frac{L\log(1/\delta)}{\epsilon^3} u_f(\epsilon)\right), \tag{3.5}$$

where $\epsilon \lesssim u_f(\epsilon) \lesssim 1$.

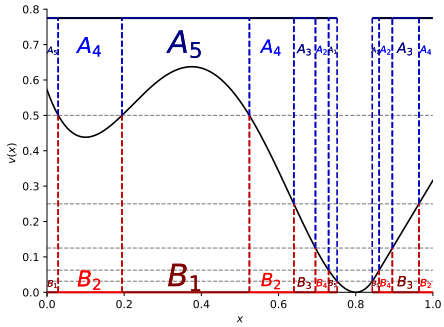

Figure 3.4: Example for the simplified structure used in Theorem 3.2.2.

The second theory part complements the lower bounds from Theorem 3.2.2 with matching (up to a log-factor) upper bounds for the $1d$-setting, by proposing Algorithm 1. We propose an algorithm complementing Theorem 3.2.2, which is optimal for the one-dimensional setting.

Algorithm 1: Reject and Refine (RR)

Input: Inverted bandit $v(\cdot)$, constants L, ϵ.
Initialize $G_0 = [0,1]$, $t = 1$.
repeat
  for all $h \in G_{t-1} \cap H_t$ **do**
    Draw n_t samples from $v(h)$.
    Compute $\hat{v}(h)$
    Construct $1 - \frac{\delta}{|H_t| 2^t}$ CI of length $\frac{1}{2^{t+3}}$.
    $a_t^* \leftarrow \operatorname{argmin}_{h \in H_1} \hat{v}(h)$
    $E_t \leftarrow \bigcup_{h : \hat{v}(h) - \hat{v}(a_t^*) > \frac{12}{2^{t+4}}} \left[h - \frac{1}{2^{t+4}}, h + \frac{1}{2^{t+4}} \right]$
    $G_t \leftarrow G_{t-1} \setminus E_t$
    $t \leftarrow t + 1$
  end for
until $2^{-t} \leq \epsilon$
Output: $a^* = \operatorname{argmin}_{a_t^*} \hat{v}(a_t^*)$.

Algorithm 1 solves an equivalent optimization problem to BAI, namely the one of minimizing $v(x) = 1 - \mu(x)$. Intuitively, it samples in each round from an initially uniform grid H_t of points on $[0,1]$ and uses sub-Gaussianity of μ to construct confidence intervals for all estimators of $v(x_i)$ with $x_i \in H_t$. These confidence intervals, together with the Lipschitz property, are then used to determine whether the neighborhoods around the points x_i could contain the global optimum x^*. Neighborhoods that do not contain x^* with high probability are excluded in subsequent rounds, while the grid of remaining points is refined in the following round. This process is illustrated in Figure 3.5. We refer to **Paper II** for a rigorous introduction of Algorithm 1. The algorithm

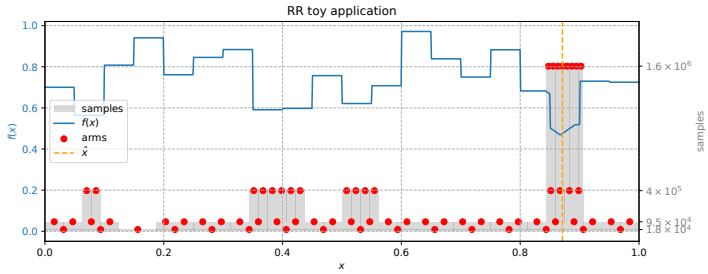

Figure 3.5: Toy function **(blue)** and example run of Algorithm 1 with D rounds. The red dots indicate the *arms* sampled from and the gray area the number of samples for each arm. The minimum estimated by the algorithm $\hat{x}$ is depicted by the orange dashed line.

achieves the following sample complexity.

Theorem 3.2.3 (Informal)**.** *Let $v : [0,1] \to \mathbb{R}$ satisfy assumptions (i) and (ii). Then Algorithm 1 achieves sample complexity*

$$\tau_\delta^\epsilon \leq \tilde{\mathcal{O}}\left(\frac{L\log(1/\delta)}{\epsilon^3} u_f(\epsilon)\right), \tag{3.6}$$

where $\epsilon \lesssim u_f(\epsilon) \lesssim 1$.

Algorithm 1 is a direct BAI analogue of tree-based regret minimization algorithms like the method proposed in [37]. The latter quantifies the simple regret it achieves in terms of the *zooming dimension*, which we show to be related to $u_f(\epsilon)$ in the limit $\epsilon \to 0$, enabling a direct comparison. In the one-dimensional setting, both the upper and lower bounds coincide with the corresponding results for regret minimization, indicating that the two objectives are asymptotically equivalent when $\epsilon \to 0$.

Corollary 3.2.4 (Informal)**.** *Let $v : [0,1] \to \mathbb{R}$ satisfy assumptions (i) and (ii). Then, when $\epsilon \to 0$, $\mathbb{E}[\tau_\delta^\epsilon]$ matches the lower bound for the simple regret an optimal regret minimization algorithm exhibits.*

Although we consider an extension of our theory to the multidimensional setting possible under similar assumptions as in [37], we abstain from doing so since Algorithm 1 would lose its practicality already for moderately large d. Instead, we propose surrogate approaches that enable the application of the $1d$ algorithm to higher-dimensional problems. We evaluate these methods empirically on a set of test cases, including a toy problem, an instance exhibiting barren plateaus, and a standard application of QAOA.

The surrogates are very simple and only meant to probe the multidimensional setting in a proof-of-concept fashion. Notably, they match or in some cases outperform the state of the art finite-difference based algorithms. We interpret these results as an argument against the use of the latter, rather than an argument for the surrogates. However, they also show the potential of the bandit-framework to this type of problems.

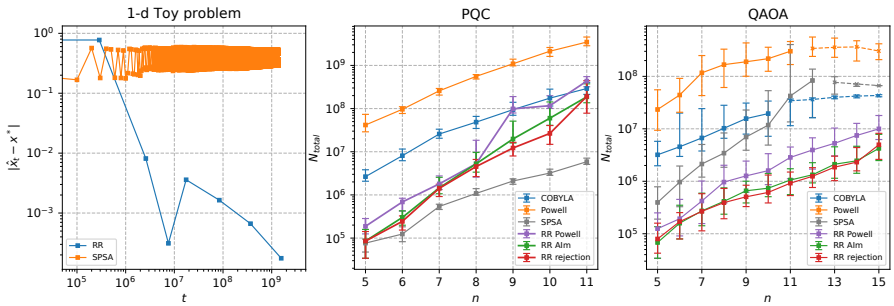

Figure 3.6: (1): Distance $|\hat{x}_t - x^*|$ to the true optimum at sample t for runs of Algorithm 1 and SPSA **(left)**. (2): Median and $(0.25, 0.75)$-quantiles of the sample complexity N_{total} for optimizing a PQC **(middle)** and MaxCut-QAOA **(right)** below the thresholds $C = 0.4$ and $C = 0.2$, respectively. N_{total} refers to the sample complexity and n to the number of qubits. The medians are computed over 20 and 100 simulations, respectively. Failure of convergence is indicated by a dashed line.

Chapter 4

Discussion and Future Work

Paper I opens up promising directions to practically powerful methods with provable guarantees. Nevertheless, several limitations remain. In particular, the current results are restricted to ground states, and the trainability of the proposed neural network approach is not yet theoretically proven. A first extension would be to generalize the framework to broader classes of states, such as thermal states or states arising from Lindbladian dynamics. The latter may be addressed using existing results on overparameterized neural networks [38]. From a practical perspective, a further drawback is that the model must be retrained for each observable; addressing this limitation would improve its applicability.
Another major shortcoming is the quasi-polynomial number of training data needed to achieve prediction error ϵ. The algorithms do not take advantage of the structure for improvements beyond lower-dimensional approximation; incorporating finer structural properties, such as the behaviour of higher-order derivatives, may lead to improved convergence rates. Corresponding lower-bounds remain another important open question.
Further, recent advances on non-local observables [18] propose an algorithm that extends the framework beyond geometrically local settings, representing an interesting extension to the aforementioned research directions. Finally, it would be of interest to extend the approach to learning parametrized Hamiltonians themselves, as considered in [39].

Paper II is limited by its restriction to one-dimensional settings. Extending the framework to more general settings would therefore be of interest. In this context, finite element methods may provide a natural approach to directly estimate c^* from Theorem 3.2.1 in a track-and-stop fashion. Alternatively, the Bayesian learning framework offers a promising direction for addressing the same problem.

Finally, establishing a bound of the form of Theorem 3.1.1 for Gaussian processes could enable the development of an adaptive algorithm analogous to that in **Paper II** for optimizing parametrized quantum states. This would

provide a Bayesian learning counterpart to the method in **Paper II** with rigorous guarantees, while additionally allowing to exploit the local structure some parametrized quantum states may exhibit.

Bibliography

[1] G. Acampora et al., *Quantum computing and artificial intelligence: Status and perspectives*, 2025. arXiv: 2505.23860 [quant-ph]. [Online]. Available: https://arxiv.org/abs/2505.23860 (cit. on p. 3).

[2] D. Conde-Torres, A. Seco-González, Á. Piñeiro and R. Garcia-Fandiño, "Mapping the quantum computing landscape: Growth, collaboration, and thematic convergence," *EPJ Quantum Technology*, vol. 13, no. 1, p. 7, 2026 (cit. on p. 3).

[3] P. W. Shor, "Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer," *SIAM review*, vol. 41, no. 2, pp. 303–332, 1999 (cit. on p. 3).

[4] Y. Alexeev et al., "A perspective on quantum computing applications in quantum chemistry using 25–100 logical qubits," *Journal of Chemical Theory and Computation*, vol. 21, no. 22, pp. 11 335–11 357, 2025 (cit. on p. 3).

[5] H. Liu, G. H. Low, D. S. Steiger, T. Häner, M. Reiher and M. Troyer, "Prospects of quantum computing for molecular sciences," *Materials Theory*, vol. 6, no. 1, p. 11, 2022 (cit. on p. 3).

[6] F. Bova, A. Goldfarb and R. G. Melko, "Commercial applications of quantum computing," *EPJ quantum technology*, vol. 8, no. 1, p. 2, 2021 (cit. on p. 3).

[7] J. Kempe, A. Kitaev and O. Regev, "The complexity of the local hamiltonian problem," *Siam journal on computing*, vol. 35, no. 5, pp. 1070–1097, 2006 (cit. on p. 3).

[8] H.-Y. Huang, R. Kueng and J. Preskill, "Provable machine learning algorithms for quantum many-body problems," *to appear soon*, (cit. on pp. 3, 4).

[9] T. Albash and D. A. Lidar, "Adiabatic quantum computation," *Reviews of Modern Physics*, vol. 90, no. 1, p. 015 002, 2018 (cit. on p. 3).

[10] M. Cerezo et al., "Variational quantum algorithms," *Nature Reviews Physics*, vol. 3, no. 9, pp. 625–644, 2021, ISSN: 2522-5820. DOI: 10.1038/s42254-021-00348-9. [Online]. Available: https://doi.org/10.1038/s42254-021-00348-9 (cit. on p. 3).

[11] K. Mitarai, M. Negoro, M. Kitagawa and K. Fujii, "Quantum circuit learning," *Physical Review A*, vol. 98, no. 3, Sep. 2018, ISSN: 2469-9934. DOI: 10.1103/physreva.98.032309. [Online]. Available: http://dx.doi.org/10.1103/PhysRevA.98.032309 (cit. on pp. 4, 12).

[12] M. Larocca et al., "A review of barren plateaus in variational quantum computing," *CoRR*, vol. abs/2405.00781, 2024 (cit. on pp. 4, 12).

[13] M. Cerezo et al., *Does provable absence of barren plateaus imply classical simulability? or, why we need to rethink variational quantum computing*, 2024. arXiv: 2312.09121 [quant-ph]. [Online]. Available: https://arxiv.org/abs/2312.09121 (cit. on pp. 4, 12).

[14] H.-Y. Huang, R. Kueng and J. Preskill, "Predicting many properties of a quantum system from very few measurements," *Nat. Phys.*, vol. 16, 1050—1057, 2020. [Online]. Available: https://doi.org/10.1038/s41567-020-0932-7 (cit. on p. 4).

[15] L. Lewis, H.-Y. Huang, V. T. Tran, S. Lehner, R. Kueng and J. Preskill, "Improved machine learning algorithm for predicting ground state properties," *Nature Communications*, vol. 15, no. 1, p. 895, 2024 (cit. on pp. 4, 6, 13–16).

[16] E. Onorati, C. Rouzé, D. S. França and J. D. Watson, "Efficient learning of lattice quantum systems and phases of matter," *arXiv preprint arXiv:2301.12946*, 2023 (cit. on p. 4).

[17] E. Onorati, C. Rouzé, D. S. França and J. D. Watson, "Provably efficient learning of phases of matter via dissipative evolutions," *arXiv preprint arXiv:2311.07506*, 2023 (cit. on p. 4).

[18] Š. Šmíd and R. Bondesan, "Efficient learning of long-range and equivariant quantum systems," *Quantum*, vol. 9, p. 1597, 2025 (cit. on pp. 4, 21).

[19] A. Garivier and E. Kaufmann, "Optimal best arm identification with fixed confidence," in *29th Annual Conference on Learning Theory*, V. Feldman, A. Rakhlin and O. Shamir, Eds., ser. Proceedings of Machine Learning Research, vol. 49, Columbia University, USA: PMLR, 2016, pp. 998–1027. [Online]. Available: https://proceedings.mlr.press/v49/garivier16a.html (cit. on pp. 4, 11).

[20] S. Chakravarty, B. I. Halperin and D. R. Nelson, "Two-dimensional quantum heisenberg antiferromagnet at low temperatures," *Physical Review B*, vol. 39, no. 4, p. 2344, 1989 (cit. on p. 6).

[21] H.-Y. Huang, R. Kueng, G. Torlai, V. V. Albert and J. Preskill, "Provably efficient machine learning for quantum many-body problems," *Science*, vol. 377, no. 6613, eabk3333, 2022 (cit. on p. 6).

[22] S. Bachmann, S. Michalakis, B. Nachtergaele and R. Sims, "Automorphic equivalence within gapped phases of quantum lattice systems," *Commun. Math. Phys.*, vol. 309, no. 3, pp. 835–871, 2012 (cit. on p. 6).

[23] M. B. Hastings and X.-G. Wen, "Quasiadiabatic continuation of quantum states: The stability of topological ground-state degeneracy and emergent gauge invariance," *Phys. Rev. B*, vol. 72, no. 4, p. 045 141, 2005 (cit. on p. 6).

[24] T. J. Osborne, "Simulating adiabatic evolution of gapped spin systems," *Phys. Rev. A*, vol. 75, no. 3, p. 032 321, 2007 (cit. on p. 6).

[25] E. H. Lieb and D. W. Robinson, "The finite group velocity of quantum spin systems," in *Statistical mechanics*, Springer, 1972, pp. 425–431 (cit. on p. 6).

[26] S. Shalev-Shwartz and S. Ben-David, *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014 (cit. on pp. 7, 8).

[27] S. Mishra and T. K. Rusch, "Enhancing accuracy of deep learning algorithms by training with low-discrepancy sequences," *SIAM Journal on Numerical Analysis*, vol. 59, no. 3, pp. 1811–1834, 2021. DOI: 10.1137/20M1344883. eprint: https://doi.org/10.1137/20M1344883 (cit. on p. 9).

[28] T. De Ryck, S. Lanthaler and S. Mishra, "On the approximation of functions by tanh neural networks," *Neural Networks*, vol. 143, 732–750, Nov. 2021, ISSN: 0893-6080. DOI: 10.1016/j.neunet.2021.08.015. [Online]. Available: http://dx.doi.org/10.1016/j.neunet.2021.08.015 (cit. on p. 9).

[29] T. Lattimore and C. Szepesvári, *Bandit algorithms*. Cambridge University Press, 2020 (cit. on p. 9).

[30] R. Degenne, W. M. Koolen and P. Ménard, "Non-asymptotic pure exploration by solving games," in *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox and R. Garnett, Eds., vol. 32, Curran Associates, Inc., 2019. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2019/file/8d1de7457fa769ece8d93a13a59c8552-Paper.pdf (cit. on p. 11).

[31] P.-A. Wang, R.-C. Tzeng and A. Proutiere, "Fast pure exploration via frank-wolfe," in *Advances in Neural Information Processing Systems*, M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang and J. W. Vaughan, Eds., vol. 34, Curran Associates, Inc., 2021, pp. 5810–5821. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2021/file/2dffbc474aa176b6dc957938c15d0c8b-Paper.pdf (cit. on p. 11).

[32] E. Carlsson, D. Basu, F. Johansson and D. Dubhashi, "Pure exploration in bandits with linear constraints," in *Proceedings of The 27th International Conference on Artificial Intelligence and Statistics*, S. Dasgupta, S. Mandt and Y. Li, Eds., ser. Proceedings of Machine Learning Research, vol. 238, PMLR, 2024, pp. 334–342. [Online]. Available: https://proceedings.mlr.press/v238/carlsson24a.html (cit. on p. 11).

[33] Z. Holmes, K. Sharma, M. Cerezo and P. J. Coles, "Connecting ansatz expressibility to gradient magnitudes and barren plateaus," *PRX Quantum*, vol. 3, no. 1, Jan. 2022, ISSN: 2691-3399. DOI: 10.1103/prxquantum.3.010313. [Online]. Available: http://dx.doi.org/10.1103/PRXQuantum.3.010313 (cit. on pp. 11, 12).

[34] J. van de Wetering and M. Amy, "Optimising quantum circuits is generally hard," *arXiv preprint arXiv:2310.05958*, 2023 (cit. on p. 12).

[35] A. A. Mele et al., *Noise-induced shallow circuits and absence of barren plateaus*, 2024. arXiv: 2403.13927 [quant-ph]. [Online]. Available: https://arxiv.org/abs/2403.13927 (cit. on p. 12).

[36] K. Zhang, L. Liu, M.-H. Hsieh and D. Tao, "Escaping from the barren plateau via gaussian initializations in deep variational quantum circuits," in *Advances in Neural Information Processing Systems*, S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho and A. Oh, Eds., vol. 35, Curran Associates, Inc., 2022, pp. 18 612–18 627. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2022/file/7611a3cb5d733e628081431445cb01fd-Paper-Conference.pdf (cit. on p. 12).

[37] A. D. Bull, "Adaptive-treed bandits," *Bernoulli*, vol. 21, no. 4, pp. 2289–2307, 2015. DOI: 10.3150/14-BEJ644. [Online]. Available: https://doi.org/10.3150/14-BEJ644 (cit. on p. 19).

[38] S. Du, J. Lee, H. Li, L. Wang and X. Zhai, "Gradient descent finds global minima of deep neural networks," in *International conference on machine learning*, PMLR, 2019, pp. 1675–1685 (cit. on p. 21).

[39] C.-F. Chen, A. Anshu and Q. T. Nguyen, "Learning quantum gibbs states locally and efficiently," *arXiv preprint arXiv:2504.02706*, 2025 (cit. on p. 21).

Part II

Appended Papers

Paper I

Predicting Ground State Properties: Constant Sample Complexity and Deep Learning Algorithms

M. Wanner, L. L. Lewis, C. Bhattacharyya, D. Dubhashi, A. Gheorghiu

Advances in Neural Information Processing Systems 37 (NeurIPS 2024)

The paper has been reformatted for uniformity.

https://doi.org/10.52202/079017-1070

Abstract

A fundamental problem in quantum many-body physics is that of finding ground states of local Hamiltonians. A number of recent works gave *provably efficient* machine learning (ML) algorithms for learning ground states. Specifically, Huang et al. in [1], introduced an approach for learning properties of the ground state of an n-qubit gapped local Hamiltonian H from only $n^{\mathcal{O}(1)}$ data points sampled from Hamiltonians in the same phase of matter. This was subsequently improved by Lewis et al. in [2], to $\mathcal{O}(\log n)$ samples when the geometry of the n-qubit system is known. In this work, we introduce two approaches that achieve a *constant* sample complexity, independent of system size n, for learning ground state properties. Our first algorithm consists of a simple modification of the ML model used by Lewis et al. and applies to a property of interest known in advance. Our second algorithm, which applies even if a description of the property is not known, is a deep neural network model. While empirical results showing the performance of neural networks have been demonstrated, to our knowledge, this is the first rigorous sample complexity bound on a neural network model for predicting ground state properties. We also perform numerical experiments on systems of up to 45 qubits that confirm the improved scaling of our approach compared to [1], [2].

1 Introduction

One of the most important problems in quantum many-body physics is that of finding ground states of quantum systems. This is due to the fact that the ground state describes the behavior of electronic systems (e.g., metals, magnets, etc.) at room temperature well. Thus, understanding the ground state can provide insights into, for example, chemical properties of molecules, leading to many potential applications in chemistry and materials science. However, despite extensive research [3], [4], [5], [6], [7], [8], [9], [10], [11], [12], [13], [14], an efficient classical algorithm solving this problem in full generality remains out of reach. On the other hand, researchers have successfully leveraged classical *machine learning* (ML) techniques to solve (albeit largely heuristically) the ground state problem and other related quantum many-body problems [15], [16], [17], [18], [19], [20], [21], [22], [23], [24], [25], [26], [27], [28], [29], [30], [31], [32], [33], [34], [35], [36], [37], [38], [39], [40], [41], [42], [43], [44], [45], [46]. Rather than solving these problems directly from first principles, ML algorithms are given some training data collected from physical experiments and are asked to generalize it to new inputs. Intuitively, this additional data can make the problem easier and thus may open the door to obtaining provably efficient classical ML algorithms for finding ground states. This data-driven approach is in some sense necessary, since finding the ground state from the Hamiltonian alone is known to be QMA-hard in general [47], and thus out of reach for both efficient classical and quantum algorithms.

In a recent work [1], Huang et al. proposed the first *provably efficient* ML algorithm for predicting ground state properties of gapped geometrically local Hamiltonians. In particular, the algorithm in [1] uses an amount of training data (or *sample complexity*) that scales as $\mathcal{O}(n^{1/\epsilon})$, where n is the system size and ϵ is the prediction error of the ML algorithm. Recently, [2] improved this guarantee, achieving $\mathcal{O}(\log(n)2^{\mathrm{polylog}(1/\epsilon)})$, an *exponential improvement* with respect to the system size n. The same sample complexity was obtained by [48] for the task of learning thermal state properties with exponential decay of correlations. Moreover, [49] extended this to Lindbladian phases of matter [50] with local rapid mixing, including both ground states of gapped Hamiltonians and thermal states. The work of [51] obtains a similar guarantee assuming the continuity of quantum states in the parameter range of interest but focusing on the scaling with respect to $1/\epsilon$ rather than system size.

These previous works drastically improve the sample complexity of the original Huang et al. result [1], but none prove sample complexity *lower bounds* for their respective tasks, leaving open the possibility of further reducing the sample complexity. In addition, [2], [48], [49] all use fairly simple learning models, i.e., regularized linear regression and taking empirical averages of classical shadows [52], respectively. With the emergence of neural networks as a popular model in practical ML, one may wonder if these more powerful ML tools may be useful to predict ground state properties as well. In fact, recent works [36], [37] empirically demonstrate a favorable sample complexity using neural-network-based ML algorithms. However, there are currently no rigorous theoretical guarantees regarding the amount of training data needed to achieve

a desired prediction error. These remarks lead us to the following two central questions of this work.

Question 1. *Can classical ML algorithms predict ground state properties with even less than* $\mathcal{O}(\log(n)2^{\mathrm{polylog}(1/\epsilon)})$ *data?*

This is especially relevant for systems approaching the thermodynamic limit, where the system size can be arbitrarily large. Needing fewer samples also means less work for experimentally preparing ground states of the system. The second question, stated as an open question in [1], [2] is:

Question 2. *Can we obtain rigorous sample complexity guarantees for neural-network-based ML algorithms for predicting ground state properties?*

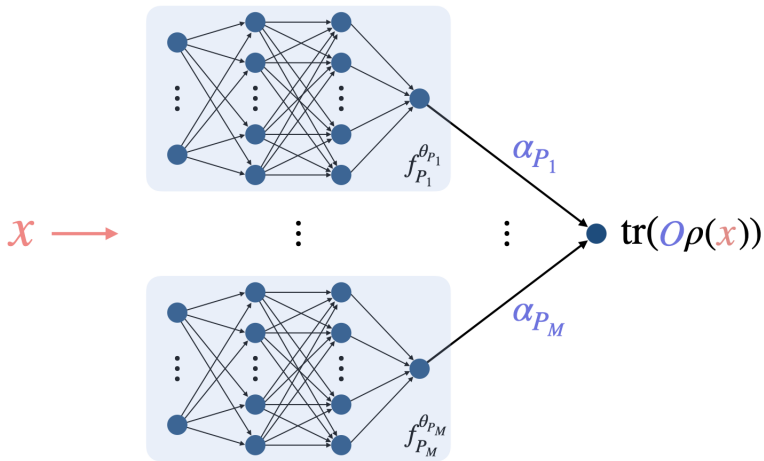

Figure 4.1: **A deep network model for predicting ground state properties.** Given a vector $x \in [-1,1]^m$ that parameterizes a quantum many-body Hamiltonian $H(x)$, the algorithm uses geometric structure to create "local" neural network models $f_{P_i}^{\theta_{P_i}}$. The ML algorithm then combines the outputs of these local models to predict a property $\mathrm{Tr}[O\rho(x)]$, where $\rho(x)$ is the ground state of $H(x)$. Here, we decompose $O = \sum_{i=1}^{M} \alpha_{P_i} P_i$ for Pauli operators P_i, where the final layer takes a linear combination of the outputs of the local models weighted by some trainable parameters w_{P_i} that intuitively should approximate the Pauli coefficients α_{P_i}.

We give positive answers to both questions. We consider the same assumptions as [2] with minimal additional ones that we mention here. First, we show that a simple modification to the approach in [2] allows us to achieve a sample complexity that is *independent of the system size*. This does, however require knowledge of the property we wish to predict in advance, whereas this is not a requirement in [2]. We view this as a reasonable assumption, since in practice we can imagine preparing ground states of some system in order to measure a specific property of interest. More formally, we show the following.

Theorem 1.1 (Informal)**.** *Let $H(x)$ be an n-qubit gapped, geometrically local Hamiltonian with ground state $\rho(x)$. Given an observable O, with a known decomposition as a sum of local Pauli operators and given training data $\{(x_\ell, y_\ell)\}_{\ell=1}^N$ sampled from an arbitrary distribution, with $y_\ell \approx \mathrm{Tr}[O\rho(x_\ell)]$, there is an ML algorithm for predicting ground state properties $\mathrm{Tr}[O\rho(x)]$ to within precision $\epsilon > 0$ using $N = \mathcal{O}\left(2^{\mathrm{polylog}(1/\epsilon)}\right)$ training samples.*

Note that the number of samples N depends only on the desired prediction error ϵ and is independent of the system size. In particular, this means that for a fixed prediction error our algorithm requires only a *constant* amount of training data. Moreover, the computational complexity of our algorithm improves upon [2], having $\mathcal{O}(n)$ runtime, compared to the previous $\mathcal{O}(n \log n)$. While removing the $\log n$ factor may seem like a small improvement, in practice this can make a significant difference. For instance, for a system of $n \sim 1000$ qubits, removing the $\log n$ factor would result in a ten-fold reduction in training data and time.

We achieve this using techniques from [2] and replacing ℓ_1-regularized linear regression [53], [54], [55] with ridge regression [55], [56]. Much like in [2], this result also extends to learning classical representations of $\rho(x)$. In other words, if the algorithm is instead given *classical shadows* [52] of the ground state as training data, it can then predict a classical representation of $\rho(x)$ for new parameters x. This can mitigate the requirement that the observable is known in Theorem 1.1, as predicting properties from a classical representation clearly requires knowledge of the observable.

In the statement of Theorem 1.1, we suppressed several details (such as the fact that x and x_ℓ should be sampled from the same distribution) to give a high level description of the result. The formal statement, including all the assumptions required for proving the result, and the corollary regarding learning classical shadows can be found in Section B.

Our second result shows the same sample complexity guarantee for a *neural network* ML algorithm (Figure 4.1) [57], in which one does not need to know the observable being measured in advance. An additional constraint that we require in this case is that the training data is not sampled according to an arbitrary distribution, but a distribution satisfying some technical assumptions (discussed in Section C.3). With this caveat, we show the following.

Theorem 1.2 (Informal)**.** *Let $H(x)$ be an n-qubit gapped, geometrically local Hamiltonian with ground state $\rho(x)$. For any observable O, expressible as a sum of local Pauli operators and given training data $\{(x_\ell, y_\ell)\}_{\ell=1}^N$, sampled from a distribution satisfying certain assumptions with $y_\ell \approx \mathrm{Tr}[O\rho(x_\ell)]$, there is a neural network ML algorithm for predicting ground state properties $\mathrm{Tr}[O\rho(x)]$, for uniform x, to within precision $\epsilon > 0$ using $N = \mathcal{O}\left(2^{\mathrm{polylog}(1/\epsilon)}\right)$ training samples under mild assumptions on training.*

We prove this result by making use of the Koksma-Hlawka inequality from quasi-Monte Carlo theory [58], [59], [60], [61], [62] and combining it with the spectral flow formalism [63], [64], [65]. The formal statement and its proof can be found in Section C. Similar to Theorem 1.1, we can also extend this

result to learning classical representations of $\rho(x)$ when given classical shadow training data. Furthermore, we perform numerical experiments on system sizes of up to 45 qubits, which support our theoretical findings, and show that, in practice, our deep learning algorithm outperforms previous methods [2].

We also remark that, much like the setting in [51], a more favorable scaling with respect to ϵ can be achieved if the number of parameters that the Hamiltonian depends on is constant. In other words, for the Hamiltonian $H(x)$ with $x \in [-1,1]^m$, it was shown in [51] that if $m = \mathcal{O}(1)$ the sample complexity scales as $N = \mathsf{poly}(1/\epsilon, \log(n))$. For our results, taking m to be constant yields $N = \mathsf{poly}(1/\epsilon)$, preserving the independence on the system size while also achieving a polynomial scaling in $1/\epsilon$.

2 Preliminaries

2.1 Problem statement

First, we formally describe the problem setting, which is the same as [2]. We consider a family of n-qubit Hamiltonians $H(x)$ smoothly parameterized by an m-dimensional vector $x \in [-1,1]^m$. We assume that these Hamiltonians are gapped for all choices of parameters $x \in [-1,1]^m$ and geometrically local such that they can be written as a sum of local terms

$$H(x) = \sum_{j=1}^{L} h_j(\vec{x}_j), \tag{2.1}$$

where the parameter vector x is a concatenation of the constant-dimensional vectors $\vec{x}_1, \ldots, \vec{x}_L$. Each of these constant-dimensional vectors $\vec{x}_j$ parameterizes the local interaction term $h_j(\vec{x}_j)$. Crucially, we assume that each local term h_j only depends on a constant number of parameters rather than the entire parameter vector x. We also assume that the underlying geometry of the n-qubit system is known.

Throughout this work, we use $\rho(x)$ to denote the ground state of the Hamiltonian $H(x)$ and O to denote an observable that can be written as a sum of geometrically observables with bounded spectral norm $\|O\|_\infty \leq 1$. Here, the ground states $\rho(x)$ form a gapped quantum phase of matter. Given samples of quantum states drawn from this phase, we wish to predict expectation values of observables O with respect to other states in the same phase. In other words, we are given training data $\{(x_\ell, y_\ell)\}_{\ell=1}^N$, where $y_\ell \approx \mathrm{Tr}[O\rho(x_\ell)]$ approximates the ground state property for a parameter choice $x_\ell \in [-1,1]^m$ sampled from some distribution $\mathcal{D}$ over the parameter space. We aim to learn a function $h^*(x)$ that approximates the ground state property $\mathrm{Tr}[O\rho(x)]$ for some unseen parameter x while minimizing the amount of training data, or sample complexity, N. How well we learn the ground state property is quantified by the average prediction error

$$\mathop{\mathbb{E}}_{x\sim\mathcal{D}} |h^*(x) - \mathrm{Tr}[O\rho(x)]|^2 \leq \epsilon. \tag{2.2}$$

For our first result described in Theorem 1.1, we also assume that the observable O is known. In practice, a scientist often has a specific ground state property in mind that they wish to study, so we view this as a natural assumption. Moreover, this is still an interesting learning problem, as when obtaining the training data via quantum experiments, preparing the ground state $\rho(x)$ in the laboratory for a new choice of parameters x may be difficult experimentally. This in turn means that accurately predicting some property $\mathrm{Tr}[O\rho(x)]$ for a new choice of x may be challenging, even if the property of interest, O, is known. ML algorithms can allow us to circumvent this issue and generalize from the results of few training data points without needing to prepare the ground state directly.

For our second result in Theorem 1.2, the hypothesis function $h^*(x)$ is computed via a neural network. We require the following additional assumptions. First, we require that all mixed first order derivatives of Hamiltonian

$$\left\| \frac{\partial^m}{\partial x_1 \dots \partial x_m} H(x) \right\|_\infty \leq 1 \tag{2.3}$$

exist and are bounded. This is not much stronger than [2], which assumes that directional derivatives $\partial h_j / \partial \hat{u}$ are bounded by one for any direction $\hat{u}$. Moreover, we also need the training data to be sampled from a distribution $\mathcal{D}$ with probability density function g satisfying the following assumptions: g has full support and is continuously differentiable on $[-1,1]^m$. Also, g is of the form

$$g(x) = \prod_{j=1}^{L} g_j(\vec{x}_j). \tag{2.4}$$

This resembles our assumption on the form of the Hamiltonian $H(x)$. Furthermore, the average prediction error is measured with respect to the same distribution $\mathcal{D}$. Unlike our previous result, in this case, we no longer require knowledge of the observable O.

2.2 Review of previous algorithm

In this section, we review the previous algorithm from [2], as our proofs rely on similar ideas. For full details, we refer the reader to [2] and our more detailed presentation in Section A.1.

The ML algorithm is given a training data set $\{(x_\ell, y_\ell)\}_{\ell=1}^N$, where x_ℓ is sampled from some distribution $\mathcal{D}$ over the parameter space $[-1,1]^m$ and y_ℓ approximates the ground state property: $|y_\ell - \mathrm{Tr}[O\rho(x_\ell)]| \leq \epsilon$. The goal is to learn some function $h^*(x)$ that achieves a low average prediction error

$$\mathop{\mathbb{E}}_{x\sim\mathcal{D}} |h^*(x) - \mathrm{Tr}[O\rho(x)]|^2 \leq \epsilon. \tag{2.5}$$

The ML algorithm proposed in [2] requires several geometric definitions. We use $S^{(\mathrm{geo})}$ to denote the set of all geometrically local Pauli observables throughout. Fix a geometrically local Pauli observable $P \in S^{(\mathrm{geo})} \subseteq \{I, X, Y, Z\}^{\otimes n}$.

Let $\delta_1, \delta_2, B > 0$ be efficiently-computable hyperparameters that we define in Section A.1. Define the set I_P of coordinates c such that x_c parameterizes some local term $h_{j(c)}$ that is close to the Pauli P. Here, the distance between two observables d_{obs} is defined as the minimum distance between the qubits that the observables act on, where the distance between qubits is given by the geometry of the system, which we assume to be known. Formally, we define this set of local coordinates as

$$I_P \triangleq \{c \in \{1, \ldots, m\} : d_{\text{obs}}(h_{j(c)}, P) \leq \delta_1\}. \tag{2.6}$$

The intuition behind this set of coordinates is that it indexes the parameters x_c that influence the ground state property $\text{Tr}[P\rho(x)]$ corresponding to the Pauli P. Using this intuition, because these parameters x_c for $c \in I_P$ are most influential for the property we are trying to learn, we can define a new effective parameter space in which all other parameters are set to zero. Moreover, it even suffices to discretize the parameters x_c for $c \in I_P$ as points on a lattice. This gives the set X_P defined as

$$X_P \triangleq \left\{ \begin{array}{l} x \in [-1,1]^m : \text{if } c \notin I_P, x_c = 0 \\ \qquad\qquad\quad \text{if } c \in I_P, x_c \in \{0, \pm\delta_2, \pm 2\delta_2, \ldots, \pm 1\} \end{array} \right\}. \tag{2.7}$$

For each vector $x \in X_P$, we can also define a set $T_{x,P}$, which is the set of parameters x' that are close to x for coordinates in I_P:

$$T_{x,P} \triangleq \left\{ x' \in [-1,1]^m : -\frac{\delta_2}{2} < x_c - x'_c \leq \frac{\delta_2}{2},\ \forall c \in I_P \right\}. \tag{2.8}$$

The ML algorithm from [2] utilizes these objects to encode the geometric locality of the system. The algorithm consists of two steps. First, it maps the parameter space $[-1,1]^m$ to a high dimensional space $\mathbb{R}^{m_\phi}$ for

$$m_\phi \triangleq \sum_{P \in S^{(\text{geo})}} |X_P| \tag{2.9}$$

via a nonlinear feature map ϕ. Second, it runs ℓ_1-regularized linear regression (LASSO) [53], [54], [66] over the feature space.

This first step encodes the geometry of the problem. In particular, the feature map is defined as follows, where each coordinate of $\phi(x)$ is indexed by $x' \in X_P$ and $P \in S^{(\text{geo})}$

$$\phi(x)_{x',P} \triangleq \mathbb{1}[x \in T_{x',P}]. \tag{2.10}$$

In this way, the feature map $\phi(x)$ identifies the nearest lattice point to x. The idea is that one can approximate the ground state property well by only approximating it at these representative points and summing over all $P \in S^{(\text{geo})}, x' \in X_P$.

Following the feature mapping, the ML algorithm uses LASSO [53], [54], [66] to learn functions of the form $\{h(x) = \mathbf{w} \cdot \phi(x) : \|w\|_1 \leq B\}$ for a chosen hyperparameter $B > 0$. We denote the learned function by $h^*(x) = \mathbf{w}^* \cdot \phi(x)$.

For our purposes, we set $B = 2^{\mathcal{O}(\text{polylog}(1/\epsilon_1))}$. This algorithm obtains the following rigorous guarantee.

Theorem 2.1 (Theorem 1 in [2]). *Given $n, \delta > 0$, $\frac{1}{e} > \epsilon > 0$ and a training data set $\{(x_\ell, y_\ell)\}_{\ell=1}^N$ of size*

$$N = \log(n/\delta)2^{\text{polylog}(1/\epsilon)}, \tag{2.11}$$

where x_ℓ is sampled from an unknown distribution $\mathcal{D}$ and $|y_\ell - \text{Tr}[O\rho(x_\ell)]| \leq \epsilon$ for any observable O with eigenvalues between -1 and 1 that can be written as a sum of geometrically local observables. With a proper choice of the efficiently computable hyperparameters δ_1, δ_2, and B, the learned function $h^*(x) = \mathbf{w}^* \cdot \phi(x)$ satisfies

$$\mathop{\mathbb{E}}_{x\sim\mathcal{D}} |h^*(x) - \text{Tr}[O\rho(x)]|^2 \leq \epsilon \tag{2.12}$$

with probability at least $1 - \delta$. The training and prediction time of the classical ML model are bounded by $\mathcal{O}(nN) = n\log(n/\delta)2^{\text{polylog}(1/\epsilon)}$.

A crucial step in the proof of this result is that ground state properties can indeed be approximated by linear functions over the feature space. Along the way, [2] prove that the ground state property can be approximated by a linear combination of "local functions," which are local in the sense that they only depend on parameters with coordinates in the set I_P. For further details, we refer the reader to Section A.1 and [2].

3 Main results

In this section, we discuss our rigorous guarantees for predicting ground state properties with constant sample complexity and with neural-network-based ML algorithms.

3.1 Constant sample complexity

In this section, we show that a simple modification of the algorithm from [2] can achieve a sample complexity that is independent of the system size n, under the additional assumption that the observable O is known. Let $O = \sum_{P\in\{I,X,Y,Z\}^{\otimes n}} \alpha_P P$ be an observable that can be written as a sum of geometrically local observables. Because O is assumed to be known, we can find this decomposition of O in terms of the Pauli observables P.

The overall structure of the algorithm remains the same: perform a nonlinear feature mapping followed by linear regression. Moreover, we use the same geometric definitions reviewed in the previous section. However, there are two key differences from the previous algorithm [2]. First, we change the feature mapping of [2]. Consider a feature mapping $\tilde{\phi} : [-1, 1]^m \to \mathbb{R}^{m_\phi}$, for m_ϕ as in Equation (2.9), given by

$$\tilde{\phi}(x)_{x',P} \triangleq \text{sign}(\alpha_P)\sqrt{|\alpha_P|}\mathbb{1}\{x \in T_{x',P}\}, \tag{3.1}$$

where each coordinate of $\phi(x)$ is indexed by $P \in S^{(\mathrm{geo})}, x' \in X_P$. The set $T_{x',P}$ is defined in Equation (2.8). The second difference from [2] is that we use ridge regression [55], [56] instead of ℓ_1-regularized regression [53], [54], [55]. Recall that ℓ_1-regularized regression learns hypothesis functions of the form $\{h(x) = \mathbf{w} \cdot \tilde{\phi}(x) : \|\mathbf{w}\|_1 \leq B\}$ for some hyperparameter $B > 0$. In contrast, ridge regression replaces the ℓ_1-norm constraint $\|\mathbf{w}\|_1 \leq B$ with an ℓ_2-norm constraint: $\|\mathbf{w}\|_2 \leq \Lambda$, for some hyperparameter $\Lambda > 0$. Namely, for a chosen efficiently-computable hyperparameter $\Lambda > 0$, ridge regression finds a vector $\mathbf{w}^*$ that minimizes the training error subject to the constraint that $\|\mathbf{w}\|_2 \leq \Lambda$, i.e.,

$$\min_{\substack{\mathbf{w}\in\mathbb{R}^{m_\phi} \\ \|\mathbf{w}\|_2 \leq \Lambda}} \frac{1}{N} \sum_{\ell=1}^{N} |\mathbf{w} \cdot \tilde{\phi}(x_\ell) - y_\ell|^2, \tag{3.2}$$

Standard results in machine learning theory give sample complexity upper bounds for ridge regression in terms of Λ and the ℓ_2-norm of the feature vector $\tilde{\phi}(x)$ [55], [56]. The key idea is that by defining the feature map as in Equation (3.1), we can still approximate the ground state property by a linear function over the feature space, as in [2], to obtain a low training error. Meanwhile, by incorporating the Pauli coefficients α_P into the feature map, we can bound the ℓ_2-norm of $\tilde{\phi}(x)$ by a quantity independent of system size, leveraging bounds on the ℓ_1-norm of the Pauli coefficients [2], [67]. We note that naively applying ridge regression with the feature map from [2] does not achieve the same guarantees and in fact gives worse scaling than [2]. Similarly, we can also choose a suitable $\Lambda > 0$ independent of system size. Thus, we obtain the following guarantee.

Theorem 3.1 (Constant sample complexity)**.** *Given $n, \delta > 0$, $1/e > \epsilon > 0$ and a training data set $\{(x_\ell, y_\ell)\}_{\ell=1}^N$ of size*

$$N = \log(1/\delta) 2^{\mathrm{polylog}(1/\epsilon)}, \tag{3.3}$$

where x_ℓ is sampled from an unknown distribution $\mathcal{D}$ and $|y_\ell - \mathrm{Tr}[O\rho(x_\ell)]| \leq \epsilon$ for any observable O with eigenvalues between -1 and 1 that can be written as a sum of geometrically local observables. With a proper choice of the efficiently computable hyperparameters $\delta_1, \delta_2, \Lambda$, the learned function $h^*(x)$ satisfies

$$\mathop{\mathbb{E}}_{x\sim\mathcal{D}} |h^*(x) - \mathrm{Tr}[O\rho(x)]|^2 \leq \epsilon \tag{3.4}$$

with probability least $1 - \delta$. The training and prediction time of the classical ML model are bounded by $\mathcal{O}(n)\mathrm{polylog}(1/\delta)2^{\mathrm{polylog}(1/\epsilon)}$.

We compare this result to Theorem 2.1. For a constant prediction error $\epsilon = \mathcal{O}(1)$, our proposed algorithm achieves a constant sample complexity $N = \mathcal{O}(1)$, compared to the logarithmic sample complexity $N = \mathcal{O}(\log n)$ of [2]. Moreover, the computational complexity of our algorithm improves upon [2], achieving a linear-in-n runtime, compared to the previous $\mathcal{O}(n \log n)$. The scaling with respect to the prediction error ϵ is the same as the previous algorithm [2]. This means that regardless of how large our quantum system is,

we need the same amount of samples to predict ground state properties well. This is especially important for settings in which obtaining training data for large systems is difficult.

Thus far, we have only considered the setting in which we learn a specific ground state property $\mathrm{Tr}[O\rho(x)]$ for a fixed observable O. Because our training data is given in the form $\{(x_\ell, y_\ell)\}_{\ell=1}^N$, where y_ℓ approximates $\mathrm{Tr}[O\rho(x)]$ for this fixed observable O, if we want to predict a new property for the same ground state $\rho(x)$, we would need to generate new training data. Thus, it may be more useful to learn a ground state representation, from which we could predict $\mathrm{Tr}[O\rho(x)]$ for many different choices of observables O without requiring new training data. In this case, suppose we are instead given training data $\{x_\ell, \sigma_T(\rho(x_\ell))\}_{\ell=1}^N$, where $\sigma_T(\rho(x_\ell))$ is a classical shadow representation [52], [68], [69], [70], [71] of the ground state $\rho(x_\ell)$. An immediate corollary of Theorem B.1 is that we can predict ground state representations with the same sample complexity. This follows from the same proof as Corollary 5 in [2].

Corollary 3.2 (Learning representations of ground states)**.** *Let $n, \delta > 0$, $1/e > \epsilon > 0$ and $\delta > 0$. Given training data $\{(x_\ell, \sigma_T(\rho(x_\ell))\}_{\ell=1}^N$ of size*

$$N = \log(1/\delta) 2^{\mathcal{O}(\mathrm{polylog}(1/\epsilon))}, \tag{3.5}$$

where x_ℓ is sampled from $\mathcal{D}$ and $\sigma_T(\rho(x_\ell)$ is the classical shadow representation of the ground state $\rho(x_\ell)$ using T randomized Pauli measurements. For $T = \tilde{\mathcal{O}}(\log(n/\delta)/\epsilon^2)$, with probability at least $1-\delta$, the ML algorithm will produce a ground state representation $\hat{\rho}_{N,T}(x)$ that achieves

$$\mathop{\mathbb{E}}_{x\sim\mathcal{D}} |\mathrm{Tr}[O\hat{\rho}_{N,T}(x)] - \mathrm{Tr}[O\rho(x)]|^2 \leq \epsilon \tag{3.6}$$

for any observable with eigenvalues between -1 and 1 that can be written as a sum of geometrically local observables.

3.2 Rigorous guarantees for neural networks

In this section, we prove the existence of a deep neural network model that can predict ground state properties using a constant number of training samples. In particular, we prove that after training on a constant number of samples from a distribution $\mathcal{D}$ on $[-1,1]^m$ satisfying certain technical assumptions, our model can achieve a low prediction error under mild assumptions on training. In this case, for predicting properties $\mathrm{Tr}[O\rho(x)]$, the observable O need not be known in advance. However, we need to assume that all mixed first order derivatives of the Hamiltonian

$$\left\| \frac{\partial^m}{\partial x_1 \dots \partial x_m} H(x) \right\|_\infty \leq 1 \tag{3.7}$$

exist and are bounded. Moreover, the distribution $\mathcal{D}$ has probability density function g satisfying the following assumptions: g has full support, is continuously differentiable on $[-1,1]^m$, and is of the form

$$g(x) = \prod_{j=1}^L g_j(\vec{x}_j). \tag{3.8}$$

As in the previous algorithm [2], we leverage the geometry of the n-qubit system to approximate the ground state properties by a linear combination of smooth local functions, which are local in the sense that they only depend on parameters with coordinates in the local coordinate set I_P defined in Equation (2.6). Crucially, the size $\tilde{m} \triangleq |I_P|$ of the domains of these local functions is independent of the system size.

However, instead of using a feature map and linear regression to learn the ground state properties, we utilize a deep neural network model defined as follows. Inspired by the local approximation of ground state properties, we define "local models" $f_P^{\theta_P} : [-1,1]^{\tilde{m}} \to \mathbb{R}$, which are neural networks consisting of three layers of affine transformations and applications of a nonlinear activation function. In particular, $f_P^{\theta_P}$ has two hidden layers with the affine transformations given by the trainable weights and biases denoted by θ_P. We take hyperbolic tangent, tanh, as the activation function. These local models are then combined into a model $f^{\Theta,w} : [-1,1]^m \to \mathbb{R}$ given by

$$f^{\Theta,w}(x) = \sum_{P \in S^{(\mathrm{geo})}} w_P f_P^{\theta_P}(x), \tag{3.9}$$

where $w_P \in \mathbb{R}$ are the weights in the last layer and $\Theta = \{\theta_P\}_{P \in S^{(\mathrm{geo})}}$. This model is schematically illustrated in Figure 4.1. We refer to Definition C.1 in Section C for a full description of the model.

Consider training data $\{(x_\ell, y_\ell)\}_{\ell=1}^N$, where x_ℓ are sampled according to a distribution $\mathcal{D}$ satisfying the assumptions described above and $|y_\ell - \mathrm{Tr}[O\rho(x_\ell)]| \leq \epsilon$. The ML algorithm first initializes the weights via standard deep learning initialization procedures, e.g., Xavier initialization [72]. Then, the algorithm performs quasi-Monte Carlo training given the training data, (e.g., Adam [73]), to find weights Θ^*, w^* which minimize the training objective function

$$\frac{1}{N} \sum_{\ell=1}^{N} |f^{\Theta,w}(x_\ell) - y_\ell|^2 + \lambda \|w\|_1, \tag{3.10}$$

where λ is some regularization parameter that may depend on ϵ.

For this algorithm, we prove the following theorem bounding the average prediction error of our deep neural network model.

Theorem 3.3 (Neural network sample complexity guarantee)**.** *Let $1/e > \epsilon > 0$. Let $\mathcal{D}$ be a distribution with probability density function g satisfying the following properties: g has full support, is continuously differentiable, and is of the form*

$$g(x) = \prod_{j=1}^{L} g_j(\vec{x}_j). \tag{3.11}$$

Let $f^{\Theta^*,w^*} : [-1,1]^m \to \mathbb{R}$ be a neural network model trained on data $\{(x_\ell, y_\ell)\}_{\ell=1}^N$ of size

$$N = \mathcal{O}\left(2^{\mathrm{polylog}(1/\epsilon)}\right), \tag{3.12}$$

where the x_ℓ's are sampled from $\mathcal{D}$ and $|y_\ell - \mathrm{Tr}[O\rho(x_\ell)]| \leq \epsilon$. Suppose that f^{Θ^*,w^*} achieves a value no larger than $\mathcal{O}(\epsilon)$ on the training objective (Equation (3.10)) with $\lambda(\epsilon) = \mathcal{O}(\epsilon)$. Additionally, suppose that all parameters Θ_i^* of f^{Θ^*,w^*} satisfy $|\Theta_i^*| \leq W_{\max}$, for some $W_{max} > 0$ that is independent of n. Then

$$\mathop{\mathbb{E}}_{x\sim\mathcal{D}} |f^{\Theta^*,w^*}(x) - \mathrm{Tr}[O\rho(x)]|^2 \leq \epsilon. \tag{3.13}$$

Similar to Theorem 3.1, for a constant prediction error $\epsilon = \mathcal{O}(1)$, the deep neural network algorithm achieves constant sample complexity $N = \mathcal{O}(1)$. In contrast to Theorem 3.1, we do not require knowledge about the observable, O. This is a direct consequence of the regularity of w_P, which is achieved when the training objective is small. Theorem 3.5 guarantees, that a model with such regularity can yield a small prediction error.

There are, however, some caveats compared to the previous result. First, the training data is restricted to being sampled from a distribution satisfying our technical assumptions stated in the theorem, in contrast to Theorem 3.1 which holds for data sampled from any arbitrary unknown distribution. Second, in regards to the model, the weights must be bounded by a constant $W_{\max}$. Finally, we cannot guarantee *a priori* that the network will indeed achieve a low training error. This is due to the fact that our training objective is non-convex and thus, globally optimal weights cannot be found efficiently in general [74]. Even so, we are still able to prove the existence of suitable weights such that the resulting network approximates $\mathrm{Tr}[O\rho(x)]$ for any $x \in [-1,1]^m$ (see Theorem 3.5 in the next section).

However, we view the assumptions made in Theorem 3.3 as being mild in practice. Small training objectives are commonly achieved in deep learning so we expect our training algorithm to produce a model which fulfills the assumptions of Theorem 3.3 after $\mathcal{O}(1)$ training steps and $\mathcal{O}(n)$ runtime. Moreover, it is known that gradient descent provably converges to the global optimum for *overparametrized* deep neural networks, while the weights remain small, when properly initialized [75]. We emphasize this further when discussing the numerical experiments in Figure 4.2 and Section D. To our knowledge, Theorem 3.3 is the first rigorous sample complexity bound on a neural network model for predicting ground state properties.

We also note that if the training data is instead sampled according to a *low-discrepancy sequence* (LDS) [59], [60], [61], [62], [76], [77], [78], [79], [80], [81], we can obtain better guarantees, but these improvements are hidden in the polylogarithmic factors in the exponential and so give effectively the same sample complexity. We discuss learning given data from a low-discrepancy sequence in Section C. Intuitively, this is a collection of points in the parameter space that covers the space in such a way that there are no large gaps, or discrepancies.

Similar to Corollary 3.2, if we are instead given training data $\{x_\ell, \sigma_T(\rho(x_\ell))\}_{\ell=1}^N$, where $\sigma_T(\rho(x_\ell))$ is a classical shadow representation [52], [68], [69], [70], [71] of the ground state $\rho(x_\ell)$, then an immediate corollary of Theorem 3.3 is that we can predict ground state representations with the same sample complexity.

Corollary 3.4 (Learning representations of ground states with neural nets)**.** *Let $n, \delta > 0$, $1/e > \epsilon > 0$ and $\delta > 0$. Given training data $\{(x_\ell, \sigma_T(\rho(x_\ell))\}_{\ell=1}^N$ of size*

$$N = \sqrt{\log(1/\delta)} 2^{\mathcal{O}(\text{polylog}(1/\epsilon))}, \tag{3.14}$$

where x_ℓ is sampled from a distribution $\mathcal{D}$ satisfying the same assumptions as Theorem 3.3 and $\sigma_T(\rho(x_\ell)$ is the classical shadow representation of the ground state $\rho(x_\ell)$ using T randomized Pauli measurements. For $T = \tilde{\mathcal{O}}(\log(n/\delta)/\epsilon^2)$, with probability at least $1 - \delta$, the ML algorithm will produce a ground state representation $\hat{\rho}_{N,T}(x)$ that achieves

$$\mathop{\mathbb{E}}_{x\sim\mathcal{D}} |\text{Tr}[O\hat{\rho}_{N,T}(x)] - \text{Tr}[O\rho(x)]|^2 \leq \epsilon \tag{3.15}$$

for any observable with eigenvalues between -1 and 1 that can be written as a sum of geometrically local observables.

3.2.1 Proof ideas for neural network guarantee

To prove Theorem 3.3, we first show that our neural network model $f^{\Theta,w}$ can approximate the ground state properties well. In particular, we show that there exist weights Θ', w' such that $f^{\Theta',w'}$ approximates the ground state properties and thus achieves small value for the training objective (Equation (3.10)). Then, we bound the prediction error using tools from deep learning and quasi-Monte Carlo theory [58], [59], [60], [61], [62].

We ensure the existence of $f^{\Theta',w'}$ in the following theorem.

Theorem 3.5. *For any $1/e > \epsilon > 0$ and width W, there exist weights Θ', w' such that the neural network model $f^{\Theta',w'}$ satisfies*

$$|f^{\Theta',w'}(x) - \text{Tr}[O\rho(x)]|^2 \leq \epsilon, \tag{3.16}$$

for any $x \in [-1, 1]^m$. Moreover, each parameter Θ_i of the network has a magnitude of $|\Theta_i| = 2^{\mathcal{O}(\text{polylog}(1/\epsilon))}$.

In particular, this implies that for a suitable choice of regularization parameter $\lambda = \mathcal{O}(\epsilon)$, the training objective from Equation (3.10) is also small. We prove this statement by combining results in deep learning about tanh neural networks approximating functions [57] with the geometric locality of the system and smoothness of the ground state properties. We note that the weights Θ', w' in Theorem 3.5 are not necessarily the weights Θ^*, w^* found via the neural network training procedure in Theorem 3.3. Because the training objective is non-convex, we cannot guarantee convergence to these weights Θ', w'. However, assuming that f^{Θ^*,w^*} does indeed achieve a low training error (which is often satisfied in practice), we are able to rigorously guarantee that the model will generalize well and achieve a low prediction error in Theorem 3.3.

Notice that the guarantee of Theorem 3.5 holds for all x and, in particular, does not require our assumptions on the distribution $\mathcal{D}$. The assumption that the network is trained on such data only becomes relevant when bounding the prediction error. While not explicitly stated here, we also note that Theorem 3.5 gives a bound on the number of trainable parameters $|\Theta_i|$ that

has a similar dependence on ϵ as the model in [2]. Furthermore, the parameters are independent of system size, n. Additional smoothness assumptions on the Hamiltonian $H(x)$ can yield mild improvements on the dependence in terms of ϵ, as briefly discussed in Section C.1. Moreover, because of this bound on $|\Theta_i|$, applying an additional penalty on the ℓ_2-norm of the weights Θ can help ensure that the weights remain small. In practice, this is usually satisfied during training when the weights are initialized properly and the inputs are regularized. This was observed, for example, in [57]. Thus, the condition that $|\Theta_i^*| \leq W_{\max}$ is often satisfied in practice and is not considered a strong assumption in deep learning.

To prove the prediction error bound in Theorem 3.3 assuming that a low training error is achieved, we combine techniques from quasi-Monte Carlo theory applied to deep learning [58] (see Section A.2 for a review) along with our knowledge of the geometry of the n-qubit system. In contrast to [58], we need to characterize the dimension of the input domain in our approach. The reason for doing this is that the approximation error depends on the size $\tilde{m} = |I_P|$ of I_P (Equation (2.6)) of our local models $f_P^{\theta_P}$.

The central result we use here is the Koksma-Hlawka inequality [60] (see Theorem A.12 in Section A.2) from quasi-Monte Carlo theory. This produces a bound on the prediction error in terms of the star-discrepancy (see Definition A.8 in Section A.2) and the Hardy-Krause variation. The star-discrepancy can be controlled by known bounds on the star-discrepancy of random points [82]. We bound the Hardy-Krause variation by explicitly computing the mixed derivatives of the local models $f_P^{\theta_P}$ and the ground state properties $\mathrm{Tr}[O\rho(x)]$. In particular, we bound the latter using tools from the spectral flow formalism [63], [64], [65], and this is where the assumption in Equation (3.7) is needed. Putting these steps together, we arrive at the rigorous guarantee in Theorem 3.3.

4 Numerical experiments

We conduct numerical experiments to observe the performance of our neural network model in practice. The results demonstrate that our assumptions in Theorem 3.3 are often satisfied in practice and that our deep learning algorithm outperforms the previous best-known method [2]. The code can be found at `https://github.com/marcwannerchalmers/learning_ground_states.git`.

We consider the classical neural network model discussed in the previous section and defined formally in Definition C.1. For each of the local models $f_P^{\theta_P}$, we use fully connected deep neural networks with five hidden layers of width 200. We train the model with the AdamW optimization algorithm [83]. We measure the training error and prediction error via the root-mean-square error (RMSE). The model is discussed further in Section D.

As in [2], we consider the two-dimensional antiferromagnetic random Heisenberg model with spin-1/2 particles placed on sites in a two-dimensional lattice. The corresponding Hamiltonian is

$$H = \sum_{\langle ij \rangle} J_{ij}(X_iX_j + Y_iY_j + Z_iZ_j), \tag{4.1}$$

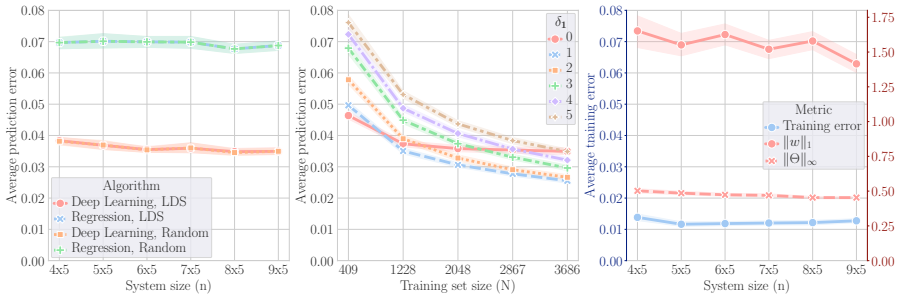

Figure 4.2: **Numerical experiments. (Left)** Comparison with previous methods. Each point indicates the prediction error (RMSE) of our deep learning model or the regression model of [2], fixing the training set size $N = 3686$ and the size of the local neighorbood $\delta_1 = 0$ (Equation (2.6)). We train both algorithms on either LDS or uniformly random points, which achieve similar performance. **(Center)** Scaling with training size. Each point indicates the prediction error of our deep learning model given LDS training data for various δ_1 and training data sizes. **(Right)** Neural network weights and training error. Blue points correspond to the training error of the neural network model. Red points correspond to the ℓ_1 norm of parameters in the last layer or the largest absolute value of the parameters of the neural network, fixing $N = 3686$ and $\delta_1 = 1$. This shows that the assumptions in Theorem 3.3 are achieved in practice. The shaded areas denote the 1-sigma error bars across the assessed ground state properties.

where $\langle ij \rangle$ denotes all pairs of neighboring sites on the lattice. We consider lattice sizes of $4 \times 5 = 20$ up to $9 \times 5 = 45$. The coupling terms J_{ij} correspond to the parameters x of the Hamiltonian and are sampled uniformly from $[0, 2]$ (and then mapped to lie in $[-1, 1]$ for our ML algorithm). The goal of the numerical experiments is to predict two-body correlation functions, i.e., the expectation value of

$$C_{ij} = \frac{1}{3}(X_i X_j + Y_i Y_j + Z_i Z_j) \tag{4.2}$$

for all neighboring sites $\langle ij \rangle$.

To this end, we generate training data similarly to [1], [2], using the density-matrix renormalization group (DMRG) [9] based on matrix-product-states (MPS) [84]. In this way, we approximate the ground state and corresponding correlation functions for the Hamiltonian in Equation (4.1) for different choices of coupling parameters J_{ij}. Our ML model is then trained on this data and predicts two-body correlation functions for an unseen choice of coupling parameters. To assess the performance of our model, we consider both uniformly randomly distributed J_{ij} and coupling parameters, which are distributed as a low-discrepancy sequence (Sobol sequence).

In Figure 4.2 (Left), we see that our deep learning algorithm consistently outperforms the previous best-known ML algorithm from [2], achieving ap-

proximately half the prediction error on the same training data. We also observe that training on LDS points or uniformly random points does not significantly alter the performance of either ML algorithm. The prediction error also exhibits a constant scaling with respect to system size, agreeing with our rigorous guarantee Theorem 3.3. Another noteworthy observation is that the ML algorithm's performance on LDS is nearly equivalent to its performance on uniformly random points. We attribute this to the significant impact of local approximation error when the size of the local neighborhood δ_1 (Equation (2.6)) is small and the rapid increase in the dimensionality of local models as δ_1 increases, which outweighs the practical benefits of using LDS. We discuss this further in Section D.

Figure 4.2 (Center) illustrates the prediction error scaling with respect to the training set size for various choices of δ_1 (size of the local neighborhood from Equation (2.6)). For a choice of $\delta_1 = 0$, the error arising from approximating the ground state property via local functions dominates the prediction error. For $\delta_1 > 0$, we observe a smaller error from local approximations and thus achieve a smaller prediction error when the training set size N is large enough, whereas the generalization error increases with δ_1 for fixed training size N. This is consistent with our theoretical results.

Finally, Figure 4.2 (Right) illustrates that our assumptions in Theorem 3.3 are mild in practice. Namely, the blue points show that a small training error can be achieved. The red points also demonstrate that the ℓ_1-norm of the parameters in the last layer and the largest absolute value of the parmeters in the trained neural network remain small. In particular, in Figure 4.2, the weights exhibit a scaling *independent* of system size n. Hence, we find that the assumptions needed to guarantee the prediction error bound in Theorem 3.3, namely that the training objective is small and the weights of the neural network are small and independent of system size, are fulfilled in our numerical experiments. We provide further details of the numerical experiments in Section D.

5 Discussion

We have shown that we can construct ML models for predicting ground state properties that require only a constant number of training samples, for a fixed prediction error. Specifically, we showed that a simple modification to the linear regression model in [2] only requires $2^{\mathsf{polylog}(\epsilon^{-1})}$ samples in order to achieve a prediction error of ϵ, provided that we know a decomposition of the observable of interest in terms of Pauli operators. We then showed that a neural network model which is trained on $2^{\mathsf{polylog}(\epsilon^{-1})}$ training samples and which achieves $\mathcal{O}(\epsilon)$ training error on these samples will also have a prediction error of at most ϵ. In this case, knowledge of the observable O is no longer required. Furthermore, numerical experiments show that our deep learning algorithm outperforms previous methods [2], and the assumptions in Theorem 3.3 are satisfied in practice.

Our work leaves open several avenues for future exploration. First, it would

be desirable to understand the conditions under which we can prove convergence for the training error. For instance, could we utilize similar techniques as [75], [85] to show convergence to a global optimum in a non-convex landscape? Following [49], [50], we would also like to know whether the results obtained for neural networks can be extended to thermal states or Lindbladian phases of matter. Finally, for both results it would be desirable to improve the scaling with respect to the error ϵ. Currently, the models have quasipolynomial scaling in $1/\epsilon$ and the only case in which we know how to achieve $\mathsf{poly}(1/\epsilon)$ scaling is when the number of parameters, m, is constant (as in [51]).

Acknowledgements

MW and DD are supported by SSF (Swedish Foundation for Strategic Research), grant number FUS21-0063. LL is supported by a Marshall Scholarship. CB is partially supported by a grant from DIA-COE. AG is supported by the Knut and Alice Wallenberg Foundation through the Wallenberg Centre for Quantum Technology (WACQT). This work was done in part while a subset of the authors were visiting the Simons Institute for the Theory of Computing.

Bibliography

[1] H.-Y. Huang, R. Kueng, G. Torlai, V. V. Albert and J. Preskill, "Provably efficient machine learning for quantum many-body problems," *Science*, vol. 377, no. 6613, eabk3333, 2022 (cit. on pp. 1 (I), 1 (I), 2 (I), 2 (I), 2 (I), 3 (I), 15 (I), 81 (I), 81 (I)).

[2] L. Lewis, H.-Y. Huang, V. T. Tran, S. Lehner, R. Kueng and J. Preskill, "Improved machine learning algorithm for predicting ground state properties," *Nature Communications*, vol. 15, no. 1, p. 895, 2024 (cit. on pp. 1 (I), 1 (I), 2 (I), 2 (I), 3 (I), 3 (I), 3 (I), 3 (I), 4 (I), 4 (I), 4 (I), 5 (I), 5 (I), 6 (I), 6 (I), 6 (I), 6 (I), 7 (I), 8 (I), 8 (I), 8 (I), 8 (I), 8 (I), 8 (I), 9 (I), 9 (I), 9 (I), 9 (I), 9 (I), 9 (I), 9 (I), 9 (I), 10 (I), 11 (I), 14 (I), 14 (I), 14 (I), 15 (I), 15 (I), 15 (I), 16 (I), 16 (I), 29 (I), 29 (I), 29 (I), 29 (I), 29 (I), 30 (I), 30 (I), 31 (I), 31 (I), 31 (I), 32 (I), 32 (I), 33 (I), 33 (I), 33 (I), 38 (I), 38 (I), 38 (I), 39 (I), 39 (I), 42 (I), 47 (I), 52 (I), 80 (I), 80 (I), 81 (I), 81 (I), 81 (I), 82 (I), 82 (I)).

[3] P. Hohenberg and W. Kohn, "Inhomogeneous electron gas," *Phys. Rev.*, vol. 136, B864–B871, 3B 1964. DOI: 10.1103/PhysRev.136.B864. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRev.136.B864 (cit. on p. 2 (I)).

[4] W. Kohn, "Nobel lecture: Electronic structure of matter—wave functions and density functionals," *Rev. Mod. Phys.*, vol. 71, pp. 1253–1266, 5 1999. DOI: 10.1103/RevModPhys.71.1253. [Online]. Available: https://link.aps.org/doi/10.1103/RevModPhys.71.1253 (cit. on p. 2 (I)).

[5] A. W. Sandvik, "Stochastic series expansion method with operator-loop update," *Phys. Rev. B*, vol. 59, R14157–R14160, 22 1999. DOI: 10.1103/PhysRevB.59.R14157. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevB.59.R14157 (cit. on p. 2 (I)).

[6] D. Ceperley and B. Alder, "Quantum Monte Carlo," *Science*, vol. 231, no. 4738, pp. 555–560, 1986, ISSN: 0036-8075. DOI: 10.1126/science.231.4738.555. eprint: https://science.sciencemag.org/content/231/4738/555.full.pdf. [Online]. Available: https://science.sciencemag.org/content/231/4738/555 (cit. on p. 2 (I)).

[7] F. Becca and S. Sorella, *Quantum Monte Carlo Approaches for Correlated Systems*. Cambridge University Press, 2017. DOI: 10.1017/9781316417041 (cit. on p. 2 (I)).

[8] J. Gubernatis, N. Kawashima and P. Werner, *Quantum Monte Carlo Methods*. Cambridge University Press, 2016 (cit. on p. 2 (I)).

[9] S. R. White, "Density matrix formulation for quantum renormalization groups," *Physical review letters*, vol. 69, no. 19, p. 2863, 1992 (cit. on pp. 2 (I), 15 (I)).

[10] S. R. White, "Density-matrix algorithms for quantum renormalization groups," *Phys. Rev. B*, vol. 48, no. 14, p. 10 345, 1993 (cit. on p. 2 (I)).

[11] G. Vidal, "Class of quantum many-body states that can be efficiently simulated," *Physical review letters*, vol. 101, no. 11, p. 110 501, 2008 (cit. on p. 2 (I)).

[12] A. Peruzzo et al., "A variational eigenvalue solver on a photonic quantum processor," *Nat. Commun.*, vol. 5, p. 4213, 2014 (cit. on p. 2 (I)).

[13] J. I. Cirac, D. Perez-Garcia, N. Schuch and F. Verstraete, "Matrix product states and projected entangled pair states: Concepts, symmetries, theorems," *Reviews of Modern Physics*, vol. 93, no. 4, p. 045 003, 2021 (cit. on p. 2 (I)).

[14] T. S. Cubitt, "Dissipative ground state preparation and the dissipative quantum eigensolver," *arXiv preprint arXiv:2303.11962*, 2023 (cit. on p. 2 (I)).

[15] G. Carleo et al., "Machine learning and the physical sciences," *Rev. Mod. Phys.*, vol. 91, p. 045 002, 4 2019. DOI: 10.1103/RevModPhys.91.045002. [Online]. Available: https://link.aps.org/doi/10.1103/RevModPhys.91.045002 (cit. on p. 2 (I)).

[16] J. Carrasquilla, "Machine learning for quantum matter," *Adv. Phys.: X*, vol. 5, no. 1, p. 1 797 528, 2020. DOI: 10.1080/23746149.2020.1797528. eprint: https://doi.org/10.1080/23746149.2020.1797528. [Online]. Available: https://doi.org/10.1080/23746149.2020.1797528 (cit. on p. 2 (I)).

[17] D.-L. Deng, X. Li and S. Das Sarma, "Machine learning topological states," *Phys. Rev. B*, vol. 96, p. 195 145, 19 2017. DOI: 10.1103/PhysRevB.96.195145. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevB.96.195145 (cit. on p. 2 (I)).

[18] J. Carrasquilla and R. G. Melko, "Machine learning phases of matter," *Nat. Phys.*, vol. 13, p. 431, 2017. [Online]. Available: https://doi.org/10.1038/nphys4035 (cit. on p. 2 (I)).

[19] G. Carleo and M. Troyer, "Solving the quantum many-body problem with artificial neural networks," *Science*, vol. 355, no. 6325, pp. 602–606, 2017, ISSN: 0036-8075. DOI: 10.1126/science.aag2302 (cit. on p. 2 (I)).

[20] G. Torlai and R. G. Melko, "Learning thermodynamics with Boltzmann machines," *Physical Review B*, vol. 94, no. 16, p. 165 134, 2016. DOI: 10.1103/PhysRevB.94.165134. Accessed: 20th Jan. 2017. [Online]. Available: http://link.aps.org/doi/10.1103/PhysRevB.94.165134 (cit. on p. 2 (I)).

[21] Y. Nomura, A. S. Darmawan, Y. Yamaji and M. Imada, "Restricted boltzmann machine learning for solving strongly correlated quantum systems," *Phys. Rev. B*, vol. 96, p. 205 152, 20 2017. DOI: 10.1103/PhysRevB.96.205152. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevB.96.205152 (cit. on p. 2 (I)).

[22] E. P. L. van Nieuwenburg, Y.-H. Liu and S. D. Huber, "Learning phase transitions by confusion," *Nat. Phys.*, vol. 13, p. 435, 2017. [Online]. Available: https://doi.org/10.1038/nphys4037 (cit. on p. 2 (I)).

[23] L. Wang, "Discovering phase transitions with unsupervised learning," *Phys. Rev. B*, vol. 94, p. 195 105, 19 2016. DOI: 10.1103/PhysRevB.94.195105. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevB.94.195105 (cit. on p. 2 (I)).

[24] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals and G. E. Dahl, "Neural message passing for quantum chemistry," *arXiv preprint arXiv:1704.01212*, 2017 (cit. on p. 2 (I)).

[25] G. Torlai, G. Mazzola, J. Carrasquilla, M. Troyer, R. Melko and G. Carleo, "Neural-network quantum state tomography," *Nat. Phys.*, vol. 14, no. 5, pp. 447–450, 2018. DOI: 10.1038/s41567-018-0048-5. [Online]. Available: https://doi.org/10.1038/s41567-018-0048-5 (cit. on p. 2 (I)).

[26] R. A. Vargas-Hernández, J. Sous, M. Berciu and R. V. Krems, "Extrapolating quantum observables with machine learning: Inferring multiple phase transitions from properties of a single phase," *Physical review letters*, vol. 121, no. 25, p. 255 702, 2018 (cit. on p. 2 (I)).

[27] K. Schütt, M. Gastegger, A. Tkatchenko, K.-R. Müller and R. J. Maurer, "Unifying machine learning and quantum chemistry with a deep neural network for molecular wavefunctions," *Nat. Commun.*, vol. 10, no. 1, pp. 1–10, 2019 (cit. on p. 2 (I)).

[28] I. Glasser, N. Pancotti, M. August, I. D. Rodriguez and J. I. Cirac, "Neural-network quantum states, string-bond states, and chiral topological states," *Phys. Rev. X*, vol. 8, p. 011 006, 1 2018. DOI: 10.1103/PhysRevX.8.011006. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevX.8.011006 (cit. on p. 2 (I)).

[29] M. C. Caro et al., "Out-of-distribution generalization for learning quantum dynamics," *arXiv preprint arXiv:2204.10268*, 2022 (cit. on p. 2 (I)).

[30] J. F. Rodriguez-Nieva and M. S. Scheurer, "Identifying topological order through unsupervised machine learning," *Nat. Phys.*, vol. 15, no. 8, pp. 790–795, 2019 (cit. on p. 2 (I)).

[31] Z. Qiao, M. Welborn, A. Anandkumar, F. R. Manby and T. F. Miller III, "Orbnet: Deep learning for quantum chemistry using symmetry-adapted atomic-orbital features," *J. Chem. Phys.*, vol. 153, no. 12, p. 124 111, 2020 (cit. on p. 2 (I)).

[32] K. Choo, A. Mezzacapo and G. Carleo, "Fermionic neural-network states for ab-initio electronic structure," *Nat. Commun.*, vol. 11, no. 1, p. 2368, May 2020, ISSN: 2041-1723. DOI: 10.1038/s41467-020-15724-9. [Online]. Available: https://doi.org/10.1038/s41467-020-15724-9 (cit. on p. 2 (I)).

[33] H. Kawai and Y. O. Nakagawa, "Predicting excited states from ground state wavefunction by supervised quantum machine learning," *Machine Learning: Science and Technology*, vol. 1, no. 4, p. 045 027, 2020 (cit. on p. 2 (I)).

[34] J. R. Moreno, G. Carleo and A. Georges, "Deep learning the hohenberg-kohn maps of density functional theory," *Physical Review Letters*, vol. 125, no. 7, p. 076 402, 2020 (cit. on p. 2 (I)).

[35] K. Kottmann, P. Corboz, M. Lewenstein and A. Acín, "Unsupervised mapping of phase diagrams of 2d systems from infinite projected entangled-pair states via deep anomaly detection," *SciPost Physics*, vol. 11, no. 2, p. 025, 2021 (cit. on p. 2 (I)).

[36] H. Wang, M. Weber, J. Izaac and C. Y.-Y. Lin, "Predicting properties of quantum systems with conditional generative models," *arXiv preprint arXiv:2211.16943*, 2022 (cit. on pp. 2 (I), 2 (I)).

[37] V. T. Tran et al., "Using shadows to learn ground state properties of quantum hamiltonians," *Machine Learning and Physical Sciences Workshop at the 36th Conference on Neural Information Processing Systems (NeurIPS)*, 2022 (cit. on pp. 2 (I), 2 (I)).

[38] K. Mills, M. Spanner and I. Tamblyn, "Deep learning and the schrödinger equation," *Phys. Rev. A*, vol. 96, p. 042 113, 4 2017. DOI: 10.1103/PhysRevA.96.042113. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevA.96.042113 (cit. on p. 2 (I)).

[39] N Saraceni, S Cantori and S Pilati, "Scalable neural networks for the efficient learning of disordered quantum systems," *Physical Review E*, vol. 102, no. 3, p. 033 301, 2020 (cit. on p. 2 (I)).

[40] C. Huang and B. M. Rubenstein, "Machine learning diffusion monte carlo forces," *The Journal of Physical Chemistry A*, vol. 127, no. 1, pp. 339–355, 2022 (cit. on p. 2 (I)).

[41] M. Rupp, A. Tkatchenko, K.-R. Müller and O. A. Von Lilienfeld, "Fast and accurate modeling of molecular atomization energies with machine learning," *Physical review letters*, vol. 108, no. 5, p. 058 301, 2012 (cit. on p. 2 (I)).

[42] F. A. Faber et al., "Prediction errors of molecular machine learning models lower than hybrid dft error," *Journal of chemical theory and computation*, vol. 13, no. 11, pp. 5255–5264, 2017 (cit. on p. 2 (I)).

[43] B. S. Rem et al., "Identifying quantum phase transitions using artificial neural networks on experimental data," *Nature Physics*, vol. 15, no. 9, pp. 917–920, 2019 (cit. on p. 2 (I)).

[44] X.-Y. Dong, F. Pollmann, X.-F. Zhang et al., "Machine learning of quantum phase transitions," *Physical Review B*, vol. 99, no. 12, p. 121 104, 2019 (cit. on p. 2 (I)).

[45] J. Biamonte, P. Wittek, N. Pancotti, P. Rebentrost, N. Wiebe and S. Lloyd, "Quantum machine learning," *Nature*, vol. 549, no. 7671, pp. 195–202, 2017 (cit. on p. 2 (I)).

[46] L. Coopmans and M. Benedetti, "On the sample complexity of quantum boltzmann machine learning," *arXiv preprint arXiv:2306.14969*, 2023 (cit. on p. 2 (I)).

[47] J. Kempe, A. Kitaev and O. Regev, "The complexity of the local hamiltonian problem," *Siam journal on computing*, vol. 35, no. 5, pp. 1070–1097, 2006 (cit. on p. 2 (I)).

[48] E. Onorati, C. Rouzé, D. S. França and J. D. Watson, "Efficient learning of lattice quantum systems and phases of matter," *arXiv preprint arXiv:2301.12946*, 2023 (cit. on pp. 2 (I), 2 (I)).

[49] E. Onorati, C. Rouzé, D. S. França and J. D. Watson, "Provably efficient learning of phases of matter via dissipative evolutions," *arXiv preprint arXiv:2311.07506*, 2023 (cit. on pp. 2 (I), 2 (I), 17 (I)).

[50] A. Coser and D. Pérez-García, "Classification of phases for mixed states via fast dissipative evolution," *Quantum*, vol. 3, p. 174, 2019 (cit. on pp. 2 (I), 17 (I)).

[51] Y. Che, C. Gneiting and F. Nori, "Exponentially improved efficient machine learning for quantum many-body states with provable guarantees," *arXiv preprint arXiv:2304.04353*, 2023 (cit. on pp. 2 (I), 5 (I), 5 (I), 17 (I), 43 (I), 60 (I)).

[52] H.-Y. Huang, R. Kueng and J. Preskill, "Predicting many properties of a quantum system from very few measurements," *Nat. Phys.*, vol. 16, 1050—1057, 2020. [Online]. Available: `https://doi.org/10.1038/s41567-020-0932-7` (cit. on pp. 2 (I), 4 (I), 10 (I), 12 (I), 39 (I), 47 (I)).

[53] F. Santosa and W. W. Symes, "Linear inversion of band-limited reflection seismograms," *SIAM Journal on Scientific and Statistical Computing*, vol. 7, no. 4, pp. 1307–1330, 1986. DOI: `10.1137/0907087`. eprint: `https://doi.org/10.1137/0907087`. [Online]. Available: `https://doi.org/10.1137/0907087` (cit. on pp. 4 (I), 7 (I), 7 (I), 9 (I), 31 (I)).

[54] R. Tibshirani, "Regression shrinkage and selection via the lasso," *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 58, no. 1, pp. 267–288, 1996 (cit. on pp. 4 (I), 7 (I), 7 (I), 9 (I), 31 (I)).

[55] S. Shalev-Shwartz and S. Ben-David, *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014 (cit. on pp. 4 (I), 4 (I), 9 (I), 9 (I), 9 (I), 34 (I), 38 (I), 41 (I), 41 (I)).

[56] C. Saunders, A. Gammerman and V. Vovk, "Ridge regression learning algorithm in dual variables," in *Proceedings of the Fifteenth International Conference on Machine Learning*, 1998, pp. 515–521 (cit. on pp. 4 (I), 9 (I), 9 (I), 38 (I)).

[57] T. De Ryck, S. Lanthaler and S. Mishra, "On the approximation of functions by tanh neural networks," *Neural Networks*, vol. 143, 732–750, Nov. 2021, ISSN: 0893-6080. DOI: 10.1016/j.neunet.2021.08.015. [Online]. Available: http://dx.doi.org/10.1016/j.neunet.2021.08.015 (cit. on pp. 4 (I), 13 (I), 14 (I), 47 (I), 48 (I), 48 (I), 49 (I), 49 (I), 54 (I), 54 (I), 60 (I)).

[58] S. Mishra and T. K. Rusch, "Enhancing accuracy of deep learning algorithms by training with low-discrepancy sequences," *SIAM Journal on Numerical Analysis*, vol. 59, no. 3, pp. 1811–1834, 2021. DOI: 10.1137/20M1344883. eprint: https://doi.org/10.1137/20M1344883 (cit. on pp. 4 (I), 13 (I), 14 (I), 14 (I), 33 (I), 34 (I), 36 (I), 47 (I), 48 (I)).

[59] S. K. Zaremba, "The mathematical basis of monte carlo and quasi-monte carlo methods," *SIAM review*, vol. 10, no. 3, pp. 303–314, 1968 (cit. on pp. 4 (I), 12 (I), 13 (I), 33 (I), 34 (I), 34 (I)).

[60] R. E. Caflisch, "Monte carlo and quasi-monte carlo methods," *Acta numerica*, vol. 7, pp. 1–49, 1998 (cit. on pp. 4 (I), 12 (I), 13 (I), 14 (I), 33 (I), 34 (I), 34 (I), 35 (I), 35 (I), 36 (I), 36 (I), 36 (I)).

[61] H. Niederreiter, *Random number generation and quasi-Monte Carlo methods*. SIAM, 1992 (cit. on pp. 4 (I), 12 (I), 13 (I), 33 (I), 34 (I), 34 (I), 36 (I), 36 (I), 36 (I), 36 (I)).

[62] P. L'Ecuyer and C. Lemieux, "Recent advances in randomized quasi-monte carlo methods," *Modeling uncertainty: An examination of stochastic theory, methods, and applications*, pp. 419–474, 2002 (cit. on pp. 4 (I), 12 (I), 13 (I), 33 (I), 34 (I), 34 (I)).

[63] S. Bachmann, S. Michalakis, B. Nachtergaele and R. Sims, "Automorphic equivalence within gapped phases of quantum lattice systems," *Commun. Math. Phys.*, vol. 309, no. 3, pp. 835–871, 2012 (cit. on pp. 4 (I), 14 (I), 48 (I), 75 (I), 75 (I)).

[64] M. B. Hastings and X.-G. Wen, "Quasiadiabatic continuation of quantum states: The stability of topological ground-state degeneracy and emergent gauge invariance," *Phys. Rev. B*, vol. 72, no. 4, p. 045 141, 2005 (cit. on pp. 4 (I), 14 (I), 48 (I), 75 (I), 75 (I)).

[65] T. J. Osborne, "Simulating adiabatic evolution of gapped spin systems," *Phys. Rev. A*, vol. 75, no. 3, p. 032 321, 2007 (cit. on pp. 4 (I), 14 (I), 48 (I), 75 (I), 75 (I)).

[66] M. Mohri, A. Rostamizadeh and A. Talwalkar, *Foundations of machine learning*. The MIT Press, 2018 (cit. on pp. 7 (I), 7 (I), 31 (I), 33 (I), 33 (I), 41 (I), 44 (I)).

[67] H.-Y. Huang, S. Chen and J. Preskill, "Learning to predict arbitrary quantum processes," *PRX Quantum*, vol. 4, no. 4, p. 040 337, 2023 (cit. on p. 9 (I)).

[68] A. Elben et al., "Mixed-state entanglement from local randomized measurements," *Phys. Rev. Lett.*, vol. 125, p. 200 501, 20 2020. DOI: 10.1103/PhysRevLett.125.200501. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevLett.125.200501 (cit. on pp. 10 (I), 12 (I), 39 (I), 47 (I)).

[69] A. Elben et al., "The randomized measurement toolbox," *arXiv preprint arXiv:2203.11374*, 2022 (cit. on pp. 10 (I), 12 (I), 39 (I), 47 (I)).

[70] K. Wan, W. J. Huggins, J. Lee and R. Babbush, "Matchgate shadows for fermionic quantum simulation," *arXiv preprint arXiv:2207.13723*, 2022 (cit. on pp. 10 (I), 12 (I), 39 (I), 47 (I)).

[71] K. Bu, D. E. Koh, R. J. Garcia and A. Jaffe, "Classical shadows with pauli-invariant unitary ensembles," *arXiv preprint arXiv:2202.03272*, 2022 (cit. on pp. 10 (I), 12 (I), 39 (I), 47 (I)).

[72] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, JMLR Workshop and Conference Proceedings, 2010, pp. 249–256 (cit. on pp. 11 (I), 46 (I)).

[73] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014 (cit. on pp. 11 (I), 46 (I)).

[74] D. P. Kingma and J. Ba, *Adam: A method for stochastic optimization*, 2017. arXiv: 1412.6980 [cs.LG] (cit. on p. 12 (I)).

[75] S. Du, J. Lee, H. Li, L. Wang and X. Zhai, "Gradient descent finds global minima of deep neural networks," in *International conference on machine learning*, PMLR, 2019, pp. 1675–1685 (cit. on pp. 12 (I), 17 (I)).

[76] I. M. Sobol, "Uniformly distributed sequences with an additional uniform property," *USSR Computational mathematics and mathematical physics*, vol. 16, no. 5, pp. 236–242, 1976 (cit. on pp. 12 (I), 33 (I), 34 (I)).

[77] I. M. Sobol', "On the distribution of points in a cube and the approximate evaluation of integrals," *Zhurnal Vychislitel'noi Matematiki i Matematicheskoi Fiziki*, vol. 7, no. 4, pp. 784–802, 1967 (cit. on pp. 12 (I), 33 (I), 34 (I), 36 (I)).

[78] H. Niederreiter, "Point sets and sequences with small discrepancy," *Monatshefte für Mathematik*, vol. 104, pp. 273–337, 1987 (cit. on pp. 12 (I), 33 (I), 34 (I)).

[79] H. Niederreiter, "Low-discrepancy and low-dispersion sequences," *Journal of number theory*, vol. 30, no. 1, pp. 51–70, 1988 (cit. on pp. 12 (I), 33 (I), 35 (I)).

[80] J. H. Halton, "On the efficiency of certain quasi-random sequences of points in evaluating multi-dimensional integrals," *Numerische Mathematik*, vol. 2, pp. 84–90, 1960 (cit. on pp. 12 (I), 33 (I), 35 (I), 36 (I)).

[81] A. B. Owen, "Monte carlo variance of scrambled net quadrature," *SIAM Journal on Numerical Analysis*, vol. 34, no. 5, pp. 1884–1910, 1997 (cit. on pp. 12 (I), 33 (I), 35 (I), 36 (I)).

[82] C. Aistleitner and M. Hofer, *Probabilistic discrepancy bound for monte carlo point sets*, 2012. arXiv: 1211.1058 [math.NA] (cit. on pp. 14 (I), 36 (I), 61 (I)).

[83] I. Loshchilov and F. Hutter, *Decoupled weight decay regularization*, 2019. arXiv: 1711.05101 [cs.LG] (cit. on pp. 14 (I), 82 (I)).

[84] C. Cortes and V. Vapnik, "Support-vector networks," *Mach. Learn.*, vol. 20, no. 3, pp. 273–297, 1995 (cit. on p. 15 (I)).

[85] S. S. Du, X. Zhai, B. Poczos and A. Singh, *Gradient descent provably optimizes over-parameterized neural networks*, 2019. arXiv: 1810.02054 [cs.LG] (cit. on p. 17 (I)).

[86] K. O. Lye, S. Mishra and D. Ray, "Deep learning observables in computational fluid dynamics," *Journal of Computational Physics*, vol. 410, p. 109 339, 2020 (cit. on p. 33 (I)).

[87] A. B. Owen, "Multidimensional variation for quasi-monte carlo," in *Contemporary Multivariate Analysis And Design Of Experiments: In Celebration of Professor Kai-Tai Fang's 65th Birthday*, World Scientific, 2005, pp. 49–74 (cit. on p. 36 (I)).

[88] C. Aistleitner and J. Dick, *Functions of bounded variation, signed measures, and a general koksma-hlawka inequality*, 2014. arXiv: 1406.0230 [math.CA] (cit. on pp. 37 (I), 37 (I), 38 (I), 38 (I)).

[89] E. Hlawka and R. Mück, "Über eine transformation von gleichverteilten folgen ii," *Computing*, vol. 9, no. 2, pp. 127–138, 1972, ISSN: 1436-5057. DOI: 10.1007/BF02236962. [Online]. Available: https://doi.org/10.1007/BF02236962 (cit. on pp. 37 (I), 63 (I), 63 (I), 66 (I), 67 (I), 67 (I), 69 (I), 69 (I), 69 (I)).

[90] M. M. Bejani and M. Ghatee, "A systematic review on overfitting control in shallow and deep neural networks," *Artificial Intelligence Review*, pp. 1–48, 2021 (cit. on p. 60 (I)).

[91] G. Casella and R. L. Berger, "Statistical lnference," *Duxbury press*, 2002 (cit. on p. 61 (I)).

[92] M. Rosenblatt, "Remarks on a multivariate transformation," *The Annals of Mathematical Statistics*, vol. 23, no. 3, pp. 470–472, 1952, ISSN: 00034851. Accessed: 5th Mar. 2024. [Online]. Available: http://www.jstor.org/stable/2236692 (cit. on pp. 63 (I), 68 (I)).

[93] M. Zhang, A. Zhang and Y. Zhou, "Construction of uniform designs on arbitrary domains by inverse rosenblatt transformation," in May 2020, pp. 111–126, ISBN: 978-3-030-46160-7. DOI: 10.1007/978-3-030-46161-4_7 (cit. on pp. 63 (I), 68 (I)).

[94] "2. quasi-monte carlo methods for numerical integration," in *Random Number Generation and Quasi-Monte Carlo Methods*, pp. 13–22. DOI: 10.1137/1.9781611970081.ch2. eprint: https://epubs.siam.org/doi/pdf/10.1137/1.9781611970081.ch2. [Online]. Available: https://epubs.siam.org/doi/abs/10.1137/1.9781611970081.ch2 (cit. on p. 67 (I)).

[95] H. E. Haber, "Notes on the matrix exponential and logarithm," *Santa Cruz Institute for Particle Physics, University of California: Santa Cruz, CA, USA*, 2018 (cit. on p. 78 (I)).

[96] S. R. White, "Density matrix formulation for quantum renormalization groups," *Phys. Rev. Lett.*, vol. 69, pp. 2863–2866, 19 1992. DOI: 10.1103/PhysRevLett.69.2863. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevLett.69.2863 (cit. on p. 81 (I)).

[97] D. Huang, J. Niles-Weed, J. A. Tropp and R. Ward, "Matrix concentration for products," *arXiv preprint arXiv:2003.05437*, 2020 (cit. on p. 81 (I)).

Appendix

These appendices provide the technical details of the ideas discussed in the main text. In Section A, we review several important concepts for our proofs such as the algorithm and rigorous guarantee from [2] in Section A.1 and background on classical deep learning techniques in Section A.2. In Section B, we build on [2] to obtain a sample complexity upper bound for predicting ground state properties independent of system size. In Section C, we prove our guarantee for predicting ground state properties using neural networks.

A Preliminaries

A.1 Review of previous algorithm and proof

In this section, we review the previous algorithm from [2] along with intermediate results we use throughout our proofs. For full details, we refer the reader to [2]. Throughout this section, let $1/e > \epsilon_1, \epsilon_2, \epsilon_3 > 0$. One can think of ϵ_1 as the approximation error caused by the hypothesis of our ML algorithm not exactly capturing the ground state property; ϵ_2 represents the noise in the training data; ϵ_3 corresponds to the generalization error.

Recall that we consider a family of n-qubit Hamiltonians $H(x)$ smoothly parameterized by an m-dimensional vector $x \in [-1,1]^m$ that satisfies the following assumptions, which we restate from [2].

(a) *Physical system:* We consider n finite-dimensional quantum systems that are arranged at locations, or sites, in a d-dimensional space, e.g., a spin chain ($d = 1$), a square lattice ($d = 2$), or a cubic lattice ($d = 3$). Unless specified otherwise, our big-$\mathcal{O}, \Omega, \Theta$ notation is with respect to the thermodynamic limit $n \to \infty$.

(b) *Hamiltonian:* $H(x)$ decomposes into a sum of geometrically local terms $H(x) = \sum_{j=1}^{L} h_j(\vec{x}_j)$, each of which only acts on an $\mathcal{O}(1)$ number of sites in a ball of $\mathcal{O}(1)$ radius. Here, $\vec{x}_j \in \mathbb{R}^q, q = \mathcal{O}(1)$ and x is the concatenation of L vectors $\vec{x}_1, \ldots, \vec{x}_L$ with dimension $m = Lq = \mathcal{O}(n)$. Individual terms $h_j(\vec{x}_j)$ obey $\|h_j(\vec{x}_j)\|_\infty \leq 1$ and also have bounded directional derivative: $\|\partial h_j/\partial \hat{u}\|_\infty \leq 1$, where $\hat{u}$ is a unit vector in parameter space.

(c) *Ground-state subspace:* We consider the ground state $\rho(x)$ for the Hamiltonian $H(x)$ to be defined as $\rho(x) = \lim_{\beta\to\infty} e^{-\beta H(x)}/\mathrm{Tr}(e^{-\beta H(x)})$. This is equivalent to a uniform mixture over the eigenspace of $H(x)$ with the minimum eigenvalue.

(d) *Observable:* O can be written as a sum of few-body observables $O = \sum_j O_j$, where each O_j only acts on an $\mathcal{O}(1)$ number of sites. Hence, we can also write $O = \sum_{P\in S^{(\mathrm{geo})}} \alpha_P P$, where $P \in \{I, X, Y, Z\}^{\otimes n}$. We focus

on O given as a sum of geometrically local observables $\sum_j O_j$, where each O_j only acts on an $\mathcal{O}(1)$ number of sites in a ball of $\mathcal{O}(1)$ radius. Moreover, O has $\|O\|_\infty = \mathcal{O}(1)$.

We also assume that the spectral gap of $H(x)$ is bounded from below by some constant γ for all choices of parameters $x \in [-1,1]^m$.

The ML algorithm is given a training data set $\{(x_\ell, y_\ell)\}_{\ell=1}^N$, where x_ℓ is sampled from some distribution $\mathcal{D}$ over the parameter space $[-1,1]^m$ and y_ℓ approximates the ground state property: $|y_\ell - \mathrm{Tr}(O\rho(x_\ell))| \leq \epsilon_2$. The goal is to learn some function $h^*(x)$ that achieves a low average prediction error

$$\mathop{\mathbb{E}}_{x\sim\mathcal{D}} |h^*(x) - \mathrm{Tr}(O\rho(x))|^2 \leq \epsilon. \tag{A.1}$$

A.1.1 ML Algorithm

The ML algorithm proposed in [2] requires several geometric definitions. We use $S^{(\mathrm{geo})}$ to denote the set of all geometrically local Pauli observables throughout.

Let $\delta_1, \delta_2, B > 0$ be efficiently-computable hyperparameters that we define later. Then, define the set I_P of coordinates c that parameterize some local term $h_{j(c)}$ that is close to a Pauli $P \in \{I,X,Y,Z\}^{\otimes n}$. Here, the distance between two observables d_{obs} is defined as the minimum distance between the qubits that the observables act on, where the distance between qubits is given by the geometry of the system, which we assume to be known. Formally, we define the set of local coordinates as

$$I_P \triangleq \{c \in \{1,\ldots,m\} : d_{\mathrm{obs}}(h_{j(c)}, P) \leq \delta_1\}. \tag{A.2}$$

The intuition behind this set of coordinates is that it indexes the parameters x_c that influence the ground state property $\mathrm{Tr}(P\rho(x))$ corresponding to this Pauli P. Using this intuition, because these parameters x_c for $c \in I_P$ matter most for the property we are trying to learn (as [2] proves and we give the ideas for later), then we can define a new effective parameter space in which all other parameters are set to zero. Moreover, parameters x_c for $c \in I_P$ can be discretized to lie on a lattice. This gives the following set X_P

$$X_P \triangleq \left\{ x \in [-1,1]^m : \begin{array}{l} \text{if } c \notin I_P, x_c = 0 \\ \text{if } c \in I_P, x_c \in \{0, \pm\delta_2, \pm 2\delta_2, \ldots, \pm 1\} \end{array} \right\}. \tag{A.3}$$

We can also define a set $T_{x,P}$ for each vector $x \in X_P$ which is the set of parameters x' that are close to x for coordinates in I_P:

$$T_{x,P} \triangleq \left\{ x' \in [-1,1]^m : -\frac{\delta_2}{2} < x_c - x'_c \leq \frac{\delta_2}{2},\ \forall c \in I_P \right\}. \tag{A.4}$$

With these definitions in place, we set the hyperparameters as follows. Define δ_1 as

$$\delta_1 \triangleq \max\left(C_{\max} \log^2(2C/\epsilon_1), C_4, C_5, \frac{\max(5900, \alpha, 7(d+11), \theta)}{b} \right), \tag{A.5}$$

where $b, C_{\max}, C_4, C_5, \alpha, \theta, C$ are all constants. We refer to the supplementary information of [2] for a full description of these constants. Moreover, δ_2 is given by

$$\delta_2 \triangleq \frac{1}{\left\lceil \frac{2\sqrt{C'|I_P|}}{\epsilon_1} \right\rceil}, \tag{A.6}$$

where C' is a constant. Finally, we define an additional hyperparameter $B > 0$ as

$$B \triangleq 2^{\mathcal{O}(\text{polylog}(1/\epsilon_1))}. \tag{A.7}$$

The ML algorithm from [2] utilizes these objects to encode the geometric locality of the system. The algorithm consists of two steps. First, it maps the parameter space $[-1, 1]^m$ to a high dimensional space $\mathbb{R}^{m_\phi}$ for

$$m_\phi \triangleq \sum_{P \in S^{(\text{geo})}} |X_P| = \mathcal{O}(n) 2^{\mathcal{O}(\text{polylog}(1/\epsilon_1))} \tag{A.8}$$

via a nonlinear feature map ϕ. Second, it runs ℓ_1-regularized linear regression (LASSO) over the feature space.

This first step encodes the geometry of the problem. In particular, the feature map is defined as follows, where each coordinate of $\phi(x)$ is indexed by $x' \in X_P$ and $P \in S^{(\text{geo})}$

$$\phi(x)_{x',P} \triangleq \mathbb{1}[x \in T_{x',P}]. \tag{A.9}$$

In this way, the feature map $\phi(x)$ identifies the nearest lattice point to x. The idea is that one can approximate the ground state property well by only approximating it at these representative points and summing up. We make this intuition rigorous in the following section.

Following the feature mapping, our ML algorithm uses LASSO [53], [54], [66] to learn functions of the form $\{h(x) = \mathbf{w} \cdot \phi(x) : \|w\|_1 \leq B\}$. In particular, for a chosen hyperparameter $B > 0$, LASSO finds a coefficient vector $\mathbf{w}^*$ that solves the following optimization problem minimizing the training error subject to the constraint that $\|w\|_1 \leq B$

$$\min_{\substack{\mathbf{w} \in \mathbb{R}^{m_\phi} \\ \|w\|_1 \leq B}} \frac{1}{N} \sum_{\ell=1}^{N} |\mathbf{w} \cdot \phi(x_\ell) - y_\ell|^2. \tag{A.10}$$

We denote the learned function by $h^*(x) = \mathbf{w}^* \cdot \phi(x)$. Note that the learned function does not need to achieve the minimum training error, but can be some amount $\epsilon_3/2$ above it. For our purposes, we set $B = 2^{\mathcal{O}(\text{polylog}(1/\epsilon_1))}$.

This algorithm obtains the following rigorous guarantee.

Theorem A.1 (Theorem 5 in [2]). *Let $1/e > \epsilon_1, \epsilon_2, \epsilon_3 > 0$ and $\delta > 0$. Given training data $\{(x_\ell, y_\ell)\}_{\ell=1}^N$ of size*

$$N = \log(n/\delta) 2^{\mathcal{O}(\log(1/\epsilon_3) + \text{polylog}(1/\epsilon_1))}, \tag{A.11}$$

where x_ℓ is sampled from $\mathcal{D}$ and y_ℓ is an estimator of $\mathrm{Tr}(O\rho(x_\ell))$ such that $|y_\ell - \mathrm{Tr}(O\rho(x_\ell))| \leq \epsilon_2$, the ML algorithm can produce $h^*(x)$ that achieves prediction error

$$\mathop{\mathbb{E}}_{x\sim\mathcal{D}} |h^*(x) - \mathrm{Tr}(O\rho(x))|^2 \leq (\epsilon_1 + \epsilon_2)^2 + \epsilon_3 \tag{A.12}$$

with probability at least $1-\delta$. The training time for constructing the hypothesis function h and the prediction time for computing $h^*(x)$ are upper bounded by $\mathcal{O}(nN) = n\log(n/\delta)2^{\mathcal{O}(\log(1/\epsilon_3)+\mathrm{polylog}(1/\epsilon_1))}$.

A.1.2 Proof Ideas

This rigorous guarantee is proven by first showing that the training error

$$\hat{R}(h) = \min_{\mathbf{w}} \frac{1}{N}\sum_{\ell=1}^{N} |h(x_\ell) - y_\ell|^2 \tag{A.13}$$

is small.

Lemma A.2 (Lemma 15 in [2])**.** *The function*

$$g(x) = \sum_{P\in S^{(\mathrm{geo})}} \sum_{x'\in X_P} f_P(x')\mathbb{1}[x \in T_{x',P}] = \mathbf{w}'\cdot\phi(x), \tag{A.14}$$

achieves training error

$$\hat{R}(g) \leq (\epsilon_1 + \epsilon_2)^2. \tag{A.15}$$

The proof of this consists of three different steps. First, one can show that $\mathrm{Tr}(O\rho(x))$ can be approximated by a sum of smooth local functions, denoted as $f(x) = \sum_{P\in S^{(\mathrm{geo})}} f_P(x)$, where $f_P(x) = \alpha_P \mathrm{Tr}(P\rho(\chi_P(x)))$ for $O = \sum_{P\in\{I,X,Y,Z\}^{\otimes n}} \alpha_P P$ and

$$\chi_P(x)_c = \begin{cases} x_c, & c \in I_P \\ 0 & c \notin I_P \end{cases} \tag{A.16}$$

for all $c \in \{1,\ldots,m\}$. In other words, parameters that parameterize local terms h_j far away a Pauli P (x_c for $c \notin I_P$) do not contribute much to the ground state property, and thus we can simply set them to zero. Formally, this approximation is given in the following lemma.

Lemma A.3 (Corollary 2 in [2])**.** *Consider a class of local Hamiltonians $\{H(x) : x \in [-1,1]^m\}$ and an observable $O = \sum_{P\in\{I,X,Y,Z\}^{\otimes n}} \alpha_P P$ satisfying assumptions (a)-(d). There exists a constant $C > 0$ such that for any $1/e > \epsilon_1 > 0$,*

$$|\mathrm{Tr}(O\rho(x)) - f(x)| \leq C\epsilon_1 \left(\sum_{P\in S^{(\mathrm{geo})}} |\alpha_P|\right), \tag{A.17}$$

where $f(x) = \sum_{P\in S^{(\mathrm{geo})}} f_P(x)$.

Second, one can also show that this sum of local functions $f(x) = \sum_{P \in S^{(\mathrm{geo})}} f_P(x)$ can in turn be approximated by a linear function over the feature space $g(x) = \mathbf{w}' \cdot \phi(x)$, where $\mathbf{w}'$ is a vector with entries indexed by $P \in S^{(\mathrm{geo})}$ and $x' \in X_P$ given by $\mathbf{w}'_{x',P} = f_P(x')$.

Lemma A.4 (Corollary 3 in [2]). *For $g(x) = \mathbf{w}' \cdot \phi(x)$ and $f(x) = \sum_{P \in S^{(\mathrm{geo})}} f_P(x)$, then writing an observable $O = \sum_{P \in \{I,X,Y,Z\}^{\otimes n}} \alpha_P P$, we have*

$$|g(x) - f(x)| < \epsilon_1 \left(\sum_{P \in S^{(\mathrm{geo})}} |\alpha_P| \right) \tag{A.18}$$

for any x.

This tells us that the hypothesis functions of the ML algorithm indeed approximate the ground state properties well. The final piece needed is a norm inequality bounding the ℓ_1-norm of the Pauli coefficients. This allows us to bound the terms involving $|\alpha_P|$ in Lemma A.3 and Lemma A.4. In particular, we have the following bound.

Theorem A.5 (Corollary 4 in [2]). *Let $O = \sum_{P \in \{I,X,Y,Z\}^{\otimes n}} \alpha_P P$ be an observable that can be written as a sum of geometrically local observables. Then,*

$$\sum_P |\alpha_P| = \mathcal{O}(1). \tag{A.19}$$

Given these results, Lemma A.2 follows directly by triangle inequality and rescaling ϵ_1 when using Lemma A.3 and Lemma A.4. Finally, to prove Theorem A.1, it remains to bound the generalization error by ϵ_3. This follows directly from known sample complexity guarantees for the LASSO algorithm [66], which learns ℓ_1-regularized linear functions. In order to apply this known result, one needs to provide a regularization parameter, i.e., some $B > 0$ such that the ML algorithm learns functions of the form $h(x) = \mathbf{w} \cdot \phi(x)$ for $\|\mathbf{w}\|_1 \leq B$. To choose such a B, Lewis et al. bound the ℓ_1-norm of $\mathbf{w}'$, where recall $\mathbf{w}'_{x',P} = f_P(x')$.

Lemma A.6 (Lemma 14 in [2]). *Let $\mathbf{w}'$ be the vector of coefficients defined by $\mathbf{w}'_{x',P} = f_P(x')$. Then,*

$$\|\mathbf{w}'\|_1 = \sum_{P \in S^{(\mathrm{geo})}} \sum_{x' \in X_P} |f_P(x')| = 2^{\mathcal{O}(\mathrm{polylog}(1/\epsilon_1))}. \tag{A.20}$$

Using $B = 2^{\mathcal{O}(\mathrm{polylog}(1/\epsilon_1))}$ in the known guarantees for LASSO [66] gives Theorem A.1.

A.2 Deep learning with low-discrepancy sequences

In this section, we review results in classical deep learning theory for obtaining rigorous guarantees when learning from data sampled according to a low-discrepancy sequence (LDS) [59], [60], [61], [62], [76], [77], [78], [79], [80], [81]. For this discussion, we follow [58], [86].

We consider a *neural network* or *multi-layer perceptron model* as a composition of several layers of affine transformations and nonlinear activation functions. Namely, let $\sigma : \mathbb{R} \to \mathbb{R}$ be a nonlinear activation function. Then, a neural network with L layers is defined as follows. Let $d_0, \dots, d_L \in \mathbb{N}$ be the dimension (number of neurons or *width*) of each layer $k \in \{0, \dots, L\}$. Here, the zeroth layer is the *input layer* and the Lth layer is the *output layer*. At each layer $k \in \{0, \dots, L-1\}$ except for the output layer, we define an affine transformation $W_k : \mathbb{R}^{d_k} \to \mathbb{R}^{d_{k+1}}$ by $W_k(x) = A_k x + b_k$ for a matrix of *weights* $A_k \in \mathbb{R}^{d_{k+1} \times d_k}$ and a vector of *biases* $b_k \in \mathbb{R}^{d_{k+1}}$. Then, a neural network is defined as

$$f_\theta(x) = (W_{L-1} \circ \sigma \circ \cdots \circ \sigma \circ W_0)(x), \tag{A.21}$$

where σ is applied element-wise and $\theta = ((A_0, b_0), \dots, (A_{L-1}, b_{L-1}))$. The *hidden layers* are the first $L-1$ layers. Here, θ are the trainable parameters of the neural network, which can be iteratively updated through training on data. A *deep neural network* is a neural network with at least three layers: $L \geq 3$. In this work, we consider the activation function

$$\sigma(x) = \tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}. \tag{A.22}$$

We refer to such neural networks with this activation function as *tanh neural networks*.

Suppose a neural network f_θ aims to approximate some target function f, given training data $\{(x_\ell, f(x_\ell)\}_{\ell=1}^N$. Then, the training error is defined as

$$\hat{R}(\theta) = \frac{1}{N} \sum_{\ell=1}^{N} |f(x_\ell) - f_\theta(x_\ell)|^2. \tag{A.23}$$

The prediction error is then defined over the whole domain, including unseen data, as

$$R(\theta) = \mathop{\mathbb{E}}_{x \sim \mathcal{D}} |f(x) - f_\theta(x)|^2, \tag{A.24}$$

where the training data is sampled from some distribution $\mathcal{D}$. A canonical result in deep learning theory [55] is that the generalization error can be bounded by roughly

$$R(\theta) \lesssim \hat{R}(\theta) + \mathcal{O}\left(\frac{1}{\sqrt{N}}\right), \tag{A.25}$$

where $\lesssim$ indicates that we only state this result schematically. Importantly, this means that in order for the neural network f_θ to approximate f with high accuracy, many training data points N are needed, which is undesirable. In order to fix this issue, [58] combines ideas from deep learning with tools from quasi-Monte Carlo methods [59], [60], [61], [62] to achieve a generalization error bound of

$$R(\theta) \lesssim \hat{R}(\theta) + \tilde{\mathcal{O}}\left(\frac{1}{N}\right), \tag{A.26}$$

where $\tilde{\mathcal{O}}$ indicates that we are suppressing polylogarithmic factors. The key tool used here is low-discrepancy sequences [59], [60], [61], [62], [76], [77], [78],

[79], [80], [81]. Intuitively, this is a collection of points that covers domain of the function f in such a way that there are no large gaps, or discrepancies. By filling these gaps, one can ensure that the training data accurately represents the target function, more so than even uniformly random data. We leverage these ideas to obtain our rigorous guarantee on the sample complexity of a deep learning algorithm for predicting ground state properties. In the following, we formally define low-discrepancy sequences and a key inequality in quasi-Monte Carlo theory for obtaining our generalization bound.

First, we define the discrepancy of a sequence, which is a measure of uniformity.

Definition A.7 (Discrepancy [60])**.** *Let λ be the Lebesgue measure, $N \in \mathbb{N}$. Let $x = \{x_\ell\}_{\ell=1}^N$ be a sequence of points with $x_\ell \in [0,1]^d$ for all ℓ. The* discrepancy *of the sequence x is defined as*

$$D_N(d) = \sup_{J \in E} |R_N(J)|, \tag{A.27}$$

where

$$R_N(J) = \frac{1}{N} \sum_{\ell=1}^{N} \mathbb{1}\{x_\ell \in J\} - \lambda(J) \tag{A.28}$$

for a Lebesgue-measurable set $J \subseteq [0,1]^d$. Also, E is the set of all rectangular subsets of $[0,1]^d$, i.e.,

$$E = \left\{ \prod_{i=1}^{d} [a_i, b_i) : 0 \leq a_i < b_i \leq 1 \right\}. \tag{A.29}$$

Intuitively, one can consider the discrepancy as a measure of how well the sequence fills rectangular subsets of $[0,1]^d$. If the discrepancy is small, this means that the sequence fills these subsets well. We can similarly define the star-discrepancy, where the supremum is instead taken over rectangular subsets of $[0,1]^d$ such that one endpoint is 0.

Definition A.8 (Star-discrepancy [60])**.** *Let λ be the Lebesgue measure, $N \in \mathbb{N}$. Let $x = \{x_\ell\}_{\ell=1}^N$ be a sequence of points with $x_\ell \in [0,1]^d$ for all ℓ. The* star-discrepancy *of the sequence x is defined as*

$$D_N^*(d) = \sup_{J \in E^*} |R_N(J)|, \tag{A.30}$$

where R_N is defined in Equation (A.28) *for a Lebesgue-measurable set $J \subseteq [0,1]^d$. Also, E^* is the set of all rectangular subsets of $[0,1]^d$, i.e.,*

$$E^* = \left\{ \prod_{i=1}^{d} [0, b_i) : 0 < b_i \leq 1 \right\}. \tag{A.31}$$

With these definitions, we can define low-discrepancy sequences.

Definition A.9 (Low-discrepancy sequence [60])**.** *A sequence of points* $x = \{x_\ell\}_{\ell=1}^N$ *with* $x_\ell \in [0,1]^d$ *for all* ℓ *is a* low-discrepancy sequence *if*

$$D_N^*(d) \leq C\frac{(\log N)^d}{N}, \tag{A.32}$$

where C *is a constant that possibly depends on* d *but is independent of* N*.*

The value of the constant C in this definition depends on the construction of the low-discrepancy sequence. Several constructions of low-discrepancy sequences are known [61], [77], [80], [81]. In this work, we consider Sobol sequences in base 2 [61]. For these sequences, we have the following guarantee

Theorem A.10 (Theorem 4.17 in [61])**.** *Let* $N \in \mathbb{N}$*. If* $x = \{x_\ell\}_{\ell=1}^N$ *is a Sobol sequence in base* 2 *with* $x_\ell \in [0,1]^d$ *for all* ℓ*, then the star-discrepancy satisfies*

$$D_N^*(d) \leq C(d)\frac{(\log N)^d}{N}, \tag{A.33}$$

where $C(d)$ *is a constant satisfying*

$$C(d) < \frac{1}{d!}\left(\frac{d}{\log(2d)}\right). \tag{A.34}$$

We state this result without proof and refer to [61] for details on this construction. Another important known discrepancy bound that we will use in Section C.3 is the following bound on the star-discrepancy of uniformly random points.

Lemma A.11 (Corollary 1 in [82])**.** *For any* $d \geq 1, N \geq 1$ *and* $\delta \in (0,1)$ *a (uniformly) randomly generated* d*-dimensional point set* $(x_1, \ldots, x_N)$ *satisfies*

$$D_N^*(d) \leq 5.7\sqrt{4.9 + \log\left(\frac{1}{\delta}\right)}\frac{\sqrt{d}}{\sqrt{N}} \tag{A.35}$$

with probability at least $1 - \delta$*.*

As discussed earlier, low-discrepancy sequences allow us to obtain better sample complexity guarantees for neural networks. The key result in quasi-Monte Carlo theory that enables this is the Koksma-Hlawka inequality [60]. In order to properly state it, we first need to define the Hardy-Krause variation. A full technical definition can be found in, e.g., [58], but for our purposes, it suffices to consider the following upper bound [87]. Let f be a "sufficiently smooth" function. Then, its Hardy-Krause variation can be upper bounded by

$$V_{HK}(f) \leq \hat{V}_{HK} = \int_{[0,1]^d}\left|\frac{\partial^d f(y)}{\partial y_i \cdots \partial y_d}\right| dy + \sum_{i=1}^d \hat{V}_{HK}(f_1^{(i)}), \tag{A.36}$$

where $f_1^{(i)}$ is the restriction of the function f to the boundary $y_i = 1$. If all of the mixed partial derivatives are continuous, then this inequality is actually an equality [60]. Now, we can state the Koksma-Hlawka inequality.

Theorem A.12 (Koksma-Hlawka inequality)**.** *Let $f : [0,1]^d \to \mathbb{R}$ be a function whose mixed derivatives are absolutely integrable over its domain with bounded Hardy-Krause variation $V_{HK}(f) < \infty$. Let $x = \{x_\ell\}_{\ell=1}^N$ be a sequence of N d-dimensional points in $[0,1]^d$ with star-discrepancy $D_N^*(d)$. Then*

$$\left| \int_{[0,1]^d} f(x)\, dx - \frac{1}{N} \sum_{\ell=1}^{N} f(x_\ell) \right| \leq V_{HK}(f) D_N^*(d). \tag{A.37}$$

This theorem is used in quasi-Monte Carlo methods to estimate the error of approximating an integral of a function f by the empirical average of f evaluated on a sequence of points. Notice that if the sequence x is a low-discrepancy sequence, then by definition, we can upper bound the star-discrepancy D_N^*. Moreover, recalling the definitions of prediction error and training error (Equations (A.23) and (A.24)), one can see how this relates to our task of bounding the prediction error.

To generalize our results to a wider class of distributions, we need to extend these tools for arbitrary measures, rather than just the Lebesgue measure. First, we restate the definition of discrepancy and star-discrepancy [88].

Definition A.13 (General Discrepancy [89])**.** *Let μ be a normalized Borel measure on $[0,1]^d$. Let $x = \{x_\ell\}_{\ell=1}^N$ be a sequence of points with $x_\ell \in [0,1]^d$ for all ℓ. The* discrepancy with respect to *μ of the sequence x is defined as*

$$D_N(d;\mu) = \sup_{J \in E} |R_N(J;\mu)|, \tag{A.38}$$

where

$$R_N(J;\mu) = \frac{1}{N} \sum_{\ell=1}^{N} \mathbb{1}\{x_\ell \in J\} - \mu(J) \tag{A.39}$$

for a Borel-measurable set $J \subseteq [0,1]^d$. Also, E is the set of all rectangular subsets of $[0,1]^d$, i.e.,

$$E = \left\{ \prod_{i=1}^{d} [a_i, b_i) : 0 \leq a_i < b_i \leq 1 \right\}. \tag{A.40}$$

Definition A.14 (General Star-Discrepancy [88])**.** *Let μ be a normalized Borel measure on $[0,1]^d$, and let $N \in \mathbb{N}$. Let $x = \{x_\ell\}_{\ell=1}^N$ be a sequence of points with $x_\ell \in [0,1]^d$ for all ℓ. The* star-discrepancy with respect to *μ of the sequence x is defined as*

$$D_N^*(d;\mu) = \sup_{J \in E^*} |R_N(J;\mu)|, \tag{A.41}$$

where R_N is defined in Equation (A.39) *for a Borel-measurable set $J \subseteq [0,1]^d$. Also, E^* is the set of all rectangular subsets of $[0,1]^d$, i.e.,*

$$E^* = \left\{ \prod_{i=1}^{d} [0, b_i) : 0 < b_i \leq 1 \right\}. \tag{A.42}$$

These definitions coincide with Definition A.7 and Definition A.8 when μ is the Lebesgue measure λ. Moreover, we can define general low-discrepancy sequences similarly to Definition A.9 with respect this general star-discrepancy. There is also a generalized Koksma-Hlawka inequality [88], which we state below.

Theorem A.15 (Generalized Koksma-Hlawka inequality; Theorem 1 in [88])**.** *Let $f : [0,1]^d \to \mathbb{R}$ be a measurable function whose mixed derivatives are absolutely integrable over its domain with bounded Hardy-Krause variation $V_{HK}(f) < \infty$. Let μ be a normalized Borel measure on $[0,1]^d$, and let $x = \{x_\ell\}_{\ell=1}^N$ be a sequence of N d-dimensional points in $[0,1]^d$ with general star-discrepancy $D_N^*(d;\mu)$. Then,*

$$\left| \frac{1}{N} \sum_{\ell=1}^{N} f(x_\ell) - \int_{[0,1]^d} f(x)\, d\mu(x) \right| \leq V_{HK}(f) D_N^*(d;\mu). \tag{A.43}$$

B Constant sample complexity

In this section, we show that with a simple modification of the algorithm from [2], we can reduce the sample complexity to $\mathcal{O}(1)$ for a constant prediction error. We consider all of the same definitions/notation as in Section A.1. This section is similar to Section IV in the Supplementary Information of [2]. As in [2], our algorithm first maps the parameter space $[-1,1]^m$ into a high-dimensional feature space $\mathbb{R}^{m_\phi}$ for m_ϕ given in Equation (A.8) via a feature map ϕ. Our simple modification is to use the feature map defined by

$$\tilde{\phi}(x)_{x',P} \triangleq \operatorname{sign}(\alpha_P)\sqrt{|\alpha_P|}\mathbb{1}\{x \in T_{x',P}\}, \tag{B.1}$$

where each coordinate of $\phi(x)$ is indexed by $P \in S^{(\mathrm{geo})}, x' \in X_P$. Note that defining the feature map in this way requires knowledge of the observable $O = \sum_P \alpha_P P$ corresponding to the ground state property to be predicted. However, in practice, this is a natural assumption. The hypothesis class for our proposed ML algorithm consists of linear functions in this feature space, i.e., functions of the form $h(x) = \mathbf{w} \cdot \phi(x)$. Then, our algorithm learns these functions via ridge regression [55], [56]. For a chosen hyperparameter $\Lambda > 0$, ridge regression finds a vector $\mathbf{w}^*$ that solves the following optimization problem minimizing the training error subject to the constraint that $\|\mathbf{w}\|_2 \leq \Lambda$

$$\min_{\substack{\mathbf{w} \in \mathbb{R}^{m_\phi} \\ \|\mathbf{w}\|_2 \leq \Lambda}} \frac{1}{N} \sum_{\ell=1}^{N} |\mathbf{w} \cdot \tilde{\phi}(x_\ell) - y_\ell|^2, \tag{B.2}$$

where y_ℓ approximates $\mathrm{Tr}(O\rho(x_\ell))$. We denote the learned function by $h^*(x) = \mathbf{w}^* \cdot \tilde{\phi}(x)$. Note that the learned function does not need to achieve the minimum training error, but can be some amount say $\epsilon_3/2$ above it. For our purposes, we choose the hyperparameter to be $\Lambda = 2^{\mathcal{O}(\mathrm{polylog}(1/\epsilon_1))}$, which we justify in the next section.

Note that there are two main differences from the algorithm in [2]. First, recall from Section A.1 that the feature map was previously defined as $\phi(x)_{x',P} = \mathbb{1}\{x \in T_{x',P}\}$ for $P \in S^{(\mathrm{geo})}, x' \in X_P$. Second, instead of using LASSO (ℓ_1-regularized regression), our proposed algorithm uses ridge regression.

With this algorithm, we obtain the following guarantee.

Theorem B.1 (Constant sample complexity; Detailed restatement of Theorem 3.1)**.** *Let $1/e > \epsilon_1, \epsilon_2, \epsilon_3 > 0$ and $\delta > 0$. Given training data $\{(x_\ell, y_\ell)\}_{\ell=1}^N$ of size*

$$N = \log(1/\delta) 2^{\mathcal{O}(\log(1/\epsilon_3) + \mathrm{polylog}(1/\epsilon_1))}, \tag{B.3}$$

where x_ℓ is sampled from $\mathcal{D}$ and y_ℓ is an estimator of $\mathrm{Tr}(O\rho(x_\ell))$ such that $|y_\ell - \mathrm{Tr}(O\rho(x_\ell))| \leq \epsilon_2$, the ML algorithm can produce $h^*(x)$ that achieves prediction error

$$\mathop{\mathbb{E}}_{x\sim\mathcal{D}} |h^*(x) - \mathrm{Tr}(O\rho(x))|^2 \leq (\epsilon_1 + \epsilon_2)^2 + \epsilon_3 \tag{B.4}$$

with probability at least $1 - \delta$. The training time for constructing the hypothesis function h^* and the prediction time for computing $h^*(x)$ are upper bounded by

$$\mathcal{O}(n)\mathrm{polylog}(1/\delta) 2^{\mathcal{O}(\log(1/\epsilon_3) + \mathrm{polylog}(1/\epsilon_1))}. \tag{B.5}$$

Comparing to Theorem A.1, notice that our sample complexity guarantee is completely independent of system size n.

The theorem in the main text corresponds to $\epsilon_1 = 0.2\epsilon, \epsilon_2 = \epsilon$, and $\epsilon_3 = 0.4\epsilon$. In this way, $(\epsilon_1 + \epsilon_2)^2 \leq 1.44\epsilon^2 \leq 0.53\epsilon$ and $(\epsilon_1 + \epsilon_2)^2 + \epsilon_3 \leq \epsilon$.

So far, we have only considered the setting in which we learn a specific ground state property $\mathrm{Tr}(O\rho(x))$ for a fixed observable O. Because our training data is given in the form $\{(x_\ell, y_\ell)\}_{\ell=1}^N$, where y_ℓ approximates $\mathrm{Tr}(O\rho(x))$ for this fixed observable O, if we want to predict a new property for the same ground state $\rho(x)$, we would need to generate new training data. Thus, it may be more useful to learn a ground state representation, from which we could predict $\mathrm{Tr}(O\rho(x))$ for many different choices of observables O without requiring new training data. In this case, suppose we are instead given training data $\{x_\ell, \sigma_T(\rho(x_\ell))\}_{\ell=1}^N$, where $\sigma_T(\rho(x_\ell))$ is a classical shadow representation [52], [68], [69], [70], [71] of the ground state $\rho(x_\ell)$. An immediate corollary of Theorem B.1 is that we can predict ground state representations with the same sample complexity. This follows from the same proof as Corollary 5 in [2].

Corollary B.2 (Learning representations of ground states; detailed restatement of Corollary 3.2)**.** *Let $1/e > \epsilon_1, \epsilon_2, \epsilon_3 > 0$ and $\delta > 0$. Given training data $\{(x_\ell, \sigma_T(\rho(x_\ell))\}_{\ell=1}^N$ of size*

$$N = \log(1/\delta) 2^{\mathcal{O}(\log(1/\epsilon_3) + \mathrm{polylog}(1/\epsilon_1))}, \tag{B.6}$$

where x_ℓ is sampled from $\mathcal{D}$ and $\sigma_T(\rho(x_\ell)$ is the classical shadow representation of the ground state $\rho(x_\ell)$ using T randomized Pauli measurements.

For $T = \mathcal{O}(\log(nN/\delta)/\epsilon_2^2) = \tilde{\mathcal{O}}(\log(n/\delta)/\epsilon_2^2)$, the ML algorithm can produce a ground state representation $\hat{\rho}_{N,T}(x)$ that achieves

$$\mathop{\mathbb{E}}_{x\sim\mathcal{D}} |\mathrm{Tr}(O\hat{\rho}_{N,T}(x)) - \mathrm{Tr}(O\rho(x))|^2 \leq (\epsilon_1 + \epsilon_2)^2 + \epsilon_3 \tag{B.7}$$

with probability at least $1-\delta$, for any observable with eigenvalues between -1 and 1 that can be written as a sum of geometrically local observables.

We note that the number of measurements T needed to generate the training data scales as $\log(n)$, but the amount of training data still remains constant with respect to system size. We do not consider the number of measurements as contributing to the sample complexity because in our setting, the ML algorithm is given this training data as input and does not generate it itself.

B.1 Training error bound

To prove Theorem B.1, we first derive a bound on the training error. Recall that the training error is defined as

$$\hat{R}(h) = \min_{\mathbf{w}} \frac{1}{N} \sum_{\ell=1}^{N} |h(x_\ell) - y_\ell|^2. \tag{B.8}$$

Define the vector $\tilde{\mathbf{w}}$ with entries indexed by $P \in S^{(\mathrm{geo})}, x' \in X_P$ by

$$\tilde{\mathbf{w}}_{x',P} \triangleq \sqrt{|\alpha_P|}\mathrm{Tr}(P\rho(\chi_P(x))), \tag{B.9}$$

where $\chi_P(x)$ is defined in Equation (A.16). Then, notice that

$$\begin{aligned}
\tilde{g}(x) &\triangleq \tilde{\mathbf{w}} \cdot \tilde{\phi}(x) && \text{(B.10)}\\
&= \sum_{P\in S^{(\mathrm{geo})}} \sum_{x'\in X_P} \mathrm{sign}(\alpha_P)|\alpha_P|\mathrm{Tr}(P\rho(\chi_P(x)))\mathbb{1}\{x \in T_{x',P}\} && \text{(B.11)}\\
&= \sum_{P\in S^{(\mathrm{geo})}} \sum_{x'\in X_P} \alpha_P\mathrm{Tr}(P\rho(\chi_P(x)))\mathbb{1}\{x \in T_{x',P}\} && \text{(B.12)}\\
&= \mathbf{w}' \cdot \phi(x) && \text{(B.13)}\\
&= g(x), && \text{(B.14)}
\end{aligned}$$

where $g(x) = \mathbf{w}' \cdot \phi(x)$ with $\mathbf{w}'_{x',P} = \alpha_P\mathrm{Tr}(P\rho(\chi_P(x)))$ and $\phi(x)_{x',P} = \mathbb{1}\{x \in T_{x',P}\}$. By Lemma A.2, we know that $g(x)$ approximates the ground state property with low training error, and thus, in turn, $\tilde{g}(x)$ also approximates the ground state property well. The existence of $\tilde{\mathbf{w}}$ such that $\tilde{g}(x) = \tilde{\mathbf{w}} \cdot \tilde{\phi}(x)$ guarantees that the function $h^*(x) = \mathbf{w}^* \cdot \tilde{\phi}(x)$ found by performing via ridge regression will also yield a small training error. More formally, we have the following guarantee

Lemma B.3 (Training error)**.** *The function*

$$\tilde{g}(x) = \tilde{\mathbf{w}} \cdot \tilde{\phi}(x) = \sum_{P\in S^{(\mathrm{geo})}} \sum_{x'\in X_P} \alpha_P\mathrm{Tr}(P\rho(\chi_P(x)))\mathbb{1}\{x \in T_{x',P}\} \tag{B.15}$$

achieves training error

$$\hat{R}(\tilde{g}) \leq (\epsilon_1 + \epsilon_2)^2. \tag{B.16}$$

Since $\tilde{g}(x) = g(x)$, this follows directly from Lemma A.2. Moreover, we can obtain an ℓ_2-norm bound on $\tilde{\mathbf{w}}$. We can utilize this upper bound to choose the hyperparameter $\Lambda > 0$ such that $\|\mathbf{w}\|_2 \leq \Lambda$. Thus, we have the following lemma,

Lemma B.4 (ℓ_2-Norm bound)**.** *Let $\tilde{\mathbf{w}}$ be the vector of coefficients defined in Equation* (B.9)*. Then, we have*

$$\|\tilde{\mathbf{w}}\|_2^2 = \sum_{P \in S^{(\text{geo})}} \sum_{x' \in X_P} |\alpha_P| |\text{Tr}(P\rho(\chi_P(x)))|^2 = 2^{\mathcal{O}(\text{polylog}(1/\epsilon_1))}. \tag{B.17}$$

Proof. This is a simple consequence of Lemma A.6. Explicitly, we have

$$\begin{aligned} \|\tilde{\mathbf{w}}\|_2 &= \sum_{P \in S^{(\text{geo})}} \sum_{x' \in X_P} |\alpha_P| |\text{Tr}(P\rho(\chi_P(x)))|^2 && \text{(B.18)} \\ &\leq \sum_{P \in S^{(\text{geo})}} \sum_{x' \in X_P} |\alpha_P| |\text{Tr}(P\rho(\chi_P(x)))| && \text{(B.19)} \\ &= 2^{\mathcal{O}(\text{polylog}(1/\epsilon_1))}, && \text{(B.20)} \end{aligned}$$

where the second line follows because $\text{Tr}(P\rho(\chi_P(x))) \leq 1$ and the last line follows by Lemma A.6. □

This justifies our choice of $\Lambda = 2^{\mathcal{O}(\text{polylog}(1/\epsilon_1))}$. Now consider the learned function $h^*(x) = \mathbf{w}^* \cdot \tilde{\phi}(x)$, where $\mathbf{w}^*$ is found by minimizing the training error subject to the constraint that $\|\mathbf{w}\|_2 \leq \Lambda$. We do not require the learned function to achieve the minimum training error, but it can be some amount $\epsilon_3/2$ above it, i.e.,

$$\hat{R}(h^*) \leq \frac{\epsilon_3}{2} + \min_{\substack{\mathbf{w} \\ \|\mathbf{w}\|_2 \leq \Lambda}} \frac{1}{N} \sum_{\ell=1}^{N} |\mathbf{w} \cdot \tilde{\phi}(x_\ell) - y_\ell|^2. \tag{B.21}$$

Since we chose $\Lambda = 2^{\mathcal{O}(\text{polylog}(1/\epsilon_1))}$ and we showed in Lemma B.4 that $\|\tilde{\mathbf{w}}\|_2 \leq \Lambda$, then the minimum training error is at most $\hat{R}(\tilde{g})$. We also know that this is bounded by $(\epsilon_1 + \epsilon_2)^2$ by Lemma B.3. This then implies

$$\hat{R}(h^*) \leq \frac{\epsilon_3}{2} + \hat{R}(g) \leq (\epsilon_1 + \epsilon_2)^2 + \frac{\epsilon_3}{2}. \tag{B.22}$$

B.2 Prediction error bound

To prove Theorem B.1, it remains to bound the prediction error. We can use a standard result from machine learning theory on the prediction error of ridge regression algorithms [55], [66].

Theorem B.5 (Theorem 26.12 in [55])**.** *Suppose that $\mathcal{D}$ is a distribution over $\mathcal{X} \times \mathcal{Y}$ such that with probability 1 we have that $\|\boldsymbol{x}\|_2 \leq R$.*
Let $\mathcal{H} = \{\boldsymbol{x} \mapsto \boldsymbol{w} \cdot \boldsymbol{x} : \|\boldsymbol{w}\|_2 \leq \Lambda\}$ and let $l : \mathcal{H} \times Z \to \mathbb{R}$ be a loss function of the form $\ell(\boldsymbol{w}, (\boldsymbol{x}, y)) = \phi(\boldsymbol{w} \cdot \boldsymbol{x}, y)$ such that for all $y \in \mathcal{Y}, a \mapsto \phi(a, y)$ is a ρ-Lipschitz function and such that $\max_{a \in [-\Lambda R, \Lambda R]} |\phi(a, y)| \leq c$. Then, for any

$\delta \in (0,1)$, *with probability of at least* $1-\delta$ *over the choice of an i.i.d. sample of size* N,

$$\forall h \in \mathcal{H}, \qquad R(h) \leq \hat{R}_S(h) + \frac{2\rho\Lambda R}{\sqrt{N}} + c\sqrt{\frac{2\log(2/\delta)}{N}}. \tag{B.23}$$

Here, $R(h)$ denotes the prediction error for the hypothesis h. With this, we can complete the proof of Theorem B.1.

Proof of Theorem B.1. First, let us reframe the theorem in our setting. Consider the input space $\mathcal{X}$ to be the parameter space $[-1,1]^m$ and our input variable is $\boldsymbol{x} = \tilde{\phi}(x)$. Since the observables we consider have spectral norm at most 1, the output space fulfils $\mathcal{Y} \subseteq [-1,1]$. The hypothesis set is $\mathcal{H} = \{x \mapsto \mathbf{w} \cdot \tilde{\phi}(x) : \|\mathbf{w}\|_2 \leq \Lambda\}$, where in the previous section, we set $\Lambda = 2^{\mathcal{O}(\mathrm{polylog}(1/\epsilon_1))}$.

It remains to check the conditions of the theorem. We begin by showing that $\|\boldsymbol{x}\|_2 \leq R$ for some $R > 0$. We have the following computation:

$$\begin{aligned}
\|\boldsymbol{x}\|_2 = \tilde{\phi}(x) \cdot \tilde{\phi}(x) &= \left\|\tilde{\phi}(x)\right\|_2^2 && \text{(B.24)}\\
&= \sum_{P \in S^{(\mathrm{geo})}} \sum_{x' \in X_P} |\mathrm{sign}(\alpha_P)\sqrt{|\alpha_P|}\mathbb{1}\{x \in T_{x',P}\}|^2 && \text{(B.25)}\\
&= \sum_{P \in S^{(\mathrm{geo})}} \sum_{x' \in X_P} |\alpha_P|\mathbb{1}\{x \in T_{x',P}\} && \text{(B.26)}\\
&= \sum_{P \in S^{(\mathrm{geo})}} |\alpha_P| && \text{(B.27)}\\
&= \mathcal{O}(1), && \text{(B.28)}
\end{aligned}$$

where the second to last line follows because for a given P, $x \in T_{x',P}$ for exactly one $x' \in X_P$. This is shown in Corollary 3 of [2]. Also, the last line follows by Theorem A.5. Thus, we can take $R = \mathcal{O}(1)$.

Finally, note that $\ell(\boldsymbol{w}, (\boldsymbol{x}, y)) = |\boldsymbol{w} \cdot \boldsymbol{x} - y| = \phi(\boldsymbol{w} \cdot \boldsymbol{x}, y)$. Therefore, $\phi(a, y)$ is a 1-Lipschitz function and fulfils

$$\max_{a \in [-\Lambda R, \Lambda R]} |\phi(a,y)| = \max_{a \in [-\Lambda R, \Lambda R]} |a - y| \leq \Lambda R + 1. \tag{B.29}$$

Thus, we can consider $\rho = 1$ and $c = \mathcal{O}(1) \cdot 2^{\mathcal{O}(\mathrm{polylog}(1/\epsilon_1))} + 1$.

By Equation (B.22), we know that the learned model $h^*(x) = \mathbf{w}^* \cdot \tilde{\phi}(x)$ achieves

$$\hat{R}(h^*) \leq (\epsilon_1 + \epsilon_2)^2 + \frac{\epsilon_3}{2}. \tag{B.30}$$

Plugging in $R = \mathcal{O}(1)$, $\rho = 1$, $\Lambda = 2^{\mathcal{O}(\mathrm{polylog}(1/\epsilon_1))}$ and $c = \mathcal{O}(1) \cdot 2^{\mathcal{O}(\mathrm{polylog}(1/\epsilon_1))} + 1$ into Theorem B.5, we have

$$\begin{aligned}
&R(h^*) && \text{(B.31)}\\
\leq&(\epsilon_1 + \epsilon_2)^2 + \frac{\epsilon_3}{2} && \text{(B.32)}
\end{aligned}$$

$$+\frac{1}{\sqrt{N}}\left(2\mathcal{O}(1)\cdot 2^{\mathcal{O}(\mathrm{polylog}(1/\epsilon_1))}+\left(\mathcal{O}(1)\cdot 2^{\mathcal{O}(\mathrm{polylog}(1/\epsilon_1))}+1\right)\sqrt{2\log(2/\delta)}\right) \tag{B.33}$$

with probability at least $1-\delta$. In order to bound the prediction error by $(\epsilon_1+\epsilon_2)^2+\epsilon_3$, we need N to be large enough such that

$$\frac{1}{\sqrt{N}}\left(2\mathcal{O}(1)\cdot 2^{\mathcal{O}(\mathrm{polylog}(1/\epsilon_1))}+\left(\mathcal{O}(1)\cdot 2^{\mathcal{O}(\mathrm{polylog}(1/\epsilon_1))}+1\right)\sqrt{2\log(2/\delta)}\right) \tag{B.34}$$

$$\leq\frac{\epsilon_3}{2}. \tag{B.35}$$

Solving for N in this inequality and simplifying we have

$$N=\frac{4}{\epsilon_3^2}2^{\mathcal{O}(\mathrm{polylog}(1/\epsilon_1))}(1+\sqrt{\log(1/\delta)})^2=2^{\mathcal{O}(\log(1/\epsilon_3)+\mathrm{polylog}(1/\epsilon_1))}\log(1/\delta). \tag{B.36}$$

Thus, for this N, we can guarantee that $R(h^*)\leq(\epsilon_1+\epsilon_2)^2+\epsilon_3$, as claimed. □

On another note, when considering a scenario with a fixed number of parameters $m=\mathcal{O}(1)$, much like the setting in [51], the expression derived from the result in Lemma C.10 exhibits polynomial dependence on ϵ. One can incorporate the constant number of parameters by setting $\tilde{m}=m$. Thus, we recover the exact ground state properties $\mathrm{Tr}(P\rho(x))$ in f_P and the approximation error resulting from applying Lemma A.3 vanishes completely. Furthermore, we can slightly adapt the proof of Lemma B.4 and obtain

$$\|\tilde{\mathbf{w}}\|_2^2=\sum_{P\in S^{(\mathrm{geo})}}\sum_{x'\in X_P}|\alpha_P||\mathrm{Tr}(P\rho(\chi_P(x)))|=\max_{P\in S^{(\mathrm{geo})}}|X_P|\sum_{Q\in S^{(\mathrm{geo})}}|\alpha_P| \tag{B.37}$$

$$=\mathcal{O}(\epsilon^{-m}), \tag{B.38}$$

where the last step is performed similarly as in the proof of Lemma A.6.

B.3 Computational time for training and prediction

It remains to analyze the computational time for the ML algorithm's training and prediction.

Proof of computational time in Theorem B.1. The training time is dominated by the time required for ridge regression over the feature space defined by the feature map ϕ. Recall that the optimization problem under considerations is

$$\min_{\substack{\mathbf{w}\in\mathbb{R}^{m_\phi}\\ \|\mathbf{w}\|_2\leq\Lambda}}\frac{1}{N}\sum_{\ell=1}^{N}|\mathbf{w}\cdot\tilde{\phi}(x_\ell)-y_\ell|^2. \tag{B.39}$$

One can show that this is a convex optimization problem so that we can solve its equivalent dual problem instead. This dual optimization problem is given by

$$\max_{\alpha\in\mathbb{R}^N}-\alpha^\intercal(K+\lambda I)\alpha+2\alpha\cdot Y, \tag{B.40}$$

where the kernel matrix is $K = X^\intercal X$, for the feature matrix $X \in \mathbb{R}^{m_\phi \times N}$ defined by $X = (\tilde{\phi}(x_1) \cdots \tilde{\phi}(x_N))$ and the response vector $Y = (y_1, \ldots, y_N)^\intercal$. If κ is the maximum time it takes to compute a kernel entry $K(x, x') = \tilde{\phi}(x) \cdot \tilde{\phi}(x')$, then one can show that the time to solve this dual problem is $\mathcal{O}(\kappa N^2 + N^3)$. Moreover, prediction can be executed in $\mathcal{O}(\kappa N)$. For more details in this analysis, we refer the reader to, e.g., Section 11.3.2 of [66].

In our case, $\kappa = \mathcal{O}(m_\phi)$ since $\tilde{\phi}(x) \in \mathbb{R}^{m_\phi}$ and the kernel is simply the dot product of two of these vectors. By Equation (A.8), we know that

$$m_\phi = \mathcal{O}(n) 2^{\mathcal{O}(\mathrm{polylog}(1/\epsilon_1))} \tag{B.41}$$

so that $\kappa = \mathcal{O}(n) 2^{\mathcal{O}(\mathrm{polylog}(1/\epsilon_1))}$. Moreover, by Theorem B.1,

$$N = \log(1/\delta) 2^{\mathcal{O}(\log(1/\epsilon_3) + \mathrm{polylog}(1/\epsilon_1))}. \tag{B.42}$$

Plugging this into the time required to solve the dual problem for kernel ridge regression, we have

$$\mathcal{O}(\kappa N^2 + N^3) = \mathcal{O}(n)\mathrm{polylog}(1/\delta) 2^{\mathcal{O}(\log(1/\epsilon_3) + \mathrm{polylog}(1/\epsilon_1))}. \tag{B.43}$$

Moreover, the prediction time is given by

$$\mathcal{O}(\kappa N) = \mathcal{O}(n)\mathrm{polylog}(1/\delta) 2^{\mathcal{O}(\log(1/\epsilon_3) + \mathrm{polylog}(1/\epsilon_1))}. \tag{B.44}$$

□

C Rigorous guarantees for neural networks

In this section, we derive a rigorous guarantee on the sample complexity of a deep-learning based model for predicting ground state properties. Similarly to the previous sections, let $1/e > \epsilon_1, \epsilon_2, \epsilon_3 > 0$ throughout. One can think of ϵ_1 as the approximation error caused by our neural network model not exactly capturing the ground state property; ϵ_2 represents the noise in the training data; ϵ_3 corresponds to the generalization error.

Recall again the setup, where we consider a family of n-qubit Hamiltonians $H(x) = \sum_{j=1}^{L} h_j(\vec{x}_j)$ parameterized by an m-dimensional vector $x \in [-1, 1]^m$, which satisfies the assumptions (a)-(c) in Section A.1. Let $\rho(x)$ denote the ground state of $H(x)$. We consider the task of predicting ground state properties $\mathrm{Tr}(O\rho(x))$ for some observable O that satisfies assumption (d) in Section A.1, where we are given training data $\{(x_\ell, y_\ell)\}_{\ell=1}^N$ with $y_\ell \approx \mathrm{Tr}(O\rho(x_\ell))$. In particular, suppose $|y_\ell - \mathrm{Tr}(O\rho(x_\ell))| \leq \epsilon_2$. Furthermore, we also assume that all mixed partial derivatives of order $\tilde{m} \triangleq |I_P|$ of h_j are bounded as

$$\left\| \frac{\partial^{\tilde{m}}}{\partial x_1 \partial x_2 \ldots \partial x_{\tilde{m}}} h_j(x) \right\|_\infty \leq 1, \tag{C.1}$$

where I_P is the set of local coordinates defined in Equation (A.2). Here, we denote the number of local coordinates by $\tilde{m} = |I_P|$ for ease of notation. This is similar in spirit to assumption (b), in which we assume that the local terms

have bounded directional derivatives: $\|\partial h_j/\partial \hat{u}\|_\infty \leq 1$, where $\hat{u}$ is a unit vector in parameter space.

Let $S^{(\mathrm{geo})}$ denote the set of geometrically local Pauli observables. Our deep neural network model consists of $|S^{(\mathrm{geo})}| = \mathcal{O}(n)$ "local" multi-layer perceptron models (defined in generally in Section A.2) with two hidden layers and tanh activation functions. Their outputs are combined through a linear layer without activation function. Formally, our model is defined as follows.

Definition C.1 (Deep neural network model)**.** *The neural network model is given by a function $f^{\Theta,w} : [-1,1]^m \to \mathbb{R}$ defined by*

$$f^{\Theta,w}(x) = \sum_{P \in S^{(\mathrm{geo})}} w_P f_P^{\theta_P}(x), \tag{C.2}$$

where the "local models" $f_P^{\theta_P} : [-1,1]^{\tilde{m}} \to \mathbb{R}$ are given by

$$f_P^{\theta_P}(x) = (W_{\mathrm{out}} \circ \tanh \circ W_{\mathrm{hidden}} \circ \tanh \circ W_{\mathrm{in}} \circ \tau^{-1})(x), \tag{C.3}$$

with $\tau^{-1}(x) = (x+1)/2$ and $\theta_P = [(W_{\mathrm{in}}, b_{\mathrm{in}}), (W_{\mathrm{hidden}}, b_{\mathrm{hidden}}), (W_{\mathrm{out}}, b_{\mathrm{out}})]$. Here, $W_{\mathrm{in}} \in \mathbb{R}^{\tilde{m} \times W}$, $b_{\mathrm{in}} \in \mathbb{R}^W$, $W_{\mathrm{hidden}} \in \mathbb{R}^{W \times W}$, $b_{\mathrm{hidden}} \in \mathbb{R}^W$, $W_{\mathrm{out}} \in \mathbb{R}^{W \times 1}$ and $b_{\mathrm{out}} \in \mathbb{R}$, where W denotes the width of the hidden layers. The weights are given by $\Theta = \{\theta_P : P \in S^{(\mathrm{geo})}\}$ in the local models and $w \in \mathbb{R}$ in the last layer. Furthermore, we denote the individual parameters by $\Theta_i \in \mathbb{R}$.

Using this model, we can establish an objective function that we aim to minimize during the training process. Specifically, this objective function comprises the mean square error along with a lasso penalty applied to the weights w in the final layer.

Definition C.2 (Training objective)**.** *Let $f^{\Theta,w}$ be a neural network model as in Definition C.1. Let $\{(x_\ell, y_\ell)\}_{\ell=1}^N$ be the training data set and $\lambda > 0$ be some regularization parameter that may depend on $\epsilon_1, \epsilon_2 > 0$. The training objective is given by*

$$\frac{1}{N} \sum_{\ell=1}^{N} |f^{\Theta,w}(x_\ell) - y_\ell|^2 + \lambda \|w\|_1. \tag{C.4}$$

Our proposed ML algorithm then operates as in Algorithm 1.

After training our model using Algorithm 1, we obtain the following rigorous guarantee.

Theorem C.3 (Neural network sample complexity guarantee)**.** *Let $1/e > \epsilon_1, \epsilon_2, \epsilon_3 > 0$. Let $f^{\Theta^*,w^*} : [-1,1]^m \to \mathbb{R}$ be a neural network model produced from Algorithm 1 trained on data $\{(x_\ell, y_\ell)\}_{\ell=1}^N$ of size*

$$N = 2^{\mathcal{O}(\mathrm{polylog}(1/\epsilon_1) + \mathrm{polylog}(1/\epsilon_3))}, \tag{C.5}$$

where the x_ℓ's form a low-discrepancy Sobol sequence and $|y_\ell - \mathrm{Tr}(O\rho(x_\ell))| \leq \epsilon_2$. Suppose that f^{Θ^*,w^*} achieves a training error of at most $((\epsilon_1+\epsilon_2)^2 + \epsilon_3)/2$. Additionally, suppose that all parameters Θ_i^* of f^{Θ^*,w^*} satisfy $|\Theta_i^*| \leq W_{\max}$,

Algorithm 1: Deep learning-based prediction of ground state properties

Sample N low-discrepancy points $\{x_\ell\}_{\ell=1}^N$;
Collect training labels $\{y_\ell\}_{\ell=1}^N$, where $y_\ell \approx \mathrm{Tr}(O\rho(x_\ell))$;
Data: $\{(x_\ell, y_\ell)\}_{\ell=1}^N$;
Fix $|I_P|$;
Initialize model architecture according to Definition C.1 with appropriate hyperparameter δ_1, width W as in Theorem C.8 and weights Θ, w using an appropriate initialization method (e.g., Xavier initialization [72]);
Train with respect to the objective in Definition C.2 with appropriate hyperparameter $\lambda > 0$ using a quasi-Monte Carlo training algorithm, e.g., Adam [73] until convergence;
Obtain locally optimal parameters Θ^*, w^*;
Result: Classical representation f^{Θ^*, w^*};

for some $W_{\max} > 0$ that is independent of the system size n. Then the neural network f^{Θ^*, w^*} achieves prediction error

$$\mathop{\mathbb{E}}_{x \sim U[-1,1]^m} |f^{\Theta^*, w^*}(x) - \mathrm{Tr}(O\rho(x))|^2 \leq 2(\epsilon_1 + \epsilon_2)^2 + \epsilon_3, \tag{C.6}$$

where $x \sim U[-1,1]^m$ denotes sampling x from a uniform distribution over $[-1,1]^m$.

We prove this theorem in the next two sections (Sections C.1 and C.2). As a corollary of this, we obtain the theorem stated in the main text. We discuss the assumptions that the distribution $\mathcal{D}$ must satisfy in depth and prove the corollary in Section C.3. The theorem in the main text (Theorem 3.3) corresponds to $\epsilon_1 = \epsilon_3 = 0.1\epsilon$ and $\epsilon_2 = \epsilon$. Hence, $2(\epsilon_1 + \epsilon_2)^2 \leq 2.44\epsilon^2 \leq 0.9\epsilon$ for $1/e > \epsilon > 0$ and so $2(\epsilon_1 + \epsilon_2)^2 + \epsilon_3 \leq \epsilon$.

Corollary C.4 (Neural network sample complexity guarantee; detailed restatement of Theorem 3.3)**.** *Let $1/e > \epsilon_1, \epsilon_2, \epsilon_3 > 0$, $\mathcal{D}$ a distribution with PDF g satisfying the following properties: g has full support and is continuously differentiable on $[-1,1]^m$. Moreover, g is of the form*

$$g(x) = \prod_{j=1}^{L} g_j(\vec{x}_j). \tag{C.7}$$

Let $f^{\Theta^*, w^*} : [-1,1]^m \to \mathbb{R}$ be a neural network model produced from Algorithm 1 trained on data $\{(x_\ell, y_\ell)\}_{\ell=1}^N$ of size

$$N = \sqrt{\log(1/\delta)} 2^{\mathcal{O}(\mathrm{polylog}(1/\epsilon_1) + \mathrm{polylog}(1/\epsilon_3))}, \tag{C.8}$$

where the $x_\ell \sim \mathcal{D}$ and $|y_\ell - \mathrm{Tr}(O\rho(x_\ell))| \leq \epsilon_2$. Suppose that f^{Θ^*, w^*} achieves a training error of at most $((\epsilon_1 + \epsilon_2)^2 + \epsilon_3)/2$. Additionally, suppose that

all parameters Θ_i^* *of* f^{Θ^*,w^*} *satisfy* $|\Theta_i^*| \leq W_{\max}$*, for some* $W_{\max} > 0$ *that is independent of the system size* n*. Then the neural network* f^{Θ^*,w^*} *achieves prediction error*

$$\mathop{\mathbb{E}}_{x\sim\mathcal{D}} |f^{\Theta^*,w^*}(x) - \mathrm{Tr}(O\rho(x))|^2 \leq 2(\epsilon_1+\epsilon_2)^2 + \epsilon_3, \tag{C.9}$$

with probability at least $1-\delta$*.*

Moreover, while we can show the existence of suitable parameters that achieve a low training error, quantified by our training objective in Definition C.2, in general, we cannot prove that Algorithm 1 converges to parameters close to this optimum because our training objective is not convex. Thus, to obtain the guarantee in Theorem C.3, we need to assume that a low training error is indeed achieved by Algorithm 1. This is commonly satisfied in practice.

Similar to Corollary B.2, if we are instead given training data $\{x_\ell, \sigma_T(\rho(x_\ell))\}_{\ell=1}^N$, where $\sigma_T(\rho(x_\ell))$ is a classical shadow representation [52], [68], [69], [70], [71] of the ground state $\rho(x_\ell)$, then an immediate corollary of Theorem C.3 is that we can predict ground state representations with the same sample complexity. This follows from the same proof as Corollary 5 in [2].

Corollary C.5 (Learning representations of ground states with neural networks; detailed restatement of Corollary 3.4)**.** *Let* $1/e > \epsilon_1, \epsilon_2, \epsilon_3 > 0$ *and* $\delta > 0$*. Given training data* $\{(x_\ell, \sigma_T(\rho(x_\ell))\}_{\ell=1}^N$ *of size*

$$N = \sqrt{\log(1/\delta)}\, 2^{\mathcal{O}(\mathrm{polylog}(1/\epsilon_3)+\mathrm{polylog}(1/\epsilon_1))}, \tag{C.10}$$

where x_ℓ *is sampled from a distribution* $\mathcal{D}$ *satisfying the same assumptions as Corollary C.4 and* $\sigma_T(\rho(x_\ell)$ *is the classical shadow representation of the ground state* $\rho(x_\ell)$ *using* T *randomized Pauli measurements.*
For $T = \mathcal{O}(\log(nN/\delta)/\epsilon_2^2) = \tilde{\mathcal{O}}(\log(n/\delta)/\epsilon_2^2)$*, the ML algorithm can produce a ground state representation* $\hat{\rho}_{N,T}(x)$ *that achieves*

$$\mathop{\mathbb{E}}_{x\sim\mathcal{D}} |\mathrm{Tr}(O\hat{\rho}_{N,T}(x)) - \mathrm{Tr}(O\rho(x))|^2 \leq 2(\epsilon_1+\epsilon_2)^2 + \epsilon_3 \tag{C.11}$$

with probability at least $1-\delta$*, for any observable with eigenvalues between* -1 *and* 1 *that can be written as a sum of geometrically local observables.*

We review low-discrepancy sequences and techniques in quasi-Monte Carlo theory in Section A.2, which we use in our proof. To prove Theorem C.3, we first show that there exists weights Θ', w' such that our proposed neural network $f^{\Theta',w'}$ achieves a low training error, i.e., it approximates the ground state properties $\mathrm{Tr}(O\rho(x))$ well. We show this using results in classical deep learning theory about approximating arbitrary functions with neural networks [57]. Then, we use the Koksma-Hlawka inequality (Theorem A.12) to bound the prediction error of our model, similarly to [58]. As we want to derive a bound which is independent of the size of the physical system, our approach requires some additional steps. Since the dimension of the input domain of our model depends on the size of the physical system, we cannot treat it as constant as in

[58]. Therefore, we bound the prediction error with respect to local functions, whose domain size is independent of the system size.

Moreover, recall that the Koksma-Hlawka inequality produces a bound in terms of the star-discrepancy (Definition A.8) and the Hardy-Krause variation. The star-discrepancy can be bounded by considering low-discrepancy sequences (Definition A.9), and the Hardy-Krause variation can be bounded by Equation (A.36). We derive explicit bounds for the Hardy-Krause variation of the ground state properties $\mathrm{Tr}(O\rho(x))$, using tools from the spectral flow formalism [63], [64], [65]. To obtain Corollary C.4, we follow a similar proof but use results relating the discrepancy with respect to the Lebesgue measure to the discrepancy with respect to arbitrary measures and bounds on the discrepancy of uniformly random points (Section A.2).

In Section C.1, we prove that our model approximates the ground state properties well. Then, in Section C.2, we use the Koksma-Hlawka inequality to bound the prediction error of our model. Technical results explicitly bounding the mixed partial derivatives of the ground state properties are found in Section C.4. We use these in Section C.2 to bound the Hardy-Krause variation. We then generalize this to data sampled from different distributions, as in Corollary C.4, in Section C.3.

C.1 Approximation of ground state properties by neural networks

In this section, we prove that when choosing the number of parameters and width of the model appropriately, there exists a parameter set for which the deep neural network model approximates the ground state properties well. This shows the existence of a neural network with low training error. The proof is a direct application of the main result from [57], which proves that tanh neural networks can approximate sufficiently smooth functions, in combination with the bounds on the mixed derivative of $\mathrm{Tr}(P\rho(x))$ we derived in Section C.4.

We consider the local functions defined as in Section A.1.2. Namely, define $f(x) = \sum_{P\in S^{(\mathrm{geo})}} \alpha_P f_P(x)$, where $f_P(x) = \mathrm{Tr}(P\rho(\chi_P(x)))$ for $O = \sum_{P\in\{I,X,Y,Z\}^{\otimes n}} \alpha_P P$ and

$$\chi_P(x)_c = \begin{cases} x_c, & c \in I_P \\ 0 & c \notin I_P \end{cases} \tag{C.12}$$

for all $c \in \{1,\ldots,m\}$, for I_P defined in Equation (A.2). Note that here, we slightly alter the definition from Section A.1.2, where we do not include the coefficient α_P in the definition of $f_P(x)$. Because all parameters with coordinates not in I_P are set to 0, we can view f_P as a function taking inputs in $[-1,1]^{\tilde{m}}$, where recall that we use $\tilde{m} = |I_P|$ to denote the number of local coordinates.

We show that there exists a neural network that can approximate these local functions f_P well. In order to apply the result from [57] to approximate $\mathrm{Tr}(P\rho(\chi_P(x)))$, we need to transform its inputs, such that it becomes a map $[0,1]^{\tilde{m}} \to \mathbb{R}$. Therefore, we introduce an appropriate function $\tau(x) = 2x - 1$.

To avoid confusion when considering the different domains $[-1,1]^m$ versus $[0,1]^m$, if an input $x \in [-1,1]^m$, we simply denote it by x. If $x \in [0,1]^m$, we denote it by $\bar{x}$. We similarly use this notation for other domain dimensions.

In the following lemma, we use $W^{k,\infty}(\Omega)$ for $\Omega \subseteq \mathbb{R}^m$ to denote the Sobolev space

$$W^{k,\infty}(\Omega) \triangleq \left\{ f \in L^\infty(\Omega) : \frac{\partial^{|\alpha|} f}{\partial x_1^{\alpha_1} \cdots \partial x_m^{\alpha_m}} \in L^\infty(\Omega) \quad \forall \alpha \in \mathbb{N}_0^m \text{ with } |\alpha| \leq k \right\} \tag{C.13}$$

so that the Sobolev norm is defined as

$$\|f\|_{W^{k,\infty}(\Omega)} \triangleq \max_{|\alpha|=s} \left\| \frac{\partial^{|\alpha|} f}{\partial x_1^{\alpha_1} \cdots \partial x_m^{\alpha_m}} \right\|_{L^\infty(\Omega)}. \tag{C.14}$$

for $\alpha \in \mathbb{N}_0^m$ and the L^∞-norm is defined by

$$\|f\|_{L^\infty(\Omega)} = \sup_{x \in \Omega} \|f\|. \tag{C.15}$$

Lemma C.6 (Existence of approximating neural network)**.** *Let $\epsilon_1 > 0$, $s, M \in \mathbb{N}$. Let $\|H(x)\|_{W^{s,\infty}([-1,1]^m)} \leq 1$. Define functions $f_P : [0,1]^{\tilde{m}} \to \mathbb{R}$ as $f_P(\tau(\bar{x})) = \mathrm{Tr}(P\rho(\chi_P(\tau(\bar{x}))))$, where $\tau(\bar{x}) = 2\bar{x} - 1$. Then, there exist neural networks $\hat{f}_P^M$, such that*

$$\left\| f_P \circ \tau - \hat{f}_P^M \right\|_{L^\infty([0,1]^{\tilde{m}})} \leq \epsilon_1 \tag{C.16}$$

with at most $\epsilon_1^{-\frac{\tilde{m}+1}{s}} 2^{\mathcal{O}(\tilde{m}\log(\tilde{m}))}$ parameters. Furthermore, the weights scale as $2^{\mathrm{poly}(\log(1/\epsilon_1),\tilde{m},s)}$.

To prove this, we utilize the main result from [57], which states that a neural network with tanh activation functions can approximate effectively any function.

Theorem C.7 (Theorem 5.1 in [57])**.** *Let $d, s \in \mathbb{N}$, $R > 0$, $d > 0$ and $f \in W^{s,\infty}([0,1]^d)$. There exist constants $\mathcal{C}(d,k,s,f)$, $N_0(d) > 0$, such that for every $N \in \mathbb{N}$ with $N > N_0(d)$ there exists a* tanh *neural network $\hat{f}^N$ with two hidden layers, one of width at most $3\lceil \frac{s}{2} \rceil |P_{s-1,d+1}| + d(N-1)$ (where $|P_{n,d}| = \binom{n+d-1}{n}$) and another of width at most $3\lceil \frac{d+2}{2} \rceil |P_{d+1,d+1}| N^d$ (or $3\lceil \frac{s}{2} \rceil + N - 1$ and $6N$ for $d = 1$), such that,*

$$\|f - \hat{f}^N\|_{L^\infty([0,1]^d)} \leq (1+\delta)\frac{\mathcal{C}(d,0,s,f)}{N^s}, \tag{C.17}$$

and for $k = 1, \ldots, s-1$,

$$\|f - \hat{f}^N\|_{W^{k,\infty}([0,1]^d)} \tag{C.18}$$

$$\leq 3^d(1+\delta)(2(k+1))^{3k} \max\left\{ R^k, \ln^k\left(\beta N^{s+d+2}\right) \right\} \frac{\mathcal{C}(d,k,s,f)}{N^{s-k}}, \tag{C.19}$$

where we define

$$\beta = \frac{k^3 2^d \sqrt{d} \max\{1, \|f\|^{1/2}_{W^{k,\infty}([0,1]^d)}\}}{\delta \min\{1, \sqrt{\mathcal{C}(d,k,s,f)}\}}. \tag{C.20}$$

If $f \in C^s([0,1]^d)$, then it holds that

$$\mathcal{C}(d,k,s,f) = \max_{0\le l\le k} \frac{1}{(s-l)!}\left(\frac{3d}{2}\right)^{s-l} |f|_{W^{s,\infty}([0,1]^d)}, \qquad N_0(d) = \frac{3d}{2}, \tag{C.21}$$

and else it holds that

$$\mathcal{C}(d,k,s,f) = \max_{0\le l\le k} \frac{\pi^{1/4}\sqrt{s}}{(s-l-1)!}\left(5d^2\right)^{s-l} |f|_{W^{s,\infty}([0,1]^d)}, \qquad N_0(d) = 5d^2. \tag{C.22}$$

Moreover, the weights of $\hat{f}^N$ scale as $\mathcal{O}\left(\mathcal{C}^{-s/2} N^{d(d+s^2+k^2)/2} (s(s+2))^{3s(s+2)}\right)$.

The proof of Lemma C.6 follows by an application of Theorem C.7.

Proof of Lemma C.6. We directly apply Theorem C.7, with the input space dimension $\tilde{m}$, where $\tilde{m}$ is the number of local parameters $\tilde{m} = |I_P|$. By Corollary C.28, then we have

$$\left\| f_P \circ \tau - \hat{f}_P^M \right\|_{L^\infty([0,1]^{\tilde{m}})} \le \frac{(1+\delta)}{s!}\left(\frac{3\tilde{m}Cs^2}{2M}\right)^s. \tag{C.23}$$

We want to show that this is bounded above by ϵ_1. By rearranging, we find that this holds when

$$\epsilon_1^{-\frac{1}{s}} \left(\frac{(1+\delta)}{s!}\right)^{\frac{1}{s}} \frac{3}{2}\tilde{m}Cs^2 \le M. \tag{C.24}$$

Note that by composing with f_P with τ, we acquire an extra factor of 2^s, which can be considered a component of C. Using $M = \mathcal{O}(\epsilon_1^{-\frac{1}{s}}\tilde{m}s^2)$, this results in the widths of the two layers being

$$3\left\lceil \frac{s}{2} \right\rceil \binom{s+\tilde{m}}{\tilde{m}+1} + \tilde{m}(M-1) \text{ and } \left\lceil \frac{\tilde{m}+2}{2} \right\rceil \binom{2\tilde{m}+2}{d+1} M^{\tilde{m}}, \tag{C.25}$$

and therefore at most

$$(c_1\tilde{m})^{c_2\tilde{m}} \epsilon_1^{-\frac{\tilde{m}+1}{s}} = 2^{\mathcal{O}(\tilde{m}(\log(\tilde{m})+\log(1/\epsilon_1)/s))} \tag{C.26}$$

trainable weights in the neural network. The constants c_1 and c_2 are independent of $\tilde{m}$, but may depend on s. By Theorem C.7, the weights scale as

$$\mathcal{O}\left(\mathcal{C}^{-s/2}\left(\epsilon_1^{-\frac{1}{s}}\tilde{m}s^2\right)^{\tilde{m}(\tilde{m}+s^2+k^2)/2} (s(s+2))^{3s(s+2)}\right) = \epsilon_1^{-\frac{\tilde{m}+1}{s}} 2^{\mathcal{O}(\tilde{m}\log(\tilde{m}))}. \tag{C.27}$$

□

Although the dependence on s is not relevant for our final statement, it is important to comment on the effect of the smoothness of $H(x)$. The result states that the dependence of the required parameters with respect to $1/\epsilon_1$ improves with the highest degree for which all mixed derivatives of $H(x)$ are bounded. When $H(x)$ is analytic, s can be chosen to be very large so that the number of parameters in the neural network almost scales as $\mathcal{O}(\tilde{m}s^{\log(\tilde{m})})$. The constant scales rather poorly with s; however, this effect is only be visible for very small ϵ_1.

Using Lemma C.6, we can show that there exist parameters such that the complete model approximates $\mathrm{Tr}(O\rho(x))$ and obtains a small training objective (defined in Definition C.2). The theorem (Theorem 3.5) in the main text corresponds to $\epsilon_1 = 0.2\epsilon, \epsilon_2 = \epsilon$ so that $(\epsilon_1 + \epsilon_2)^2 \leq 1.44\epsilon^2 \leq 0.53\epsilon \leq \epsilon$.

Theorem C.8 (Detailed restatement of Theorem 3.5)**.** *For any $1/e > \epsilon_1, \epsilon_2 > 0$ and appropriate width W, there exist weights Θ', w' such that the neural network model $f^{\Theta',w'}$ satisfies*

$$|f^{\Theta',w'}(x) - \mathrm{Tr}(O\rho(x))| \leq \epsilon_1 \tag{C.28}$$

for any $x \in [-1,1]^m$. In particular, for any collection of N training data points $\{(x_\ell, y_\ell)\}_{\ell=1}^N$ with $|y_\ell - \mathrm{Tr}(O\rho(x_\ell))| \leq \epsilon_2$, we have

$$\frac{1}{N}\sum_{\ell=1}^{N} |f^{\Theta',w'}(x_\ell) - y_\ell|^2 + \lambda\|w'\|_1 \leq (\epsilon_1 + \epsilon_2)^2 \tag{C.29}$$

for a suitable choice of regularization parameter $\lambda = \mathcal{O}(\epsilon_1^2)$. Moreover, each parameter Θ_i of the network has a magnitude of $|\Theta_i| = 2^{\mathcal{O}(\mathrm{polylog}(1/\epsilon_1))}$.

Proof. Write $O = \sum_P \alpha_P \mathrm{Tr}(P\rho(x))$. By Theorem A.5, let $D = \mathcal{O}(1)$ be a constant such that

$$\sum_P |\alpha_P| \leq D. \tag{C.30}$$

For every Pauli P, then by Lemma C.6, there exist weights θ'_P such that a neural network $\bar{f}_P^{\theta'} : [0,1]^{\tilde{m}} \to \mathbb{R}$ with two hidden layers as in Definition C.1 approximates the local functions $f_P(\tau(\bar{x})) = \mathrm{Tr}(P\rho(\chi_P(\tau(\bar{x}))))$, where $\tau(\bar{x}) = 2\bar{x} - 1$, up to an error $\epsilon_1/(4D)$, when the width of their hidden layers is chosen as $W = \epsilon_1^{-\frac{\tilde{m}+1}{s}} 2^{\mathcal{O}(\tilde{m}\log(\tilde{m}))}$, where the number of local coordinates is given by $\tilde{m} = |I_P|$ and by the smoothness assumption Item (d), $s \geq 1$. In other words, we have

$$\left|\bar{f}_P^{\theta'_P}(\bar{x}) - \mathrm{Tr}(P\rho(\chi_P(\tau(\bar{x}))))\right| \leq \frac{\epsilon_1}{4D}, \tag{C.31}$$

where $\bar{x} \in [0,1]^{\tilde{m}}$. Because τ is simply a coordinate transformation, then we also obtain

$$\left|f_P^{\theta'_P}(x) - \mathrm{Tr}(P\rho(\chi_P(x)))\right| \leq \frac{\epsilon_1}{4D}, \tag{C.32}$$

for $x \in [-1,1]^{\tilde{m}}$.

Furthermore, by Lemma A.3 in Section A.1.2, the sum of the local functions $f(x) = \sum_P \alpha_P f_P(x)$ approximates the ground state property $\mathrm{Tr}(O\rho(x)) =$

$\sum_P \alpha_P \mathrm{Tr}(P\rho(x))$ well. In particular, combining Lemma A.3 with Theorem A.5, we have

$$\left|\sum_P \alpha_P \mathrm{Tr}(P\rho(\chi_P(x))) - \sum_P \alpha_P \mathrm{Tr}(P\rho(x))\right| \leq \frac{\epsilon_1}{4}. \tag{C.33}$$

This holds when choosing the local radius δ_1 (defined in Equation (A.5)) to be $\delta_1 = 4C\log^2(1/\epsilon_1)$ for some constant C. This implies that $\tilde{m} = |I_P| = \mathcal{O}(\mathrm{polylog}(1/\epsilon_1))$ (e.g., Equation (S33) of [2]). Thus, for the model $f^{\Theta',w'}$ with architecture defined in Definition C.1 and weights $w'_P = \alpha_P$ and $\Theta' = \{\theta'_P\}_P$, we have

$$|f^{\Theta',w'}(x) - \mathrm{Tr}(O\rho(x))| \tag{C.34}$$

$$= \left|\sum_P \alpha_P f_P^{\theta'_P}(x) - \sum_P \mathrm{Tr}(P\rho(\chi_P(x))) + \sum_P \mathrm{Tr}(P\rho(\chi_P(x))) - \sum_P \mathrm{Tr}(P\rho(x))\right| \tag{C.35}$$

$$\leq \sum_P |\alpha_P| \frac{\epsilon_1}{4D} + \left|\sum_P \mathrm{Tr}(P\rho(\chi_P(x))) - \sum_P \mathrm{Tr}(P\rho(x))\right| \tag{C.36}$$

$$\leq \frac{\epsilon_1}{4} + \left|\sum_P \mathrm{Tr}(P\rho(\chi_P(x))) - \sum_P \mathrm{Tr}(P\rho(x))\right| \tag{C.37}$$

$$\leq \frac{\epsilon_1}{2}. \tag{C.38}$$

Moreover, by definition of the training data, we have $|y_\ell - \mathrm{Tr}(O\rho(x_\ell))| \leq \epsilon_2$. Thus, by triangle inequality and choosing regularization parameter $\lambda = \epsilon_1^2/(2D)$, we have

$$\frac{1}{N}\sum_{\ell=1}^N |f^{\Theta',w'}(x_\ell) - y_\ell|^2 + \lambda\|w'\|_1 \tag{C.39}$$

$$= \frac{1}{N}\sum_{\ell=1}^N |f^{\Theta',w'}(x_\ell) - \mathrm{Tr}(O\rho(x_\ell)) + \mathrm{Tr}(O\rho(x_\ell)) - y_\ell|^2 + \lambda\|w'\|_1 \tag{C.40}$$

$$\leq (\epsilon_1/2 + \epsilon_2)^2 + \lambda\|w'\|_1 \tag{C.41}$$

$$\leq \left(\frac{\epsilon_1}{2} + \epsilon_2\right)^2 + \frac{\epsilon_1^2}{2} \tag{C.42}$$

$$\leq (\epsilon_1 + \epsilon_2)^2. \tag{C.43}$$

Finally, plugging in $\tilde{m} = \mathcal{O}(\mathrm{polylog}(1/\epsilon_1))$, by Lemma C.6, then we obtain $|\Theta_i| = 2^{\mathcal{O}(\mathrm{polylog}(1/\epsilon_1))}$, as required. □

C.2 Prediction error bound

In this section, we derive our result on the prediction error to complete the proof of Theorem C.3. The central result we use is the Koksma-Hlawka inequality (Theorem A.12) from quasi-Monte Carlo theory, which produces a bound in terms of the star-discrepancy (Definition A.8) and the Hardy-Krause variation

(Equation (A.36)). We review these tools in Section A.2. To bound the star-discrepancy, we consider a specific low-discrepancy sequence with guarantees described in Section A.2. The Hardy-Krause variation can be bounded by considering the mixed derivatives of our target function $\mathrm{Tr}(O\rho(x))$ and our neural network model. We relegate the bounds on the mixed derivatives of $\mathrm{Tr}(O\rho(x))$ to Section C.4, as the discussion is fairly technical. To bound the mixed derivatives of our model, we consider the following lemma.

Lemma C.9 (Bound on mixed derivatives of neural network)**.** *Let $k, d \in \mathbb{N}$. Let $\hat{f} : [-1,1]^d \to \mathbb{R}$ be a tanh neural network with two hidden layers of width $W \geq d$ and maximal weight $W_{\max}$. Then*

$$\|\hat{f}\|_{W^{k,\infty}([-1,1]^d)} = 2^{\mathcal{O}(k^2 \log(k) + k \log(dWW_{\max}))}. \tag{C.44}$$

Proof. Recall that a tanh deep neural network with two hidden layers is defined as a function $\hat{f} : [-1,1]^d \to \mathbb{R}$ such that

$$\hat{f}(x) = (W_{\text{out}} \circ \tanh \circ W_{\text{hidden}} \circ \tanh \circ W_{\text{in}})(x), \tag{C.45}$$

where the activation function tanh is applied element-wise. Note that this result holds for any tanh neural network with two hidden layers, where this neural network does not necessarily have to be the same model as Definition C.1.

Let $W_L \in \{W_{\text{in}}, W_{\text{hidden}}, W_{\text{out}}\}$ denote the layers of the neural network that perform an affine transformation for $L \in \{\text{in}, \text{hidden}, \text{out}\}$. We can also use $L \in \{0,1,2\}$, where 0 corresponds to in, 1 corresponds to hidden, and 2 corresponds to out. Let d_L denote the width (number of input neurons) in each layer, where we define $d_0 = d_{\text{in}}, d_1 = d_2 = W, d_3 = d_{\text{out}}$. In this way, $W_L : \mathbb{R}^{d_L} \to \mathbb{R}^{d_{L+1}}$ for $L \in \{0,1,2\}$. Explicitly, we have $W_{\text{in}} : \mathbb{R}^{d_{\text{in}}} \to \mathbb{R}^W$, $W_{\text{hidden}} : \mathbb{R}^W \to \mathbb{R}^W$, $W_{\text{out}} : \mathbb{R}^W \to \mathbb{R}^{d_{\text{out}}}$.

Since W_L performs an affine transformation, we can write it has $W_L(x) = (f_1(x), \ldots, f_{d_{L+1}}(x))$, where $x \in \mathbb{R}^{d_L}$ and f_i are linear functions $f_i(x) = w_i^\intercal x + b_i$ for $w_i \in \mathbb{R}^{d_L}, b_i \in \mathbb{R}$. For these linear layers, we observe for any function $g : \mathbb{R}^{d_g} \to \mathbb{R}^{d_L}$ with input dimension d_g and for $L \in \{0,1,2\}$, we have

$$\begin{aligned}
\max_{1 \leq i \leq d_{L+1}} \|(W_L \circ g)_i\|_{W^{k,\infty}([-1,1]^d)} &= \max_{1 \leq i \leq d_{L+1}} \|f_i(g(x))\|_{W^{k,\infty}([-1,1]^d)} && \text{(C.46)} \\
&\leq \sum_{i=1}^{d_{L+1}} \|w_i^T g(x) + b_i\|_{W^{k,\infty}([-1,1]^d)} && \text{(C.47)} \\
&\leq \max_{1 \leq j \leq d_g} \|W_L\|_1 \|g(x)_j\|_{W^{k,\infty}([-1,1]^d)}, && \text{(C.48)}
\end{aligned}$$

where we use the notation

$$\|W_L\|_1 \triangleq \sum_{i=1}^{d_{L+1}} \left(|b_i| + \sum_{j=1}^{d_L} |w_{i,j}| \right), \tag{C.49}$$

where w is a matrix with rows given by the vectors $w_i^\intercal$, $w_i \in \mathbb{R}^{d_L}$. To show this inequality, we used Hölder's inequality in the last step. With this, by factoring

out one layer W_L at a time, we can bound the Sobolev norm of $\hat{f}$. In particular, we have

$$\|\hat{f}\|_{W^{k,\infty}([-1,1]^d)} \tag{C.50}$$
$$=\|(W_{\text{out}} \circ \tanh \circ W_{\text{hidden}} \circ \tanh \circ W_{\text{in}})(x)\|_{W^{k,\infty}([-1,1]^d)} \tag{C.51}$$
$$\leq \|W_{\text{out}}\|_1 \max_{1\leq j\leq W} \|(\tanh \circ W_{\text{hidden}} \circ \tanh \circ W_{\text{in}})_j\|_{W^{k,\infty}([-1,1]^d)} \tag{C.52}$$
$$\leq \|W_{\text{out}}\|_1 16(e^2k^4d^2)^k(2k)^{k(k+1)} \max_j \|(W_{\text{hidden}} \circ \tanh \circ W_{\text{in}})_j\|^k_{W^{k,\infty}([-1,1]^d)} \tag{C.53}$$
$$\leq \|W_{\text{out}}\|_1 \|W_{\text{hidden}}\|_1^k \cdot 16(e^2k^4d^2)^k(2k)^{k(k+1)} \max_j \|(\tanh \circ W_{\text{in}})_j\|^k_{W^{k,\infty}([-1,1]^d)} \tag{C.54}$$
$$\leq \|W_{\text{out}}\|_1 \|W_{\text{hidden}}\|_1^k \cdot 16^2(e^2k^4d^2)^{2k}(2k)^{2k(k+1)} \|W_{\text{in}}\|_1^k. \tag{C.55}$$

In the second line, we used Equation (C.48). In the third line, we used the two following inequalities:

$$\left| \frac{d^m}{dx^m} \tanh(x) \right| \leq (2m)^{m+1} \min\{\exp(-2x), \exp(2x)\} \tag{C.56}$$

for all $x \in \mathbb{R}, m \in \mathbb{N}$ (see Lemma A.4 in [57]), and

$$\|g \circ f\|_{W^{n,\infty}} \leq 16(e^2n^4md^2)^n \|g\|_{W^{n,\infty}} \max_{1\leq i\leq m} \|(f)_i\|^n_{W^{n,\infty}} \tag{C.57}$$

for any functions $f \in C^n(\Omega_1; \Omega_2)$ and $g \in C^n(\Omega_2; \mathbb{R})$ defined on $\Omega_1 \subset \mathbb{R}^d$, $\Omega_2 \subset \mathbb{R}^m$ with $d, m, n \in \mathbb{N}$ (see Lemma A.7 in [57]). In the fourth and fifth lines, we used Equation (C.48) and these inequalities again. Furthermore, we used that our inputs are absolutely bounded by 1 in the last step.

We can further bound this term using that $W_{\max}$ is the maximal weight of $\hat{f}$ and the width of the hidden layers is lower bounded by $W \geq d$.

$$\|\hat{f}\|_{W^{k,\infty}([-1,1]^d)} \leq 16^2(e^2k^4d^2)^{2k}(2k)^{2k(k+1)} W_{\max}^{2k+1} W^{3k+1} d^k \tag{C.58}$$
$$= 2^{\mathcal{O}(k(k\log(k)+\log(dWW_{\max})))}. \tag{C.59}$$

□

Now we have all the necessary tools in order to derive a bound on the generalization error for our neural network model of the form given in Definition C.1. In our proof, we first bound the prediction error in terms of functions with $2\tilde{m}$-dimensional domain and on which we can directly apply the Koksma-Hlawka inequality. Then, we use the previous result to obtain an explicit bound. Due to the regularity of the parameters α_P and our model parameters w_P, this prediction error bound is independent of the system size n.

Before stating the formal result bounding the prediction error, we introduce some notation. We define the prediction error of a neural network $f^{\Theta,w}$ with weights given by Θ, w as

$$R(\Theta) \triangleq \mathop{\mathbb{E}}_{x\sim U[-1,1]^m} |f^{\Theta,w}(x) - \text{Tr}(O\rho(x))|^2, \tag{C.60}$$

where in our case, $x \sim U[-1,1]^m$ denotes x sampled from a uniform distribution over $[-1,1]^m$. We suppress w in the notation to avoid cluttering. For a neural network $f^{\Theta,w}$ generated from training on some data $\{(x_\ell, y_\ell)\}_{\ell=1}^N$, we can define the training error as

$$\hat{R}(\Theta) \triangleq \frac{1}{N}\sum_{\ell=1}^{N} |f^{\Theta,w}(x_\ell) - y_\ell|^2. \tag{C.61}$$

Moreover, as in our analysis in Section C.1, we rely on an approximation of the ground state property $\mathrm{Tr}(O\rho(x))$ by a sum of smooth local functions $\sum_P \alpha_P f_P(x)$ (Lemma A.3). Namely, combining Lemma A.3 and Theorem A.5, we have that for $\epsilon_1 > 0$, then choosing $\delta_1 > 0$ as in Equation (A.5), i.e., $\delta_1 = \mathcal{O}(\log^2(1/\epsilon_1))$,

$$\left|\sum_P \alpha_P f_P(x) - \mathrm{Tr}(O\rho(x))\right| \leq \frac{\epsilon_1}{2} \tag{C.62}$$

Note that here, again, we slightly alter the definition from Section A.1.2, where we do not include the coefficient α_P in the definition of $f_P(x)$. With these definitions, we have the following guarantee on the prediction error.

Lemma C.10 (Prediction error bound)**.** *Let $1/e > \epsilon_1, \epsilon_2 > 0$. Consider a tanh neural network $f^{\Theta,w} : [-1,1]^m \to \mathbb{R}$ with architecture defined in Definition C.2 with weights $\Theta_i \leq W_{\max}$ for some $W_{\max} > 0$ independent of the system size n and weights w in the last layer. Suppose we train $f^{\Theta,w}$ on data $\{(x_\ell, y_\ell)\}_{\ell=1}^N$ of size N, where the x_ℓ's form a low-discrepancy sequence with star-discrepancy D_N^* and $|y_\ell - \mathrm{Tr}(O\rho(x_\ell))| \leq \epsilon_2$. Then, we have*

$$R(\Theta) \leq \hat{R}(\Theta) + \frac{\epsilon_1^2}{2} + \epsilon_2^2 + (\|w\|_1 + \|w\|_1^2) D_N^* \cdot 2^{\mathcal{O}(\mathrm{polylog}(WW_{\max}/\epsilon_1))}. \tag{C.63}$$

Proof. Recall the definition of our neural network model in Definition C.1. In particular, our model is given by a function $f^{\Theta,w} : [-1,1]^m \to \mathbb{R}$ defined by

$$f^{\Theta,w}(x) = \sum_{P \in S^{(\mathrm{geo})}} w_P f_P^{\theta_P}(x), \tag{C.64}$$

where we refer to $f_P^{\theta_P} : [-1,1]^{\tilde{m}} \to \mathbb{R}$ as the local models. For $f_P(x) = \mathrm{Tr}(P\rho(\chi_P(x)))$ as considered in Equation (C.62), we can define the following quantities. Define the training error with respect to this local approximation by

$$\hat{R}_{\mathrm{loc}}(\Theta) \triangleq \frac{1}{N}\sum_{\ell=1}^{N} \left| f^{\Theta,w}(x_\ell) - \sum_P \alpha_P f_P(x_\ell) \right|^2 \tag{C.65}$$

Also define the prediction error with respect to the local approximation as

$$R_{\mathrm{loc}}(\Theta) \triangleq \mathop{\mathbb{E}}_{x \sim U[-1,1]^m} \left| f^{\Theta,w}(x) - \sum_P \alpha_P f_P(x) \right|^2, \tag{C.66}$$

where $x \sim U[-1,1]^m$ means that x is sampled according to the uniform distribution.

By Lemma A.3, for our choice of δ_1, we have Equation (C.62):

$$\left|\sum_P \alpha_P f_P(x) - \mathrm{Tr}(O\rho(x))\right| \leq \frac{\epsilon_1}{2}. \tag{C.67}$$

By the triangle inequality, we can bound the prediction error as

$$R(\Theta) = \mathop{\mathbb{E}}_{x\sim U[-1,1]^m} \left| f^{\Theta,w}(x) - \sum_P \alpha_P f_P(x) + \sum_P \alpha_P f_P(x) - \mathrm{Tr}(O\rho(x))\right|^2 \tag{C.68}$$

$$\leq R_{\mathrm{loc}}(\Theta) + \frac{\epsilon_1^2}{4}. \tag{C.69}$$

By applying the reverse triangle inequality, we can further bound this as

$$R(\Theta) \tag{C.70}$$

$$\leq \frac{\epsilon_1^2}{4} + \hat{R}_{\mathrm{loc}}(\Theta) + |R_{\mathrm{loc}}(\Theta) - \hat{R}_{\mathrm{loc}}(\Theta)| \tag{C.71}$$

$$= \frac{\epsilon_1^2}{4} + \hat{R}_{\mathrm{loc}}(\Theta) \tag{C.72}$$

$$+ \left| \mathop{\mathbb{E}}_{x\sim U[-1,1]^m} \left| f^{\Theta,w}(x) - \sum_P \alpha_P f_P(x)\right|^2 - \frac{1}{N}\sum_{\ell=1}^N \left| f^{\Theta,w}(x_\ell) - \alpha_P f_P(x_\ell)\right|^2 \right| \tag{C.73}$$

We can expand the term in the expectation/sum as follows

$$\left(\sum_P w_P f_P^{\theta_P}(x) - \alpha_P f_P(x)\right)^2 \tag{C.74}$$

$$= \left(\sum_P w_P f_P^{\theta_P}(x)\right)^2 \tag{C.75}$$

$$- 2\left(\sum_P w_P f_P^{\theta_P}(x)\right)\left(\sum_P \alpha_P f_P(x)\right) + \left(\sum_P \alpha_P f_P(x)\right)^2 \tag{C.76}$$

$$= \sum_{P_1,P_2} w_{P_1} f_{P_1}^{\theta_{P_1}}(x) w_{P_2} f_{P_2}^{\theta_{P_2}}(x) \tag{C.77}$$

$$- 2w_{P_1} f_{P_1}^{\theta_{P_1}}(x)\alpha_{P_2} f_{P_2}(x) + \alpha_{P_1} f_{P_1}(x)\alpha_{P_2} f_{P_2}(x). \tag{C.78}$$

Plugging this into the absolute value term in **??** and upper bounding it with the triangle inequality, we have

$$|R_{\mathrm{loc}}(\Theta) - \hat{R}_{\mathrm{loc}}(\Theta)| \tag{C.79}$$

$$\leq \sum_{P_1,P_2} |w_{P_1}||w_{P_2}| \left| \mathop{\mathbb{E}}_{x\sim U[-1,1]^m} [f_{P_1}^{\theta_{P_1}}(x) f_{P_2}^{\theta_{P_2}}(x)] - \frac{1}{N}\sum_{\ell=1}^{N} f_{P_1}^{\theta_{P_1}}(x_\ell) f_{P_2}^{\theta_{P_2}}(x_\ell) \right|$$

$$+2|w_{P_1}|\alpha_{P_2}| \left| \mathop{\mathbb{E}}_{x\sim U[-1,1]^m} [f_{P_1}^{\theta_{P_1}}(x) f_{P_2}(x)] - \frac{1}{N}\sum_{\ell=1}^{N} f_{P_1}^{\theta_{P_1}}(x_\ell) f_{P_2}(x_\ell) \right|$$

$$+|\alpha_{P_1}||\alpha_{P_2}| \left| \mathop{\mathbb{E}}_{x\sim U[-1,1]^m} [f_{P_1}(x) f_{P_2}(x)] - \frac{1}{N}\sum_{\ell=1}^{N} f_{P_1}(x_\ell) f_{P_2}(x_\ell) \right|. \tag{C.80}$$

Notice that in the expectation over $[-1,1]^m$, we can replace this by an expectation over the set of local parameters, i.e., the parameters with coordinates in $I_{P_1} \cup I_{P_2}$, which we denote by S_{P_1,P_2}. This is because the functions in the expectations are local functions that only depend on these local parameters. The dimension of the domain we integrate over thus becomes independent of the system size n, as $|S_{P_1,P_2}| \leq 2\tilde{m} = 2|I_P|$.

We can now bound this term further using the Koksma-Hlawka inequality (Theorem A.12). We apply a simple variable transformation $\tau(x) = 2x - 1$ so that the domain of $f_P \circ \tau$ becomes $[0,1]^{\tilde{m}}$. Furthermore, we denote the domain associated with S_{P_1,P_2} by $\Omega_{P_1,P_2} \triangleq [0,1]^{|S_{P_1,P_2}|}$. Starting with the first term in Equation (C.80), we obtain

$$\left| \mathop{\mathbb{E}}_{x\sim U[0,1]^m} [f_{P_1}^{\theta_{P_1}}(\tau(\bar{x})) f_{P_2}^{\theta_{P_2}}(\tau(\bar{x}))] - \frac{1}{N}\sum_{\ell=1}^{N} f_{P_1}^{\theta_{P_1}}(\tau(\bar{x}_\ell)) f_{P_2}^{\theta_{P_2}}(\tau(\bar{x}_\ell)) \right| \tag{C.81}$$

$$= \left| \mathop{\mathbb{E}}_{x\sim U(\Omega_{P_1,P_2})} [f_{P_1}^{\theta_{P_1}}(\tau(\bar{x})) f_{P_2}^{\theta_{P_2}}(\tau(\bar{x}))] - \frac{1}{N}\sum_{\ell=1}^{N} f_{P_1}^{\theta_{P_1}}(\tau(\bar{x}_\ell)) f_{P_2}^{\theta_{P_2}}(\tau(\bar{x}_\ell)) \right| \tag{C.82}$$

$$= \left| \int_{S_{P_1,P_2}} f_{P_1}^{\theta_{P_1}}(\tau(\bar{x})) f_{P_2}^{\theta_{P_2}}(\tau(\bar{x})) dx - \frac{1}{N}\sum_{\ell=1}^{N} f_{P_1}^{\theta_{P_1}}(\tau(\bar{x}_\ell)) f_{P_2}^{\theta_{P_2}}(\tau(\bar{x}_\ell)) \right| \tag{C.83}$$

$$\leq D_N^*(2\tilde{m}) V_{HK}\left((f_{P_1}^{\theta_{P_1}} \cdot f_{P_2}^{\theta_{P_2}}) \circ \tau \right), \tag{C.84}$$

where $\bar{x}_\ell = \tau^{-1}(x_\ell)$, such that Equation (C.81) matches the expression referenced in Equation (C.80). Note that we also applied in the last step that the star-discrepancy is increasing with respect to the dimension of the sequence. By application of the chain rule and the Cauchy-Schwartz inequality in the definition of the Hardy-Krause variation (Equation (A.36)), it is easy to see that

$$V_{HK}\left((f_{P_1}^{\theta_{P_1}} \cdot f_{P_2}^{\theta_{P_2}}) \circ \tau \right) \leq 2^{2\tilde{m}} V_{HK}(f_{P_1}^{\theta_{P_1}} \cdot f_{P_2}^{\theta_{P_2}}). \tag{C.85}$$

For all subsets $S' \subseteq S_{P_1,P_2}$, applying the product rule yields

$$\left| \frac{\partial^{|S'|}}{\partial x_{S'}} (f_{P_1}^{\theta_{P_1}} \cdot f_{P_2}^{\theta_{P_2}}) \right| \leq \sum_{A\subseteq S'} \left| \frac{\partial^{|A|}}{\partial x_A} f_{P_1}^{\theta_{P_1}} \right| \left| \frac{\partial^{|S'\setminus A|}}{\partial x_{S'\setminus A}} f_{P_2}^{\theta_{P_2}} \right| \tag{C.86}$$

$$= 2^{\mathcal{O}(\tilde{m}\log(WW_{\max}) + \tilde{m}^2\log(\tilde{m}))}, \tag{C.87}$$

where the last equality follows from applying Lemma C.9 from Section C.4 with $d = k = 2\tilde{m}$ and $|\{A : A \subseteq S'\}| = 2^{2\tilde{m}}$. Here, we are using the notation $\frac{\partial^{|B|}}{\partial x_B}$ to denote the mixed derivative with respect to all parameters $x_i \in B$ for some set B. Thus, applying Lemma C.9 again, we obtain

$$V_{HK}(f_{P_1}^{\theta_{P_1}} \cdot f_{P_2}^{\theta_{P_2}}) \leq \sum_{S' \subseteq S_{P_1,P_2}} \left| \frac{\partial^{|S'|}}{\partial x_{S'}} (f_{P_1}^{\theta_{P_1}} \cdot f_{P_2}^{\theta_{P_2}}) \right| = 2^{\mathcal{O}(\tilde{m} \log(WW_{\max}) + \tilde{m}^2 \log(\tilde{m}))}. \tag{C.88}$$

Thus, putting it all together, we see that

$$\left| \mathop{\mathbb{E}}_{x \sim U[-1,1]^m} [f_{P_1}^{\theta_{P_1}}(x) f_{P_2}^{\theta_{P_2}}(x)] - \frac{1}{N} \sum_{\ell=1}^{N} f_{P_1}^{\theta_{P_1}}(x_\ell) f_{P_2}^{\theta_{P_2}}(x_\ell) \right| \tag{C.89}$$

$$\leq 2^{2\tilde{m}} D_N^*(2\tilde{m}) 2^{\mathcal{O}(\tilde{m} \log(WW_{\max}) + \tilde{m}^2 \log(\tilde{m}))}. \tag{C.90}$$

The remaining terms in Equation (C.80) can be bounded similarly using Lemma C.27 from Section C.4. This lemma is applicable to f_P because the derivatives are with respect to the local parameters. In this way, we can upper bound Equation (C.80) by

$$|R_{\text{loc}}(\Theta) - \hat{R}_{\text{loc}}(\Theta)| \tag{C.91}$$

$$\leq \sum_{P_1,P_2} D_N^*(2\tilde{m})(|w_{P_1}||w_{P_2}| + |w_{P_1}||\alpha_{P_2}|) 2^{\mathcal{O}(\tilde{m} \log(WW_{\max}) + \tilde{m}^2 \log(\tilde{m}))} \tag{C.92}$$

$$+ \sum_{P_1,P_2} D_N^*(2\tilde{m})|\alpha_{P_1}||\alpha_{P_2}| 2^{\mathcal{O}(\tilde{m} \log(\tilde{m}))}. \tag{C.93}$$

Plugging this back in to Equation (C.71), we have

$$R(\Theta) \leq \frac{\epsilon_1^2}{4} + \hat{R}_{\text{loc}}(\Theta) \tag{C.94}$$

$$+ \sum_{P_1,P_2} D_N^*(2\tilde{m})(|w_{P_1}||w_{P_2}| + |w_{P_1}||\alpha_{P_2}|) 2^{\mathcal{O}(\tilde{m} \log(WW_{\max}) + \tilde{m}^2 \log(\tilde{m}))} \tag{C.95}$$

$$+ \sum_{P_1,P_2} D_N^*(2\tilde{m})|\alpha_{P_1}||\alpha_{P_2}| 2^{\mathcal{O}(\tilde{m} \log(\tilde{m}))}. \tag{C.96}$$

Lastly, we can bound $\hat{R}_{\text{loc}}(\Theta) \leq \epsilon_1^2/4 + \epsilon_2^2 + \hat{R}(\Theta)$ in the same way as in Equation (C.68):

$$\hat{R}_{\text{loc}}(\Theta) = \frac{1}{N} \sum_{\ell=1}^{N} \left| f^{\Theta,w}(x_\ell) - \sum_P \alpha_P f_P(\chi_P(x_\ell)) \right|^2 \tag{C.97}$$

$$\leq \frac{1}{N} \sum_{\ell=1}^{N} \left| f^{\Theta,w}(x_\ell) - y_\ell \right|^2 + |y_\ell - \text{Tr}(O\rho(x_\ell))|^2 \tag{C.98}$$

$$+ \left| \text{Tr}(O\rho(x_\ell)) - \sum_P \alpha_P f_P(\chi_P(x_\ell)) \right|^2 \tag{C.99}$$

$$\leq \hat{R}(\Theta) + \epsilon_2^2 + \frac{\epsilon_1^2}{4}. \tag{C.100}$$

Inserting $\tilde{m} = |I_P| = \mathcal{O}\left(\text{polylog}\left(1/\epsilon_1\right)\right)$ and using that $\sum_P |\alpha_P| = \mathcal{O}(1)$ (Theorem A.5) yields

$$R(\Theta) \leq \hat{R}(\Theta) + \frac{\epsilon_1^2}{2} + \epsilon_2^2 + (\|w\|_1 + \|w\|_1^2) D_N^*(2\tilde{m}) 2^{\mathcal{O}(\text{polylog}(WW_{\max}/\epsilon_1))}. \tag{C.101}$$

□

Using the previous result and the results from low-discrepancy theory (see Section A.2 for a review), we can now show that Algorithm 1 will, under mild assumptions for training, output a model, which yields low prediction error. Thus, using Lemma C.10, we can easily prove Theorem C.3.

Proof of Theorem C.3. By Theorem A.10, we know that for Sobol sequences in base 2 with points in $[0,1]^d$, the star-discrepancy is bounded by

$$D_N^*(d) \leq C(d) \frac{\log(N)^d}{N}, \tag{C.102}$$

where $C(d)$ is a constant such that

$$C(d) < \frac{1}{d!} \left(\frac{d}{\log(2d)} \right). \tag{C.103}$$

Since $C(d) = o(1)$, there exists a constant C, such that $C \geq C(d)$ for all $d > 0$. In our case, $d = 2\tilde{m}$, so we have

$$D_N^*(2\tilde{m}) \leq C \frac{\log(N)^{2\tilde{m}}}{N}. \tag{C.104}$$

Using the assumption that the training objective is not larger than $((\epsilon_1 + \epsilon_2)^2 + \epsilon_3)/2$, by Lemma C.10, we have

$$R(\Theta^*) = \mathop{\mathbb{E}}_{x \sim U[-1,1]^m} |f^{\Theta^*, w^*}(x) - \text{Tr}(O\rho(x))|^2 \tag{C.105}$$

$$\leq \frac{\epsilon_1^2}{2} + \epsilon_2^2 + \frac{(\epsilon_1 + \epsilon_2)^2 + \epsilon_3}{2} + C' \frac{\log(N)^{\text{polylog}(1/\epsilon_1)} 2^{\mathcal{O}(\text{polylog}(W_{\max}/\epsilon_1))}}{N} \tag{C.106}$$

$$\leq 2(\epsilon_1 + \epsilon_2)^2 + \frac{\epsilon_3}{2} + C' \frac{\log(N)^{\text{polylog}(1/\epsilon_1)} 2^{\mathcal{O}(\text{polylog}(W_{\max}/\epsilon_1))}}{N}, \tag{C.107}$$

where C' is a constant. We also used here that $\tilde{m} = |I_P| = \mathcal{O}(\text{polylog}(1/\epsilon_1)$. Since the training data has size $N = \mathcal{O}\left(2^{\text{polylog}(1/\epsilon_1) + \text{polylog}(1/\epsilon_3)}\right)$, $W_{\max}$ can be chosen with respect to ϵ_1, ϵ_3 and independent of the system size n such that

$$C' \frac{\log(N)^{\text{polylog}(1/\epsilon_1)} 2^{\mathcal{O}(\text{polylog}(W_{\max}/\epsilon_1))}}{N} \leq \frac{\epsilon_3}{4}. \tag{C.108}$$

In this way, we obtain

$$R(\Theta^*) \leq 2(\epsilon_1 + \epsilon_2)^2 + \epsilon_3. \tag{C.109}$$

□

Since the training objective from Definition C.2 is non-convex, we cannot guarantee that our algorithm converges to a neural network with low training error. However, the assumptions made in Theorem C.3 are rather mild in practice. Small training errors are a well-known phenomenon in deep learning and usually come at the expense of a larger prediction error, which is referred to as *overfitting*. Overfitting may arise due to excessive model complexity [90], i.e. too many trainable parameters. This is reflected by *Lemma C.*10, since the generalization error increases with the width W of the layers. The major challenge in practice lies in finding an appropriate balance between achieving a small training objective and model complexity, rather than only the latter. Furthermore, when the inputs are regularized, the weights usually remain small during training when initialized properly. This was for example observed in [57].

Finally, it is worth noting that in a scenario with a constant number of parameters $m = \mathcal{O}(1)$, similar to the setup in [51], the expression derived from the outcome in Lemma C.10 exhibits nearly linear dependence on ϵ. When incorporating the constant number of parameters by setting $\tilde{m} = m$, we recover the exact ground state properties $\mathrm{Tr}(P\rho(x))$ in f_P. Thus, ϵ_1 in Lemma C.10 becomes 0. Hence, the ability of LDS training to overcome the curse of dimensionality can unfold its full potential, since the domain dimension becomes independent of ϵ and expression Equation (C.96) reduces to a constant multiplied by $D_N^*(2m)$. By Equation (C.104), we obtain $R(\Theta) = \mathcal{O}(\epsilon^{-(1+\delta)})$ for any $\delta > 0$ and ϵ small enough, when the conditions of Theorem 3.3 are fulfilled.

C.3 Prediction on general distributions

In this section, we generalize our results to hold for a wider class of distributions. Recall that our rigorous guarantee proven so far (Theorem C.3) holds when the training data is generated according to a low-discrepancy sequence and the prediction error is measured with respect to the uniform distribution. We want to extend this result for different choices of both training and prediction error distributions. Notice that our prediction error bound (Lemma C.10) is the only place that requires these assumptions on the distributions. Thus, in this section, we establish bounds on the expected prediction error for a more general family of distributions. We consider the following two cases.

1. The training data is generated according to a general low-discrepancy sequence (in the sense of Definitions A.13 and A.14), and the prediction error is measured with respect to some distribution $\mathcal{D}$.

2. The training data consists of independently and identically distributed (i.i.d.) random samples according to a distribution $\mathcal{D}$, and the prediction error is measured with respect to the same distribution $\mathcal{D}$.

There are some conditions on the distributions that we discuss shortly. In Case 1, suppose for example that we want to provide rigorous guarantees on the prediction error when the parameters $x \in [-1, 1]^m$ are sampled from a standard

normal distribution (restricted to $[-1,1]^m$ and normalized appropriately). As normally distributed test samples are more densely populated around the mean and more sparse around the boundary of the input domain, we need to predict more accurately around the mean than close to the boundary. When using a uniform low-discrepancy sequence for training, as in Algorithm 1, the predictive capabilities of our model are not exploited properly. To remedy this, we consider the training data to form a general low-discrepancy sequence, where it is low-discrepancy with respect to a normal distribution. We can relate this general low-discrepancy sequence to an LDS with respect to the *Lebesgue measure*, which are the sequences considered in Section C.2, via the probability integral transform (see, e.g., [91]). We sometimes refer to LDS with respect to the Lebesgue measure as *uniform* low-discrepancy sequences. Formally, for any random variable X, which follows some probability distribution $P(X \geq x) \triangleq F_X(x)$, the random variable $Y = F_X(X)$ follows a uniform distribution. It turns out that the same transformation on LDS produces LDS with respect to other measures than the Lebesgue measure, as illustrated in Figure 5.1. Moreover, under some assumptions on the distribution, we can bound the discrepancy with respect to other measures in terms of the discrepancy with respect to the Lebesgue measure, which we know how to bound as in Section C.2.

In the following, we formalize this argument and adapt it to our problem setting. We refer the reader to Section A.2 to review the necessary concepts of generalized (star-)discrepancy, the Koksma-Hlawka inequality, and related results. Then, we demonstrate that a generalization of Lemma C.10 and Theorem C.3 can be achieved by incorporating these findings with slight adjustments to the proofs.

In Case 2, we consider training data sampled i.i.d. from some distribution $\mathcal{D}$ and prediction error measured with respect to the same distribution $\mathcal{D}$. To obtain a rigorous guarantee on the prediction error in this case, we leverage a probabilistic bound on the discrepancy of uniformly random points from [82]. Utilizing the previously established framework from Case 1, we can bound the discrepancy of points sampled from $\mathcal{D}$ in terms of the discrepancy of uniformly random points. This allows us to establish similar guarantees for Case 2.

Before proving each of these cases, we set up our probabilistic framework and define the Borel measure with respect to which our low-discrepancy sequence is defined. Let $g \triangleq \mathrm{PDF}(\mathcal{D})$ be the probability density function (PDF) of the data distribution and let $G \triangleq \mathrm{CDF}(\mathcal{D})$ be the corresponding cumulative distribution function (CDF). In the following, assume that the PDF g satisfies the following properties.

(a) *Strict positivity:* g has full support on $[-1,1]^m$, i.e., $g(x) > 0$ if $x \in [-1,1]^m$ and $g(x) = 0$ otherwise.

(b) *Continuity:* $g(x)$ is continuously differentiable on $[-1,1]^m$.

(c) *Component-wise independence*: The (random) variables $\vec{x}_i, \vec{x}_j$ upon which different local terms $h_i(\vec{x}_i), h_j(\vec{x}_j)$ of the Hamiltonian depend on,

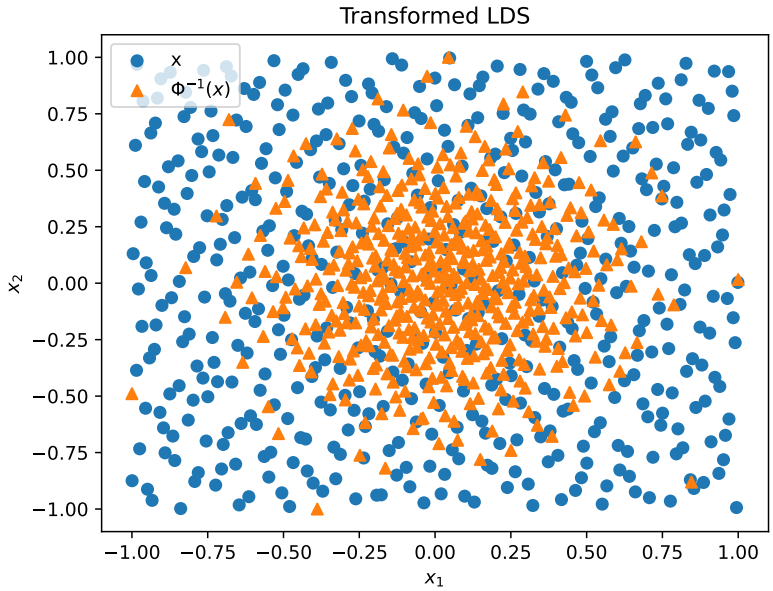

Figure 5.1: **Transformed low-discrepancy sequences.** The blue circles correspond to two-dimensional uniform Sobol points x. The orange triangles indicate the corresponding Sobol points with respect to the CDF of the standard normal distribution, denoted by Φ. The latter forms a low-discrepancy-sequence with respect to the Borel measure $\mu = \Phi$.

are independent. Hence, the PDF g is of the form

$$g(x) = \prod_{j=1}^{L} g_j(\vec{x}_j) \tag{C.110}$$

for PDFs g_j.

We implicitly assume that g also satisfies all properties of a probability density function. It should be noted that Assumptions (a), (b) could technically be relaxed. We expand more on this later. Notice that if $g : [-1,1]^m \to \mathbb{R}$ meets these requirements, the same holds for $\bar{g} \triangleq g \circ \tau : [0,1]^m \to \mathbb{R}$. Here, we use the notation from the previous section, where a bar denotes that we are working in the domain $[0,1]^m$ as opposed to $[-1,1]^m$, and $\tau(\bar{x}) = 2\bar{x} - 1$. Since the available results hold on $[0,1]^m$, we will mostly work with $\bar{g}$ and the corresponding CDF $\bar{G}$.

We continue to set up the necessary notation to formally state our prediction error bound for Case 1. Let S_{P_1,P_2}, Ω_{P_1,P_2} be as in the proof of Lemma C.10. Namely, let S_{P_1,P_2} be the parameters with coordinates in $I_{P_1} \cup I_{P_2}$, where I_P is defined in Equation (A.2), and let $\Omega_{P_1,P_2} = [0,1]^{|S_{P_1,P_2}|}$. Additionally, define $\mu_{P_1,P_2} \triangleq \prod_{j \in S_{P_1,P_2}} \bar{G}_j(\vec{x}_j)$ as the probability measure of the marginal

for all (random) variables with indices in S_{P_1,P_2}. Due to Assumption (c), μ_{P_1,P_2} depends on at most $2\tilde{m}$ variables. Furthermore, we define

$$\mu^* \triangleq \underset{\mu_{P_i,P_j}}{\arg\max}\, D_N(|S_{P_i,P_j}|;\mu_{P_i,P_j}), \tag{C.111}$$

and denote by S^* the corresponding coordinate set. S^* forms the domain of μ^*, and we use d^* to denote the dimension of the domain.

In both Case 1 and Case 2, the idea is to define a transformation F that maps random variables with an arbitrary distribution to uniformly random variables. Namely, we construct a mapping ϕ such that

$$\mathop{\mathbb{E}}_{x\sim\mathcal{D}}[u(x)] = \mathop{\mathbb{E}}_{x\sim U[-1,1]^m}[u(\phi(x))] \tag{C.112}$$

for any function u. In the following, we introduce the transform $F \triangleq \phi^{-1}$, as has been introduced in [89], [92], [93]. F can nicely be characterized using $\bar{g}$ and $\bar{G}$, and assumptions on F are easy to verify for a given data distribution. In fact, if F satisfies a Lipschitz condition, then known results [89] bound the discrepancy with respect to an arbitrary measure in terms of the discrepancy with respect to the Lebesgue measure, i.e., we can directly upper-bound $D_N(d^*;\mu^*)$ in terms of $D_N(d^*)$. Our prediction error bound for more general distributions follows from this result and the results from Section C.2.

Let g^* be defined such that $d\mu^*(x) = g^*(x)dx$. Also, let $A, B \subseteq S^*$ be such that $A \cap B = \emptyset$ and $C = S^* \setminus (A \cup B)$. Then, we define the conditional marginal PDF as

$$g^*(x_A|X_B = x_B) \triangleq \frac{\int_{[0,1]^{|C|}} g^*(x)dx_C}{\int_{[0,1]^{|A|+|C|}} \int_0^{(x_B)_1} \cdots \int_0^{(x_B)_{|B|}} g^*(x)dx} \tag{C.113}$$

and the corresponding CDF as

$$G^*(X_A = x_A|X_B = x_B) \triangleq \int_0^{(x_A)_1} \cdots \int_0^{(x_A)_{|A|}} g^*(x_A|x_B)dx_A. \tag{C.114}$$

For convenience, we refer to the indices of x in S^* via $x_1, x_2, \ldots, x_{d^*}$. We can do this without loss of generality by permuting the order of the parameters. Using these definitions, we can now define the reverse transformation as $F : [0,1]^{d^*} \to [0,1]^{d^*}$, where the indices of F are given by

$$F_j(x) \triangleq G^*(X_j = x_j|X_1 = x_1, \ldots, X_{j-1} = x_{j-1}). \tag{C.115}$$

If random variables are distributed as $X \sim G^*$, then $F(X) \sim U[-1,1]^{d^*}$ (or equivalently $U[0,1]^{d^*}$ under the variable transformation $\tau(x) = 2x - 1$), since X_1, $X_2|X_1$, …, $X_{d^*}|X_1, \ldots, X_{d^*-1}$ are independent and

$$\prod_j F_j(X) = G^*(X). \tag{C.116}$$

Finally, with this notation set up, we can formally state our result for Case 1.

Corollary C.11 (Neural network sample complexity guarantee; generalization of Theorem C.3)**.** *Let $1/e > \epsilon_1, \epsilon_2, \epsilon_3 > 0$, and let $\mathcal{D}$ be a distribution with PDF g fulfilling assumptions (a)-(c) and F according to Equation* (C.115)*. Let $f^{\Theta^*, w^*} : [-1,1]^m \to \mathbb{R}$ be a neural network model produced from Algorithm 1 trained on data $\{(\hat{x}_\ell, \hat{y}_\ell)\}_{\ell=1}^N$ of size*

$$N = 2^{\mathcal{O}(\mathrm{polylog}(1/\epsilon_1)+\mathrm{polylog}(1/\epsilon_3))}, \tag{C.117}$$

where the x_ℓ's form a low-discrepancy Sobol sequence, $\hat{x}_\ell = F^{-1}(x_\ell)$ and $|\hat{y}_\ell - \mathrm{Tr}(O\rho(\hat{x}_\ell))| \leq \epsilon_2$. Suppose that f^{Θ^*, w^*} achieves a training error of at most $((\epsilon_1+\epsilon_2)^2 + \epsilon_3)/2$. Additionally, suppose that all parameters Θ_i^* of f^{Θ^*, w^*} satisfy $|\Theta_i^*| \leq W_{\max}$, for some $W_{\max} > 0$ that is independent of the system size n. Then the neural network f^{Θ^*, w^*} achieves prediction error

$$\mathop{\mathbb{E}}_{x\sim\mathcal{D}} |f^{\Theta^*, w^*}(x) - \mathrm{Tr}(O\rho(x))|^2 \leq 2(\epsilon_1+\epsilon_2)^2 + \epsilon_3. \tag{C.118}$$

Similarly, we also have a guarantee for Case 2, which is the version we state in the main text and the beginning of this appendix.

Corollary C.12 (Neural network sample complexity guarantee; generalization of Corollary C.11 for random data)**.** *Let $1/e > \epsilon_1, \epsilon_2, \epsilon_3 > 0$, $\mathcal{D}$ a distribution with PDF g satisfying assumptions (a)-(c). Let $f^{\Theta^*, w^*} : [-1,1]^m \to \mathbb{R}$ be a neural network model produced from 1 trained on data $\{(x_\ell, y_\ell)\}_{\ell=1}^N$ of size*

$$N = \sqrt{\log(1/\delta)} 2^{\mathcal{O}(\mathrm{polylog}(1/\epsilon_1)+\mathrm{polylog}(1/\epsilon_3))}, \tag{C.119}$$

where the $x_\ell \sim \mathcal{D}$ and $|y_\ell - \mathrm{Tr}(O\rho(x_\ell))| \leq \epsilon_2$. Suppose that f^{Θ^*, w^*} achieves a training error of at most $((\epsilon_1+\epsilon_2)^2 + \epsilon_3)/2$. Additionally, suppose that all parameters Θ_i^* of f^{Θ^*, w^*} satisfy $|\Theta_i^*| \leq W_{\max}$, for some $W_{\max} > 0$ that is independent of the system size n. Then the neural network f^{Θ^*, w^*} achieves prediction error

$$\mathop{\mathbb{E}}_{x\sim\mathcal{D}} |f^{\Theta^*, w^*}(x) - \mathrm{Tr}(O\rho(x))|^2 \leq 2(\epsilon_1+\epsilon_2)^2 + \epsilon_3, \tag{C.120}$$

with probability at least $1-\delta$.

We first prove Corollary C.11 similarly to how we proved Theorem C.3. In particular, we can prove a generalized version of Lemma C.10, where we are given a low-discrepancy sequence with respect to μ^* and wish to bound the prediction error with respect to $\mathcal{D}$, as in Case 1. Define the prediction error of a neural network $f^{\Theta,w}$ with weights given by Θ, w with respect to a distribution $\mathcal{D}$ as

$$R_{\mathcal{D}}(\Theta) \triangleq \mathop{\mathbb{E}}_{x\sim\mathcal{D}} |f^{\Theta,w}(x) - \mathrm{Tr}(O\rho(x))|^2 = \int_{[-1,1]^m} |f^{\Theta,w}(x) - \mathrm{Tr}(O\rho(x))|^2 \, dG(x), \tag{C.121}$$

where $x \sim \mathcal{D}$ denotes x sampled from the distribution $\mathcal{D}$ over $[-1,1]^m$ and $dG(x) = g(x)dx$. Again, we suppress w in the notation to avoid cluttering. Then, we have the following lemma.

Lemma C.13 (Generalized prediction error bound)**.** *Let $1/e > \epsilon_1, \epsilon_2 > 0$. Consider a tanh neural network $f^{\Theta,w} : [-1,1]^m \to \mathbb{R}$ with architecture defined in Definition C.2 with weights $\Theta_i \leq W_{\max}$ for some $W_{\max} > 0$ independent of the system size n and weights w in the last layer. Assume that G satisfies assumptions (a)-(c) and $|y_\ell - \mathrm{Tr}(O\rho(x_\ell))| \leq \epsilon_2$. Furthermore, suppose we train $f^{\Theta,w}$ on data $\{(x_\ell, y_\ell)\}_{\ell=1}^N$ of size N, where the $\tau^{-1}(x_\ell)$'s from a set with star-discrepancy at most $D_N^*(d;\mu^*)$ in each dimension d. Then, we have*

$$R_{\mathcal{D}}(\Theta) \leq \hat{R}(\Theta) + \frac{\epsilon_1^2}{2} + \epsilon_2^2 + (\|w\|_1 + \|w\|_1^2) D_N^*(d^*;\mu^*) \cdot 2^{\mathcal{O}(\mathrm{polylog}(WW_{\max}/\epsilon_1))}. \tag{C.122}$$

Moreover, if there exist constants b_1, b_2 such that $D_N^*(d) \leq b_1\sqrt{b_2 + \log(1/\delta)}\sqrt{\frac{d}{N}}$ with probability at least $1-\delta$, then there exists a constant $\tilde{b}_1$ such that

$$R_{\mathcal{D}}(\Theta) \leq \hat{R}(\Theta) + \frac{\epsilon_1^2}{2} + \epsilon_2^2 + u(w)\tilde{b}_1\sqrt{1+\log(1/\delta)}\sqrt{\frac{\tilde{m}}{N}} \cdot 2^{\mathcal{O}(\mathrm{polylog}(WW_{\max}/\epsilon_1))} \tag{C.123}$$

with probability at least $1-\delta$ and $u(w) = (\|w\|_1 + \|w\|_1^2)$.

This lemma can be proven in the same fashion as Lemma C.10 with two minor adjustments. One change is the use of a generalization of the Koksma-Hlawka inequality, which we discuss in Theorem A.15 in Section A.2. To prove the second part of Lemma C.13, we need a technical lemma (Lemma C.16) to handle the probability of failure in the bound on the star-discrepancy. We relegate this to the end of the section, as it is mainly a technicality.

Proof. We proceed in the same way as in Lemma C.10, replacing $\mathcal{R}(\Theta)$ with $\mathcal{R}_{\mathcal{D}}(\Theta)$ and replacing $\mathcal{R}_{\mathrm{loc}}(\Theta)$ with

$$R_{\mathrm{loc},\mathcal{D}}(\Theta) \triangleq \mathop{\mathbb{E}}_{x\sim\mathcal{D}} \left| f^{\Theta,w}(x) - \sum_P \alpha_P f_P(x) \right|^2. \tag{C.124}$$

We follow the proof of Lemma C.10 until Equation (C.80). This gives us

$$R_{\mathcal{D}}(\Theta) \leq \frac{\epsilon_1^2}{4} + \hat{R}_{\mathrm{loc}}(\Theta) + |R_{\mathrm{loc},\mathcal{D}}(\Theta) - \hat{R}_{\mathrm{loc}}(\Theta)|, \tag{C.125}$$

where recall that

$$\hat{R}_{\mathrm{loc}}(\Theta) = \frac{1}{N}\sum_{\ell=1}^N \left| f^{\Theta,w}(x_\ell) - \sum_P \alpha_P f_P(x_\ell) \right|^2, \tag{C.126}$$

as in Equation (C.65). Moreover, we also have the adjusted version of Equation (C.80)

$$\begin{aligned} &|R_{\mathrm{loc},\mathcal{D}}(\Theta) - \hat{R}_{\mathrm{loc}}(\Theta)| \qquad\qquad (C.127)\\ &\leq \sum_{P_1,P_2} |w_{P_1}||w_{P_2}| \left| \mathop{\mathbb{E}}_{x\sim\mathcal{D}}[f_{P_1}^{\theta_{P_1}}(x) f_{P_2}^{\theta_{P_2}}(x)] - \frac{1}{N}\sum_{\ell=1}^N f_{P_1}^{\theta_{P_1}}(x_\ell) f_{P_2}^{\theta_{P_2}}(x_\ell) \right| \end{aligned}$$

$$+2|w_{P_1}||\alpha_{P_2}|\left|\mathop{\mathbb{E}}_{x\sim\mathcal{D}}[f_{P_1}^{\theta_{P_1}}(x)f_{P_2}(x)]-\frac{1}{N}\sum_{\ell=1}^{N}f_{P_1}^{\theta_{P_1}}(x_\ell)f_{P_2}(x_\ell)\right|$$

$$+|\alpha_{P_1}||\alpha_{P_2}|\left|\mathop{\mathbb{E}}_{x\sim\mathcal{D}}[f_{P_1}(x)f_{P_2}(x)]-\frac{1}{N}\sum_{\ell=1}^{N}f_{P_1}(x_\ell)f_{P_2}(x_\ell)\right|. \tag{C.128}$$

To bound the first term, we use the generalized Koksma-Hlawka inequality (Theorem A.15) to obtain

$$\left|\mathop{\mathbb{E}}_{x\sim\mathcal{D}}[f_{P_1}^{\theta_{P_1}}(x)f_{P_2}^{\theta_{P_2}}(x)]-\frac{1}{N}\sum_{\ell=1}^{N}f_{P_1}^{\theta_{P_1}}(x_\ell)f_{P_2}^{\theta_{P_2}}(x_\ell)\right| \tag{C.129}$$

$$=\left|\int_{[-1,1]^m}|f^{\Theta,w}(x)-\mathrm{Tr}(O\rho(x))|^2\,dG(x)-\frac{1}{N}\sum_{\ell=1}^{N}f_{P_1}^{\theta_{P_1}}(x_\ell)f_{P_2}^{\theta_{P_2}}(x_\ell)\right| \tag{C.130}$$

$$=\left|\int_{[0,1]^m}f_{P_1}^{\theta_{P_1}}(\tau(\bar{x}))f_{P_2}^{\theta_{P_2}}(\tau(\bar{x}))\prod_{i=1}^{L}\bar{g}_i(\vec{\bar{x}}_i)d\bar{x}-\frac{1}{N}\sum_{\ell=1}^{N}f_{P_1}^{\theta_{P_1}}(\tau(\bar{x}_\ell))f_{P_2}^{\theta_{P_2}}(\tau(\bar{x}_\ell))\right| \tag{C.131}$$

$$=\left|\int_{\Omega_{P_1,P_2}}f_{P_1}^{\theta_{P_1}}(\tau(\bar{x}))f_{P_2}^{\theta_{P_2}}(\tau(\bar{x}))d\mu_{P_1,P_2}(\bar{x})-\frac{1}{N}\sum_{\ell=1}^{N}f_{P_1}^{\theta_{P_1}}(\tau(\bar{x}_\ell))f_{P_2}^{\theta_{P_2}}(\tau(\bar{x}_\ell))\right| \tag{C.132}$$

$$\leq D_N^*(d^*;\mu^*)V_{HK}\left((f_{P_1}^{\theta_{P_1}}\cdot f_{P_2}^{\theta_{P_2}})\circ\tau\right). \tag{C.133}$$

Here, in the first equality, we use Assumption (c) on the structure of the PDF g and use the variable transformation $\tau(x)=2x-1$. In the second equality, similarly to Lemma C.10, we notice that because the functions in the expectation are local functions that only depend on parameters in S_{P_1,P_2}, we can replace the expectation over the whole domain $[0,1]^m$ with an expectation over just the domain Ω_{P_1,P_2}. This step also crucially uses Assumption (c), where the factorization of the PDF g due to independence is needed. The last line uses the generalized Koksma-Hlawka inequality (Theorem A.15). The remainder of the proof follows in the same way as Lemma C.10.

The second part of the statement is a direct consequence of Lemma C.16. Specifically, following the proof of Lemma C.10, we use Lemma C.16 to bound the term in Equation (C.96). This is necessary because the upper bound on the star-discrepancy only holds probabilistically, so we must show that we can still use this upper bound on the sum of several star-discrepancy terms. This is more complicated than a simple union bound, and we relegate the proof and statement of Lemma C.16 to the end of this section. □

In addition to this lemma, we also need a result from [89], adapted to our definitions above. In the following, we use $D_N(\omega;d)$ to denote the discrepancy with respect to the Lebesgue measure of a specific sequence ω of length N and dimension d, as in Definition A.7. Similarly, we use $D_N(\omega;d;\mu)$ to denote the

discrepancy with respect to a measure μ of a specific sequence ω of length N and dimension d, as in Definition A.13.

Lemma C.14 (Theorem 2 in [89]). *Let $\omega = \{x_\ell\}_{\ell=1}^N$ be an arbitrary sequence on the open d-dimensional unit cube with discrepancy $D_N(\omega, d)$, and let $\hat{\omega} = \{\hat{x}_\ell\}_{\ell=1}^N$ be the sequence defined by $F\hat{x}_\ell = x_\ell$, where F is defined in Equation (C.115). Moreover, let g be a strictly positive, d-times continuously differentiable PDF, such that $g(x) \geq m > 0$ for all x. Let G be the corresponding probability measure (i.e., CDF). Furthermore, let F satisfy*

$$\|F(x) - F(y)\| \leq K\|x - y\|. \tag{C.134}$$

Then, the discrepancy of $\hat{\omega}$ with respect to G is bounded as

$$D_N(\hat{\omega}; d; G) \leq c\left(D_N(\omega; d)\right)^{\frac{1}{d}}, \tag{C.135}$$

where $c = 2d \cdot 3^d (K+1)^{d-1}$.

The authors in [89] note that the assumption on F in Equation (C.134) is certainly fulfilled when g is continuously differentiable. This is where Assumption (b) is used, where technically, we only require this Lipschitz condition on F. With Lemma C.13 and Lemma C.14, we are ready to prove Corollary C.11.

Proof of Corollary C.11. We proceed similarly as in the proof of Theorem C.3 but this time using Lemma C.13 instead of Lemma C.10. First, we bound the nonuniform discrepancy of our training inputs $\hat{\omega} = \{\hat{x}_\ell\}_{\ell=1}^N$. Recall that $\hat{x}_\ell = F^{-1}(x_\ell)$ for x_ℓ generated according to a low-discrepancy Sobol sequence (i.e., low-discrepancy with respect to the Lebesgue measure). By definition of F, then $\hat{\omega}$ has star-discrepancy $D_N^*(\hat{\omega}; d^*; \mu^*)$. By Assumption (b), we can apply Lemma C.14 to obtain

$$D_N^*(\hat{\omega}; d^*; \mu^*) \leq D_N(\hat{\omega}; d^*; \mu^*) \leq c\left(D_N(\omega; d^*)\right)^{\frac{1}{d^*}} \leq 2^{d^*} c\left(D_N^*(\omega; d^*)\right)^{\frac{1}{d^*}}. \tag{C.136}$$

Here, the first inequality follows because $D_N^*(d) \leq D_N(d)$, and the second follows by Lemma C.14. Finally, the last inequality follows from $D_N(d) \leq 2^d D_N^*(d)$ (see, e.g., [94]). Because $d^* \leq 2\tilde{m}$, we can proceed as in the proof of Theorem C.3 using the bound above.

By Theorem A.10, we know that for Sobol sequences in base 2 with points in $[0,1]^{d^*}$, the star-discrepancy is bounded by

$$D_N^*(d^*) \leq C(d^*) \frac{\log(N)^{d^*}}{N}, \tag{C.137}$$

where $C(d)$ is a constant such that

$$C(d) < \frac{1}{d!}\left(\frac{d}{\log(2d)}\right). \tag{C.138}$$

Since $C(d) = o(1)$, there exists a constant C, such that $C \geq C(d)$ for all $d > 0$. Using the assumption that the training objective is not larger than $((\epsilon_1 + \epsilon_2)^2 + \epsilon_3)/2$, by Lemma C.13, we have

$$R_{\mathcal{D}}(\Theta^*) = \mathop{\mathbb{E}}_{x \sim \mathcal{D}} |f^{\Theta^*, w^*}(x) - \mathrm{Tr}(O\rho(x))|^2 \tag{C.139}$$

$$\leq \frac{\epsilon_1^2}{2} + \epsilon_2^2 + \frac{(\epsilon_1+\epsilon_2)^2+\epsilon_3}{2} + C' \frac{\log(N) 2^{\mathcal{O}(\text{polylog}(W_{\max}/\epsilon_1))}}{N^{1/d^*}} \quad \text{(C.140)}$$

$$\leq 2(\epsilon_1+\epsilon_2)^2 + \frac{\epsilon_3}{2} + C' \frac{\log(N) 2^{\mathcal{O}(\text{polylog}(W_{\max}/\epsilon_1))}}{N^{1/\text{polylog}(1/\epsilon_1)}}, \quad \text{(C.141)}$$

where C' is a constant. We also used here that
$d^* \leq 2\tilde{m}$ and $\tilde{m} = |I_P| = \mathcal{O}(\text{polylog}(1/\epsilon_1)$. Since the training data has size $N = \mathcal{O}\left(2^{\text{polylog}(1/\epsilon_1)+\text{polylog}(1/\epsilon_3)}\right)$, $W_{\max}$ can be chosen with respect to ϵ_1, ϵ_3 and independent of the system size n such that

$$C'' \frac{\log(N) 2^{\mathcal{O}(\text{polylog}(W_{\max}/\epsilon_1))}}{N^{1/\text{polylog}(1/\epsilon_1)}} \leq \frac{\epsilon_3}{4} \quad \text{(C.142)}$$

for some constant C''. In this way, we obtain

$$R_{\mathcal{D}}(\Theta^*) \leq 2(\epsilon_1+\epsilon_2)^2 + \epsilon_3. \quad \text{(C.143)}$$

□

Finally, we prove Case 2, where the training data is sampled i.i.d. according to a distribution $\mathcal{D}$ and the prediction error is also measured with respect to $\mathcal{D}$. Note that we can drop the assumption that F^{-1} is efficiently computable for this case. The key result we need for this is a bound on the star-discrepancy for uniformly random points (Lemma A.11).

Proof of Corollary C.12. Let $\hat{x}_\ell = Fx_\ell$, where F is as in Equation (C.115). As stated in [92], [93], F transforms random variables with distribution $\mathcal{D}$ into standard uniform random variables. Hence, similarly to the proof of Corollary C.11, if $\omega = \{x_\ell\}_{\ell=1}^N$ has star-discrepancy $D_N^*(\omega; d^*; \mu^*)$, we can bound it with respect to the discrepancy of $\hat{\omega} = \{\hat{x}_\ell\}_{\ell=1}^N$:

$$D_N^*(\omega; d^*; \mu^*) \leq D_N(\omega; d^*; \mu^*) \leq c \left(D_N(\hat{\omega}; d^*)\right)^{\frac{1}{d^*}} \leq 2^{d^*} c \left(D_N(\hat{\omega}; d^*)\right)^{\frac{1}{d^*}}. \quad \text{(C.144)}$$

Here, again we use $D_N^*(d) \leq D_N(d) \leq 2^d D_N^*(d)$ and Lemma C.14. Then, by Lemma A.11, for uniformly random points, i.e., $\hat{\omega} = \{\hat{x}_\ell\}_{\ell=1}^N$, we have

$$D_N^*(d) \leq 5.7\sqrt{4.9 + \log(1/\delta)}\sqrt{\frac{d}{N}} \quad \text{(C.145)}$$

with probability at least $1-\delta$. The rest of the proof follows in the same way as Corollary C.11, but using the above discrepancy bound. Note that because this discrepancy bound only holds probabilistically, we need to use the second part of Lemma C.13. □

Our prediction error bounds for Case 1 and Case 2 (in Corollaries C.11 and C.12, respectively) look rather similar. However, one can verify that Corollary C.11 only requires about square root of the number of samples Corollary C.12 uses to achieve a certain risk bound with small enough ϵ, but this advantage is hidden in the polylogarithmic factors in the exponent. Hence, low-discrepancy data yields better theoretical guarantees. However, they did not

yield an improvement in our numerical experiments, as discussed in Section D. The size N of the training set seems to be very large for low-discrepancy data to have practical effects. In the main text, we present Corollary C.12 because it is the more general theoretical statement.

In fact, one can also improve the polylogarithmic factors in Corollary C.11 by imposing stronger assumptions on the distribution. In particular, the result in Lemma C.14 seems surprisingly weak at first glance. Multidimensional transformations do, however, constitute a major challenge, since they generally do not preserve properties such as lines remaining straight or parallel. The boxes over which one optimizes in order to compute the discrepancy can thus change severely in shape, which can strongly alter the discrepancy and makes it difficult to analyze. This can result in a rather poor scaling in terms of the discrepancy with respect to the Lebesgue measure and thus N. However, when g fulfills additional assumptions, we obtain a much better dependence on N by directly applying the Koksma-Hlawka inequality (with respect to the Lebesgue measure) to $f \circ \phi$. Unsurprisingly, this is possible when the mixed derivative of $F^{-1} = \phi$ is bounded on $[0, 1]$. This follows, when g's mixed derivative is bounded [89]. We restate this result below.

Lemma C.15 (Theorem 1 in [89]). *Let $\omega = \{x_\ell\}_{\ell=1}^N$ of size N be an arbitrary sequence on the open d-dimensional unit cube with discrepancy $D_N(\omega, d)$ and $\hat{\omega} = \{\hat{x}_\ell\}_{\ell=1}^N$ the sequence defined by $F\hat{x}_\ell = x_\ell$, where F is defined in Equation* (C.115). *Moreover, let g be a strictly positive, d-times continuously differentiable PDF, such that $g(x) \geq m > 0$ for all x. Let G be the corresponding probability measure (i.e., CDF). Furthermore, let F satisfy*

$$\frac{\partial^{|A|} F_j}{\partial x_A} \leq M \qquad 1 \leq j \leq d, \quad A \subseteq \{1, \ldots, d\}. \tag{C.146}$$

Then

$$\left| \int_{[0,1^d]} f(\hat{x}) dG(\hat{x}) - \frac{1}{N} \sum_{\ell=1}^{N} f(\hat{x}_\ell) \right| \leq d! \left(\frac{M}{m} \right)^{2d-1} D_N^*(\omega; d) V_{HK}(f). \tag{C.147}$$

On a high level, the proof works via the observation that the Jacobian of F has g as its determinant, which is strictly positive. Since $F \circ \phi$ is the identity, one can write the Jacobian of $F \circ \phi$ as a linear system of equations with the derivatives of ϕ as solution. Using Cramer's rule and the assumption on F, one can upper bound the derivative of ϕ. Applying this iteratively, one can show via induction that the mixed derivatives of ϕ are also bounded when the mixed derivatives of F are bounded. Note that (as stated in [89]) when $\mathcal{D}$ is a product of independent distributions, i.e. $g(x) = \prod_{i=1}^m g_i(x_i)$ and fulfills assumptions (a)-(c), the conditions for Lemma C.15 are also fulfilled. It is important to emphasize that the additional assumption used in Lemma C.15 yield a much better dependence of ϵ on N. However, this improvement is hidden in the polylogarithmic factors.

We dedicate the last part of this section to the proof the following statement, which we used in the proof of Lemma C.13. At a high level, this shows that

when we have a probabilistic upper bound on the star-discrepancy, we can still upper bound a sum of star-discrepancies with high probability.

Lemma C.16. *Suppose there exist constants b_1, b_2 such that $D_N^*(d) \leq b_1\sqrt{b_2 + \log(1/\delta)}\sqrt{\frac{d}{N}}$ with probability $1-\delta$. Then, there exists a constant $\tilde{b}_1$, such that for any $t > 0$*

$$\Pr\left(\sum_{P_1,P_2 \in S^{(\text{geo})}} (c_1|\alpha_{P_1}||\alpha_{P_2}| + c_2(|w_{P_1}||\alpha_{P_2}| + |w_{P_1}||w_{P_2}|))D_N^*(x_{P_1,P_2}) \geq t\right) \tag{C.148}$$

$$\leq \exp\left(-\frac{Nt^2}{\tilde{b}_1(c_1\|\alpha\|_1^2 + c_2(\|w\|_1\|\alpha\|_1 + \|w\|_1^2))^2}\right), \tag{C.149}$$

where recall S_{P_1,P_2} is the set of parameters with coordinates in $I_{P_1} \cup I_{P_2}$ (Equation (A.2)), $c_1 = 2^{\mathcal{O}(\tilde{m}\log(\tilde{m}))}$ and $c_2 = 2^{\mathcal{O}(\tilde{m}\log(WW_{\max})+\tilde{m}^2\log(\tilde{m}))}$. Thus $D_N^*(x_{P_1,P_2})$ denotes the star-discrepancy of this set of parameters in the training data.

First, we introduce two useful tools for the proof.

Theorem C.17 (Azuma's Inequality for Martingales with Subgaussian Tails; Adapted from Theorem 2 in **shamir2011variantazumasinequalitymartingales**)**.** *Let $Z_1, Z_2, \ldots, Z_n$ be a martingale difference sequence with respect to a sequence $X_1, X_2, \ldots, X_n$, and suppose there are constants $b > 1$, $c_1, \ldots, c_n > 0$, such that for any j and any $t > 0$ it holds that*

$$\Pr(Z_j > t | X_1, \ldots, X_{j-1}) \leq b\exp\left(-\frac{t^2}{c_j^2}\right). \tag{C.150}$$

Then, it holds that

$$\Pr\left(\sum_{j=1}^n Z_j > t\right) \leq \exp\left(-\frac{t^2}{28b\sum_{j=1}^n c_j^2}\right). \tag{C.151}$$

Proof of Theorem C.17. Following the steps of the proof of Theorem 2 in **shamir2011variantazu** but taking the sum over Z_j instead of the empirical average, we obtain for any $s > 0$

$$\Pr\left(\sum_{j=1}^n Z_j > t\right) \leq e^{-st}e^{7bc_n^2s^2}\,\mathbb{E}\left[\prod_{j=1}^n e^{sZ_j}\middle| X_1, \ldots, X_{n-1}\right] \leq e^{-st+7bs^2\sum_{j=1}^n c_j^2}. \tag{C.152}$$

We refer to **shamir2011variantazumasinequalitymartingales** for further details of this calculation. Choosing $s = t/\left(14b\sum_{j=1}^n c_j^2\right)$, the expression

above equals $e^{-t^2/\left(28b\sum_{j=1}^n c_j^2\right)}$, and we get the claim:

$$\Pr\left(\sum_{j=1}^n Z_j > t\right) \leq \exp\left(-\frac{t^2}{28b\sum_{j=1}^n c_j^2}\right). \tag{C.153}$$

□

We also need the following two small lemmas.

Lemma C.18. *Let $\delta > 0$. Let $X : \Omega_X \to \mathcal{X}$ and $Y : \Omega_Y \to \mathcal{Y}$ be independent random variables and $f : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}_+$ be a function such that $\Pr_{XY}(f(X,Y) \geq t) \leq \delta$ for $t > 0$. Then,*

$$\mathbb{E}[f(X,Y)|X] \leq \frac{t}{2} \tag{C.154}$$

with probability at least $1 - 2\delta$.

Proof. First, we show that $\Pr(f(X,Y) \geq t|X) \geq 1/2$ with high probability. Then, we show that this implies the claim by Markov's inequality.

First, suppose for the sake of contradiction

$$\Pr\left(\mathbb{E}[\mathbb{1}\{f(X,Y) \geq t\}|X] \geq \frac{1}{2}\right) > 2\delta.$$

Using the independence of X and Y applying Markov's inequality, we obtain

$$\begin{aligned}
\Pr(f(X,Y) \geq t) &= \mathbb{E}[\mathbb{E}[\mathbb{1}\{f(X,Y) \geq t\}|X]] && \text{(C.155)}\\
&\geq \frac{1}{2}\Pr\left(\mathbb{E}\left[\mathbb{1}\{f(X,Y) \geq t\}|X\right] \geq \frac{1}{2}\right) && \text{(C.156)}\\
&> 2\delta \cdot \frac{1}{2} = \delta, && \text{(C.157)}
\end{aligned}$$

which contradicts our initial assumption. Therefore, with probability at most 2δ (w.r.t. X), $\Pr(f(X,Y) \geq t|X) \geq \frac{1}{2}$ and hence

$$\frac{1}{2} \leq \Pr(f(X,Y) \geq t|X) \leq \frac{\mathbb{E}[f(X,Y)|X]}{t} \tag{C.158}$$

by Markov's inequality. The result follows immediately.

□

Lemma C.19. *Let $j \in [m]$ be a coordinate of the parameters. Then, $|\{P \in S^{(\mathrm{geo})} : j \in I_P\}| = tildem$, where $\tilde{m} = \mathcal{O}(|I_P|)$.*

Proof. Recall that

$$I_P = \{c \in \{1, \ldots, m\} : d_{\mathrm{obs}}(h_{j(c)}, P) \leq \delta_1\}. \tag{C.159}$$

Fixing c instead of P also results in a set of geometrically local terms in a radius δ_1 around a geometrically local term. Hence, the size of the set $\{P \in S^{(\mathrm{geo})} : j \in I_P\}$ also scales as $|I_P|$, which is at most $\tilde{m}$. □

Now we are able to provide a partial proof to Lemma C.16.

Lemma C.20. *Let S_{P_1,P_2} be the set of parameters with coordinates in $I_{P_1} \cup I_{P_2}$ and let $x_{P_1,P_2} \triangleq \{\{x \in S_{P_1,P_2}\}_\ell\}_{\ell=1}^N$ denote the training data set only for these local parameters. If there exist constants b_1, b_2 such that $D_N^*(d) \leq b_1\sqrt{b_2 + \log(1/\delta)}\sqrt{\frac{d}{N}}$ with probability at least $1-\delta$, then*

$$\Pr\left(\sum_{P_2 \in S^{(\mathrm{geo})}} |\alpha_{P_2}| D_N^*(x_{P_1,P_2}) \geq t\right) \leq \exp\left(-\frac{Nt^2}{224\|\alpha\|_2^2(\tilde{m})^2 \exp(b_1) b_2}\right) \tag{C.160}$$

for any $P_1 \in S^{(\mathrm{geo})}$ and any $t > 0$.

Proof. Let $P_1 \in S^{(\mathrm{geo})}$. Define

$$X_j \triangleq \mathbb{E}\left[\sum_{P_2 \in S^{(\mathrm{geo})}} |\alpha_{P_2}| D_N^*(x_{P_1,P_2}) \middle| Y_{j-1}, \ldots, Y_0\right], \tag{C.161}$$

where we omit the dependence $x_{P_1,P_2} = x_{P_1,P_2}(Y_1, \ldots, Y_m)$ and $Y_j \triangleq \{(x_j)_\ell\}_{\ell=1}^N$ and x_j parameterize h_j. We consider all increments, which are not contained in I_{P_1}. Hence, with slight abuse of notation, let index $j = 0$ refer to all coordinates in I_{P_1} and $Y_0 \triangleq \{\{(x_j) : j \in I_{P_1}\}_\ell\}_{\ell=1}^N$. Furthermore, let

$$X_0 \triangleq \mathbb{E}\left[\sum_{P_2 \in S^{(\mathrm{geo})}} |\alpha_{P_2}| D_N^*(x_{P_1,P_2}) \middle| Y_0\right] \tag{C.162}$$

and $Z_1 \triangleq X_1 - X_0$. Clearly, $X_0, \ldots, X_m$ is a martingale sequence and $Z_1, \ldots, Z_m$ the respective martingale difference sequence. Furthermore, note that $X_m = \sum_{P_2 \in S^{(\mathrm{geo})}} |\alpha_{P_2}| D_N^*(x_{P_1,P_2})$ and by definition of Y_0, $j \notin I_{P_1}$ for all $j > 0$. Now, since $D_N^*(x_{P_1,P_2}) \geq 0$ and $|\alpha_{P_2}| D_N^*(x_{P_1,P_2})$ cancel out if $j \notin I_{P_2}$,

$$\begin{aligned} Z_j &\leq \mathbb{E}\left[\sum_{j \in S_2 \setminus I_{P_1}} |\alpha_{P_2}| D_N^*(x_{P_1,P_2}) \middle| Y_{j-1}, \ldots, Y_0\right] && \text{(C.163)} \\ &= \sum_{j \in S_2 \setminus I_{P_1}} |\alpha_{P_2}| \, \mathbb{E}\left[D_N^*(x_{P_1,P_2}) \middle| Y_{j-1}, \ldots, Y_0\right] && \text{(C.164)} \\ &= \sum_{j \in S_2 \setminus I_{P_1}} |\alpha_{P_2}| \, \mathbb{E}\left[D_N^*(x_{P_1,P_2}) \middle| (Y_k)_{k<j, k \in I_{P_1}}\right], && \text{(C.165)} \end{aligned}$$

where $S_2 = \{P_2 \in S^{(\mathrm{geo})} : j \in I_{P_2}\}$ Then, for any $t > 0$, we have

$$\Pr(Z_j \geq t) \leq \Pr\left(\sum_{j \in S_2 \setminus I_{P_1}} |\alpha_{P_2}| \, \mathbb{E}\left[D_N^*(x_{P_1,P_2}) \middle| (Y_k)_{k<j, k \in I_{P_1} \cup I_{P_2}}\right] \geq t\right) \tag{C.166}$$

$$\leq \sum_{j\in S_2} \Pr\left(\mathbb{E}\left[D_N^*(x_{P_1,P_2})\middle|(Y_k)_{k<j,k\in I_{P_1}\cup I_{P_2}}\right] \geq \frac{t}{|\alpha_{P_2}|}\right) \quad \text{(C.167)}$$

$$\leq 2\sum_{j\in S_2} \Pr\left(D_N^*(x_{P_1,P_2}) \geq \frac{t}{2|\alpha_{P_2}|}\right) \quad \text{(C.168)}$$

$$\leq 2\sum_{j\in S_2} \exp(b_1)\exp\left(-\frac{Nt^2}{4|\alpha_{P_2}|^2 b_2\tilde{m}}\right) \quad \text{(C.169)}$$

$$\leq 2|S_2|\exp(b_1)\exp\left(-\frac{Nt^2}{4b_2\tilde{m}\sum_{j\in S_2}|\alpha_{P_2}|^2}\right) \quad \text{(C.170)}$$

$$\leq 2\tilde{m}\exp(b_1)\exp\left(-\frac{Nt^2}{4b_2\tilde{m}\sum_{j\in S_2}|\alpha_{P_2}|^2}\right), \quad \text{(C.171)}$$

In the second line, we use a union bound. In the third line, we use Lemma C.18 with $X = (Y_k)_{k<j,k\in I_{P_1}\cup I_{P_2}}$ and $Y = (Y_0,\dots,Y_{j-1})$. In the fourth line, we use a rearrangement of the probabilistic upper bound on the star-discrepancy. In the last line, we use the definition of $\tilde{m}$. Now, by Theorem C.17, for any $t > 0$

$$\Pr\left(\sum_{P_2\in S^{(\text{geo})}} |\alpha_{P_2}| D_N^*(x_{P_1,P_2}) \geq t\right) \leq \exp\left(-\frac{t^2}{28b'\sum_{i=1}^m c_i^2}\right) \quad \text{(C.172)}$$

with $b' = 2\tilde{m}\exp(b_1)$ and $c_i^2 = \frac{4b_2\tilde{m}}{N}\sum_{P_2\in S^{(\text{geo})}:i\in I_{P_2}}|\alpha_{P_2}|^2$. Furthermore,

$$\sum_{i=1}^m \sum_{P_2\in S^{(\text{geo})}:i\in I_{P_2}} |\alpha_{P_2}|^2 = \sum_{P_2\in S^{(\text{geo})}} |\alpha_{P_2}|^2 \sum_{i=1}^m \mathbb{1}\{i\in I_{P_2}\} \quad \text{(C.173)}$$

$$= \tilde{m}\sum_{P_2\in S^{(\text{geo})}} |\alpha_{P_2}|^2 \quad \text{(C.174)}$$

$$= \tilde{m}\|\alpha\|_2^2. \quad \text{(C.175)}$$

The result follows from this. □

Now, we are finally able to prove Lemma C.16.

Proof of Lemma C.16. We need to bound the weighted sum of the star-discrepancies we consider. This requires an extra step, since the star-discrepancy may vary among the sequences in the sum. Note that simply applying the union bound would result in a $\log(n)$-factor. Luckily, only the sum needs to be small, rather than all individual terms needing to be small at once. Recall that we use S_{P_1,P_2} to denote the set of parameters with coordinates in $I_{P_1}\cup I_{P_2}$. In the following, we use $D_N^*(x_{P_1,P_2})$ to denote the star-discrepancy of $\{x\in S_{P_1,P_2}\}_{\ell=1}^N$, i.e., the training data points restricted to local parameters in S_{P_1,P_2}. We aim to apply Theorem C.17 to $\sum_{P_1,P_2\in S^{(\text{geo})}}|\alpha_{P_1}||\alpha_{P_2}|D_N^*(x_{P_1,P_2})$, $\sum_{P_1,P_2}|w_{P_1}||\alpha_{P_2}|D_N^*(x_{P_1,P_2})$ and $\sum_{P_1,P_2\in S^{(\text{geo})}}|w_{P_1}||w_{P_2}|D_N^*(x_{P_1,P_2}))$.

For illustrative purposes, we only consider the first term for now. We proceed similarly to the proof of Lemma C.20. Define

$$X_j \triangleq \mathbb{E}\left[\sum_{P_1,P_2\in S^{(\mathrm{geo})}} |\alpha_{P_1}||\alpha_{P_2}| D_N^*(x_{P_1,P_2}) \middle| Y_{j-1},\dots,Y_1\right], \tag{C.176}$$

where the expectation is with respect to the inputs $Y_j \triangleq \{(x_j)_\ell\}_{\ell=1}^N$ and x_j parametrize h_j. Furthermore, let

$$X_0 \triangleq \mathbb{E}\left[\sum_{P_1,P_2\in S^{(\mathrm{geo})}} |\alpha_{P_1}||\alpha_{P_2}| D_N^*(x_{S_{P_1,P_2}})\right] \tag{C.177}$$

and $Z_1 \triangleq X_1 - X_0$. Clearly, $X_0,\dots,X_m$ is a martingale sequence and $Z_1,\dots,Z_m$ the respective martingale difference sequence. Furthermore, $X_m = \sum_{P_1,P_2\in S^{(\mathrm{geo})}} |\alpha_{P_1}||\alpha_{P_2}| D_N^*(x_{P_1,P_2})$. Now, since $D_N^*(x_{P_1,P_2}) \geq 0$ and $|\alpha_{P_1}||\alpha_{P_2}| D_N^*(x_{P_1,P_2})$ cancel out if $j \notin I_{P_1} \cup I_{P_2}$,

$$Z_j \geq \mathbb{E}\left[\sum_{P_1,P_2\in S^{(\mathrm{geo})}: j\in I_{P_1}\cup I_{P_2}} |\alpha_{P_1}||\alpha_{P_2}| D_N^*(x_{P_1,P_2}) \middle| Y_{j-1},\dots,Y_1\right] \tag{C.178}$$

$$= \sum_{P_1,P_2\in S^{(\mathrm{geo})}: j\in I_{P_1}\cup I_{P_2}} |\alpha_{P_1}||\alpha_{P_2}|\, \mathbb{E}\left[D_N^*(x_{P_1,P_2}) \middle| Y_{j-1},\dots,Y_1\right] \tag{C.179}$$

$$= 2\sum_{P_1\in S^{(\mathrm{geo})}: j\in I_{P_1}} \sum_{P_2\in S^{(\mathrm{geo})}} |\alpha_{P_1}||\alpha_{P_2}|\, \mathbb{E}\left[D_N^*(x_{P_1,P_2}) \middle| Y_{j-1},\dots,Y_1\right]. \tag{C.180}$$

In the last step, we used the observation that for j to be contained in $I_{P_1}\cup I_{P_2}$, it has to be contained in at least one of the two sets. Hence, we can enumerate the admissible coordinate sets by fixing P_1, such that I_{P_1} contains j and combine it with all I_{P_2}. The factor two arises from doing the same with P_1 when fixing P_2.

Now, for any $t > 0$, letting $S_1 = \{P_1 \in S^{(\mathrm{geo})} : j \in I_{P_1}\}$, we have

$$\Pr(Z_j \geq t) \leq \Pr\left(2\sum_{j\in S_1}\sum_{P_2\in S^{(\mathrm{geo})}} |\alpha_{P_1}||\alpha_{P_2}|\, \mathbb{E}\left[D_N^*(x_{P_1,P_2}) \middle| Y_{j-1},\dots,Y_1\right] \geq t\right) \tag{C.181}$$

$$\leq 2\sum_{j\in S_1} \Pr\left(\sum_{P_2\in S^{(\mathrm{geo})}} |\alpha_{P_2}| D_N^*(x_{P_1,P_2}) \geq \frac{t}{4|\alpha_{P_1}|}\right) \tag{C.182}$$

$$\leq 2\sum_{j\in S_1} \exp\left(-\frac{Nt^2}{16\cdot 224|\alpha_{P_1}|^2\|\alpha\|_2^2(\tilde{m})^2\exp(b_1)b_2}\right) \tag{C.183}$$

$$\leq 2|S_1|\exp\left(-\frac{Nt^2}{16\cdot 224\|\alpha\|_2^2(\tilde{m})^2\exp(b_1)b_2\sum_{j\in S_1}|\alpha_{P_1}|^2}\right) \tag{C.184}$$

$$\leq 2\tilde{m}\exp\left(-\frac{Nt^2}{16\cdot 224\|\alpha\|_2^2(\tilde{m})^2\exp(b_1)b_2\sum_{j\in S_1}|\alpha_{P_1}|^2}\right). \quad \text{(C.185)}$$

In second line, we use the union bound and Lemma C.18 with $X = (Y_k)_{k<j,k\in I_{P_1}\cup I_{P_2}}$ and $Y = (Y_k)_{k>j,k\in I_{P_1}\cup I_{P_2}}$. In the third line, we use Lemma C.20. In the last line, we use the definition of $\tilde{m}$. Applying Theorem C.17 and bounding $\sum_j c_j^2$ exactly as in the proof of Lemma C.20 yields

$$\Pr\left(\sum_{P_1,P_2\in S^{(\mathrm{geo})}}|\alpha_{P_1}||\alpha_{P_2}|D_N^*(x_{P_1,P_2})\geq t\right)\leq\exp\left(-\frac{Nt^2}{\tilde{b}_1\|\alpha\|_2^4}\right). \quad \text{(C.186)}$$

One can similarly repeat this argument for the remaining terms of

$$\sum_{P_1,P_2\in S^{(\mathrm{geo})}}(c_1|\alpha_{P_1}||\alpha_{P_2}|+c_2(|w_{P_1}||\alpha_{P_2}|+|w_{P_1}||w_{P_2}|))D_N^*(x_{P_1,P_2})). \quad \text{(C.187)}$$

Using that $\|\alpha\|_2^2\leq\|\alpha\|_1^2$ and solving for the appropriate δ yields the desired result. □

C.4 Bound on the mixed derivatives

Let $O=\sum_{P\in\{I,X,Y,Z\}^{\otimes n}}\alpha_P P$ be an observable that can be written as a sum of geometrically local observables. In the following, we derive an expression for the mixed partial derivatives of $\mathrm{Tr}(P\rho(x))$, using tools from the spectral flow formalism [63], [64], [65]. This allows us to bound the Hardy-Krause variation (Equation (A.36)) in Section C.2. Let the spectral gap of $H(x)$ be lower bounded by some constant γ for all choices of parameters $x\in[-1,1]^m$. Then, by the spectral flow formalism [63], [64], [65], the directional derivative of a ground state of $H(x)$ in the direction defined by the parameter unit vector $\hat{u}$ is given by

$$\frac{\partial}{\partial\hat{u}}\rho(x)=\hat{u}\cdot\nabla_x\rho(x)=i[D_{\hat{u}}(x),\rho(x)] \quad \text{(C.188)}$$

where

$$D_{\hat{u}}(x)=\int_{-\infty}^{+\infty}W_\gamma(t)e^{itH(x)}\frac{\partial H}{\partial\hat{u}}(x)e^{-itH(x)}dt, \quad \text{(C.189)}$$

and $W_\gamma(t)$ is defined by

$$|W_\gamma(t)|\leq\begin{cases}\frac{1}{2} & 0\leq\gamma|t|\leq\theta,\\ 35e^2(\gamma|t|)^4e^{-\frac{2}{7}\frac{\gamma|t|}{\log^2(\gamma|t|)}} & \gamma|t|>\theta.\end{cases} \quad \text{(C.190)}$$

The parameter θ is chosen to be the largest real solution of

$$35e^2(\gamma|t|)^4\exp\left(-\frac{2}{7}\frac{\gamma|t|}{\log^2(\gamma|t|)}\right)=1/2.$$

This allows to us to obtain an expression of the first order derivative of $\rho(x)$ with respect to some parameter x_k.
Consider the unit vector $\hat{u} = \hat{e}_k \triangleq (0, \dots 0, 1, 0, \dots 0)^T$, where the 1 is in the kth position. Then, the directional derivative in the direction given by e_k is

$$\frac{\partial}{\partial \hat{e}_k}\rho(x) = \hat{e}_k \cdot \nabla_x \rho(x) = \frac{\partial}{\partial x_k}\rho(x) = i[D_{\hat{e}_k}(x), \rho(x)]. \tag{C.191}$$

Hence, we obtain

$$\begin{aligned}\frac{\partial}{\partial x_1}\mathrm{Tr}(P\rho(x)) &= \mathrm{Tr}(P\frac{\partial}{\partial x_1}\rho(x)) && \text{(C.192)}\\ &= i\mathrm{Tr}(P[D_{\hat{e}_1}(x), \rho(x)]) && \text{(C.193)}\\ &= i\mathrm{Tr}([P, D_{\hat{e}_1}(x)]\rho(x)). && \text{(C.194)}\end{aligned}$$

In order to compute the mixed derivative of second order, we now apply the product rule to this expression, which yields

$$\begin{aligned}&\frac{\partial^2}{\partial x_1 \partial x_2}\alpha_P \mathrm{Tr}(P\rho(x)) && \text{(C.195)}\\ =&\frac{\partial}{\partial x_2} i\alpha_P \mathrm{Tr}([P, D_{\hat{e}_1}(x)]\rho(x)) && \text{(C.196)}\\ =&i\alpha_P \left(\mathrm{Tr}(\left[P, \frac{\partial}{\partial x_2}D_{\hat{e}_1}(x)\right]\rho(x)) - \mathrm{Tr}([P, D_{\hat{e}_1}(x)]\frac{\partial}{\partial x_2}\rho(x))\right) && \text{(C.197)}\\ =&\alpha_P \mathrm{Tr}(i\left[P, \frac{\partial}{\partial x_2}D_{\hat{e}_1}(x)\right]\rho(x)) - \mathrm{Tr}([[P, D_{\hat{e}_1}(x)], D_{\hat{e}_2}(x)]\rho(x)). && \text{(C.198)}\end{aligned}$$

Note that the terms of this expression can be treated similarly as the first partial derivative. For each additional partial derivative, we obtain terms consisting of the product with nested commutators with $\rho(x)$ under the trace. The nested commutators contain $D_{\hat{e}_j}(x)$ or partial derivatives of it, for which we will later derive an explicit form. Hence, we can apply the same scheme until we arrive at the k-th partial derivative. In order to formalize this statement, we need to introduce some additional notation.

Throughout the rest of this section, we use the notation $\frac{\partial^{|B|}}{\partial x_B}$ to denote the mixed derivative with respect to all parameters $x_i \in B$ for some set B.

Definition C.21. *Let $k \in \mathbb{N}$. Let $A \subseteq [k]$ be a set of size $|A| = m$. Define an ordering $l_1 < l_2 < \cdots < l_m$ over the elements $l_1, l_2, \dots, l_m \in A$ of A. Then, we define*

$$\bigodot_{l\in A} \partial^{B_l} l \triangleq i^m \mathrm{Tr}(\left[\left[\cdots\left[P, \frac{\partial^{|B_{l_1}|}}{\partial x_{B_{l_1}}}D_{\hat{e}_{l_1}}(x)\right]\cdots\right], \frac{\partial^{|B_{l_m}|}}{\partial x_{B_{l_m}}}D_{\hat{e}_{l_m}}(x)\right]\rho(x)), \tag{C.199}$$

where $B_j \subset [k]$. We refer to the nested commutators under the trace as summands. *The set A and collection $\{B_l\}_{l\in A} \triangleq \mathcal{B}_A$ satisfy the following conditions:*

1. *Each summand contains* $D_{\hat{e}_1}(x)$.
2. *The sets* $A, B_1, \ldots, B_m$ *satisfy* $A \cup \bigcup_{j=1}^{m} B_j = [k]$ *and* $A \cap B_j = \emptyset$, $B_i \cap B_j = \emptyset$.
3. *For each* (B_l, l) *pair, it holds that* $i > l$ *for all* $i \in B_l$.

This notation gives a compact way of expressing the terms of the mixed derivative and allows us to address each mixed derivative of the terms $D_{\hat{e}_j}$ individually. Each term $\bigodot_{l\in A} \partial^{B_l} l$ contains the product of $m+1$ matrix-valued functions (including $\rho(x)$, which depend on x. The set A denotes the partial derivatives on the factor $\rho(x)$, which have been differentiated using Equation (C.188) when applying the product rule. We will address the partial derivatives on $D_{\hat{e}_j}$ later in this section, when we derive an upper bound for $\bigodot_{l\in A} \partial^{B_l} l$.

The first condition underlines that the first partial derivative on $\rho(x)$ is necessarily computed via Equation (C.188) and thus contained in each term. The second condition reflects that each partial derivative operates on exactly one factor in each summand when applying the product rule. The third condition arises from the order, by which the partial derivatives are computed. For example, when we apply $\frac{\partial}{\partial x_j'}$ after $\frac{\partial}{\partial x_j}$, the $\frac{\partial}{\partial x_j} D_{\hat{e}_j'}$ can not occur in any term, since no term contained $D_{\hat{e}_j'}$ when the partial derivatives $\frac{\partial}{\partial x_j}$ were computed.

We can show that the mixed partial derivatives of $\alpha P \mathrm{Tr}(P\rho(x))$ can be written in terms of $\bigodot_{l\in A} \partial^{B_l} l$.

Lemma C.22 (Mixed derivative)**.** *Let* $\mathcal{A}_k = \{A \subseteq [k] : 1 \in A\}$ *and* $\mathcal{B}_A$ *be as in Definition C.21. The mixed derivative of the ground state property* $\mathrm{Tr}(P\rho(x))$ *is given by*

$$\frac{\partial^k}{\partial x_1 \ldots \partial x_k} \alpha_P \mathrm{Tr}(P\rho(x)) = \alpha_P \sum_{A\in\mathcal{A}_k} \sum_{(B_1,\ldots,B_{|A|})\in\mathcal{B}_A} \bigodot_{l\in A} \partial^{B_l} l. \tag{C.200}$$

Proof. We proceed via induction. First, we verify that $\frac{\partial}{\partial x_k} \bigodot_{l\in A} \partial^{B_l} l$ with $A \in \mathcal{A}_{k-1}$ and $A \cup \bigcup_{j=1}^{m} B_j = [k-1]$ yield summands which fulfill the criteria for summands of the k-th partial derivative stated in Definition C.21. Then, we show that each summand of the k-th derivative stems from a unique summand from the $(k-1)$-th partial derivative.

For the first part, let $|A| = m$. Furthermore, let

$$I_s(j) = \begin{cases} \{j\} & \text{if } s = k \\ \emptyset & \text{else} \end{cases}. \tag{C.201}$$

Then,

$$\frac{\partial}{\partial x_k} \bigodot_{l\in A} \partial^{B_l} l = \sum_{j=1}^{m} \bigodot_{l_j\in A} \partial^{B_{l_j} \cup I_j(l)} l + \bigodot_{l\in A\cup\{k\}} \partial^{B_l} l, \tag{C.202}$$

where each summand fulfills the properties, since $k > l$ for all $l \in A$.

Next, let $A' \in \mathcal{A}_k$ and let $\mathcal{B}_{A'}$ be the corresponding collection. Then, it is easy to see that, if $k \in B_{l_j}$, it stems from the summand $\bigodot_{l \in A'} \partial^{B_l} l$ with $B_{l_1}, \dots, B_{l_j} \setminus \{k\}, \dots, B_{l_m}$. If $k \in A'$, it stems from $\bigodot_{l \in A \setminus \{k\}} \partial^{B_l} l$. □

With this expression in hand, we can move forward to bounding the mixed derivative, which we need in order to bound the Hardy-Krause variation. First, we will upper bound the number of terms in $\bigodot_{l \in A} \partial^{B_l} l$. Then, we derive an upper bound on the individual terms. For the first step, we exploit the conditions on the sets defining the mixed derivative. When we drop the third requirement in definition Definition C.21, $|\mathcal{B}_A|$ corresponds to the number of ways of assigning $k - m$ distinct balls to m bins. Thus, we obtain the following result on the number of terms $\bigodot_{l \in A} \partial^{B_l} l$ in the k-th mixed derivative.

Lemma C.23. *Let $\mathcal{A}$ denote the set of all subsets of $[k]$. Then, the number of summands in the expression $\bigodot_{l \in A} \partial^{B_l} l$ is upper bounded by*

$$|\mathcal{B}_{\mathcal{A}}| \leq \sum_{s=1}^{k} \binom{k}{s} s^{k-s}. \tag{C.203}$$

Proof. There are $|\mathcal{A}| = \sum_{s=1}^{k} \binom{k}{s}$ different subsets of $[k]$. For each set A with $|A| = s$, there are s sets B_l. Dropping the third condition in Definition C.21, we observe that each of the $k - s$ elements in $[k] \setminus A$ can be in any of the s sets. Thus, we obtain the claimed upper bound. □

In the next step, we aim to bound each individual term of the mixed derivative $\bigodot_{l \in A} \partial^{B_l} l$. Therefore, a crucial step is to bound the spectral norm of each factor. We first derive a preliminary result on the mixed derivatives of the factors in $D_{\hat{e}_j}(x)$, which depend on x. This can be done using Duhamel's Formula for the derivative of the exponential map on $e^{H(x)}$, where we exploit that we only compute the derivative with respect to one parameter at a time, such that we can treat $H(x)$ as a function, which only depends on one parameter.

Theorem C.24 (Derivative of the exponential map; Theorem 3a in [95])**.** *Let $A(t) : \mathbb{R} \to \mathbb{C}^{n \times n}$. Then,*

$$\frac{d}{dt} e^{A(t)} = \int_0^1 e^{(1-s)A(t)} \left(\frac{dA(t)}{dt} \right) e^{sA(t)} ds. \tag{C.204}$$

Lemma C.25. *Let $k \in [n]$, $B \subseteq [n] \setminus \{k\}$, such that $\left\| \frac{\partial^{|C|} h_j}{\partial x_C} \right\|_\infty \leq 1$ $\forall C \subseteq B \cup \{k\}$. Then*

$$\left\| \frac{\partial^{|B|}}{\partial x_B} \left(e^{itH(x)} \left(\frac{\partial h_j}{\partial x_k} \right) e^{-itH(x)} \right) \right\|_\infty \leq 2^{|B|+1} (|B|+1)^{|B|+1} \tag{C.205}$$

Proof. By Theorem C.24, the mixed derivative equals the sum of terms of the form

$$T = \int_0^1 \cdots \int_0^1 \prod_l f_l(s_l) ds_l \dots ds_1, \tag{C.206}$$

where $f_l(s_l)$ can be any of $e^{(1-s_l)iH(x)}$, $e^{s_l iH(x)}$, $\frac{\partial^l h_j}{\partial x_{B_l}}$ or 1. By our assumption and the Cauchy-Schwartz inequality, each term T satisfies $\|T\|_\infty \leq 1$. Furthermore, by the product rule, the number of terms is smaller than $\prod_{j=1}^{|B|}(2j+1)$. Since each term of the lth partial derivative (including k) is the product of at most $2l+1$ factors depending on x, such that the $(l+1)$th derivative contains at most $2l+1$-times as many factors. Thus, the number of terms is bounded above by

$$\prod_{j=1}^{|B|}(2j+1) \leq \prod_{j=1}^{|B|+1}(2j) = 2^n n! \leq 2^{|B|+1}(|B|+1)^{|B|+1}, \tag{C.207}$$

as required. □

Now we can bound the terms $\bigodot_{l\in A}\partial^{B_l} l$.

Lemma C.26 (Bound components of the derivative)**.** *Let $\bigodot_{l\in A}\partial^{B_l} l$ be as in Definition C.21. Then*

$$\left|\bigodot_{l\in A}\partial^{B_l} l\right| \leq (2C_\gamma)^{|A|}\prod_{s=1}^{|A|} 2^{|B_{l_s}|+1}(|B_{l_s}|+1)^{|B_{l_s}|+1}. \tag{C.208}$$

Proof. Recall that

$$D_{\hat{u}}(x) = \int_{-\infty}^{+\infty} W_\gamma(t) e^{itH(x)} \frac{\partial H}{\partial \hat{u}}(x) e^{-itH(x)} dt, \tag{C.209}$$

where $W_\gamma(t)$, such that

$$|W_\gamma(t)| \leq \begin{cases} \frac{1}{2} & 0 \leq \gamma|t| \leq \theta, \\ 35e^2(\gamma|t|)^4 e^{-\frac{2}{7}\frac{\gamma|t|}{\log^2(\gamma|t|)}} & \gamma|t| > \theta, \end{cases} \tag{C.210}$$

where θ is chosen to be the largest real solution of

$$35e^2(\gamma|t|)^4 \exp\left(-\frac{2}{7}\frac{\gamma|t|}{\log^2(\gamma|t|)}\right) = 1/2.$$

It is also useful to note that $\sup_t |W_\gamma(t)| = 1/2$. By definition of the terms $\bigodot_{l\in A}\partial^{B_l} l$, the Cauchy-Schwartz inequality, and $\|[A,B]\|_\infty \leq 2\|A\|_\infty\|B\|_\infty$, we obtain

$$\left|\bigodot_{l\in A}\partial^{B_l} l\right| \leq I_\gamma^{|A|} 2^{|A|}\prod_{s=1}^{|A|}\sup_t \left\|\frac{\partial^{|B_{l_s}|}}{\partial x_{B_{l_s}}}\left(e^{itH(x)}\left(\frac{\partial h_{j_s}}{\partial x_s}\right)e^{-itH(x)}\right)\right\|_\infty, \tag{C.211}$$

where

$$I_\gamma = \int_{-\infty}^{+\infty} |W_\gamma(t)| dt. \tag{C.212}$$

We bound each term individually. For the first term, we proceed in a similar manner as in [2] (Lemma 3). Namely, by Equation (S32) in [2], we can bound this integral by

$$\int_{t^*}^{+\infty} |W_\gamma(t)|dt \leq \frac{245}{2} e^2 \gamma^{-1} \left(\frac{1}{1 - \frac{35 \log^2(\gamma t^*)}{\gamma t^*}} \right) (\gamma t^*)^{10} e^{-\frac{2}{7} \frac{\gamma t^*}{\log^2(\gamma t^*)}} \triangleq C'_\gamma, \tag{C.213}$$

by choosing t^* such that $\gamma t^* = \max(5900, \alpha, 7(d+11), \theta)$ for some constant α. Here, we use C'_γ to denote a constant that depends only on γ. Moreover, since $|W_\gamma(t)| \leq \frac{1}{2}$, we can conclude that

$$\int_{-\infty}^{+\infty} |W_\gamma(t)|dt \leq \int_{-t^*}^{t^*} \frac{1}{2}\, dt + 2 \int_{t^*}^{+\infty} |W_\gamma(t)|\, dt \tag{C.214}$$

$$\leq \frac{\max(5900, \alpha, 7(d+11), \theta)}{\gamma} + 2C'_\gamma \triangleq C_\gamma, \tag{C.215}$$

where C_γ is also a constant that only depends on γ. By Lemma C.25, we obtain the desired statement. □

Lemma C.27 (Bounding the k-th mixed derivative)**.** *The k-th mixed derivative of $\alpha_P \mathrm{Tr}(P\rho(x))$ is bounded by*

$$\left| \frac{\partial^k}{\partial x_1 \dots \partial x_k} \alpha_P \mathrm{Tr}(P\rho(x)) \right| \leq |\alpha_P|\, 2^{\mathcal{O}(k \log(k))} \tag{C.216}$$

Proof. First, we derive an upper bound on the terms $\bigodot_{l \in A} \partial^{B_l} l$, which is independent of A and $\mathcal{B}_A$. Proceeding from the result of Lemma C.26, we obtain

$$(2C_\gamma)^{|A|} \prod_{s=1}^{|A|} 2^{|B_{l_s}|+1} (|B_{l_s}|+1)^{|B_{l_s}|+1} \leq (2C_\gamma)^{|A|} 2^k \prod_{s=1}^{|A|} k^{|B_{l_s}|+1} \leq (C_1 k)^k, \tag{C.217}$$

where we used $|A| \leq k$ and $\sum_l (|B_l|+1) = k$ and $C_1 = 4C_\gamma$. Furthermore, from Lemma C.23, it is easy to see that $|\mathcal{B}_\mathcal{A}| \leq k^k$. Thus, we obtain

$$\left| \frac{\partial^k}{\partial x_1 \dots \partial x_k} \alpha_P \mathrm{Tr}(P\rho(x)) \right| \leq |\mathcal{B}_\mathcal{A}| |\alpha_P| (C_1 k)^k \leq |\alpha_P| C_1^k k^{2k} = |\alpha_P| 2^{\mathcal{O}(k \log(k))}. \tag{C.218}$$

□

Note that when deriving this result, we do not require that the parameters for the mixed derivatives are distinct. Assuming that $\|H(x)\|_{W^{k,\infty}([-1,1]^m)} \leq 1$, we can induce an order recover the above bound for any mixed derivative of order k.

Corollary C.28. *If $\|H(x)\|_{W^{k,\infty}([-1,1]^m)} \leq 1$, then $\|\alpha_P \mathrm{Tr}(P\rho(x))\|_{W^{k,\infty}} \leq |\alpha_P| 2^{\mathcal{O}(k \log(k))}$.*

Proof. Note that the bound from Lemma C.27 is agnostic to the explicit directions $\hat{e}_j$ of the derivatives. Thus, we can choose any mixed derivative $\lambda \in \mathbb{N}_0^k$ such that $\sum_{j=1}^k \lambda_j = k$ and fix an order $o : \mathrm{dom}(\lambda) \to [k]$. Then, we can bound the mixed derivative on $o(\lambda)$ using the same approach as in Lemma C.27 to obtain the bound. □

D Details of numerical experiments

In this section, we discuss the numerical experiments in detail.

D.1 Experimental setup

As in [2], we consider the two-dimensional antiferromagnetic Heisenberg model with spin-1/2 particles placed on sites in a two-dimensional lattice. The corresponding Hamiltonian is

$$H = \sum_{\langle ij \rangle} J_{ij}(X_i X_j + Y_i Y_j + Z_i Z_j), \tag{D.1}$$

where $\langle ij \rangle$ denotes all pairs of neighboring sites on the lattice. The coupling terms J_{ij} correspond to the parameters x of the Hamiltonian and are sampled uniformly from $[0, 2]$ (and then mapped to lie in $[-1, 1]$ for our ML algorithm). The goal of the numerical experiment is to predict the two-body correlation functions, i.e., the expectation value of

$$C_{ij} = \frac{1}{3}(X_i X_j + Y_i Y_j + Z_i Z_j) \tag{D.2}$$

for all neighboring sites $\langle ij \rangle$.

To this end, we generate data similarly to [1], [2], approximating the ground state and corresponding correlation functions for the Hamiltonian Equation (D.1) of different lattice sizes and choices of coupling parameters J_{ij}. We consider lattice sizes of $4 \times 5 = 20$ up to $9 \times 5 = 45$. For each lattice size, we generate two datasets of size 4096, one with uniformly randomly distributed J_{ij} and one where the coupling parameters are distributed as a Sobol sequence. We obtained the data by approximating the ground state using the density-matrix renormalization group (DMRG) [96] based on matrix-product-states (MPS) [97], as has been done in [1], [2]. The simulations were performed on Nvidia T4 and A40 graphical processing units (GPUs). The former were used for lattice sizes from 4×5 up to 7×5 while the latter were used for lattice sizes 8×5 and 9×5. Depending on system size, we required between ≈ 50 and 200 hours on the respective hardware component to simulate one dataset of size 4096.

Our deep learning model was also trained on Nvidia T4 and A40 GPUs. We trained the models for all respective correlation terms in parallel, by training a full model $f_{ij}^{\Theta,w}$ (we omit the indices for the model's parameters) for each term

and minimizing the combined loss function

$$\sum_{\langle ij \rangle} \sum_{\ell=1}^{N} |f_{ij}^{\Theta,w}(x_\ell) - (C_{ij})_\ell|^2 \tag{D.3}$$

for the sake of time efficiency. For each data point, we trained a combined model for 500 epochs. For the terms of the local models $f_P^{\theta_P}$, as defined in Definition C.1, we used fully connected deep neural networks with five hidden layers of width 200. For training, we used the AdamW optimization algorithm [83]. Depending on the system size and the amount of training data, this took between 0.5 and 20 hours. As a baseline, we compared against the best model from [2]. The code can be found at `https://github.com/marcwannerchalmers/learning_ground_states.git`.

D.2 Additional experiments and discussion

In this section, we discuss the results of the numerical experiments and additional experiments performed that are not mentioned in the main text.

First, we perform additional experiments that analyze the scaling of the training/prediction error with respect to various parameters such as system size, local neighborhood size, and training set size (Figures 5.2 to 5.4). Importantly, in each of these, we see that the training error is small, as required by Theorem 3.3. Thus, as discussed in the main text, this assumption is satisfied in practice.

Moreover, as shown in Figure 4.2 (Left), the empirical prediction accuracy (RMSE) of the deep learning model is approximately constant with respect to the size of the lattice. Figure 5.2 (Right) further underlines this statement. The slight increase in prediction error for $\delta_1 > 0$ (size of the local neighborhood in Equation (A.2)) present in Figure 5.2 (Right) when increasing the system size from 4×5 to 5×5 may occur due to numerical errors in the data. From system size 5×5 onwards, we rather witness random fluctuations in test errors than a systematic increase.

Furthermore, we observe that the deep learning model significantly outperforms the regression model with random Fourier features from [2]. On the one hand, we notice that the performance of the latter could be improved, since the hyperparameters considered for hyperparameter tuning were selected for a substantially smaller dataset. This is underlined by the drop in RMSE for the regression model on Figure 5.3 for $\delta_1 = 1$, whereas a smaller RMSE is possible when choosing $\delta_1 = 0$. On the other hand, we think that the vast body of deep learning research also offers room for practical improvement of our deep learning model.

For $\delta_1 = 0$, we believe that our model achieves the best possible prediction error. For training set size larger than 2048, there is little improvement on the prediction error, as opposed to all experiments with $\delta_1 > 0$ (see Figure 4.2 (Center)). Furthermore, the training error remains relatively large compared to other choices of δ_1 (see Figure 5.2). Hence, we conclude that the error arising

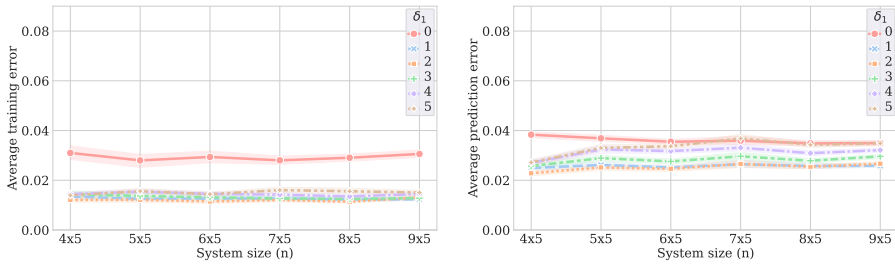

Figure 5.2: **Training/Prediction Error vs. System Size.** This figure shows the scaling of the training (left) and prediction (right) RMSE with respect to system size for different values of δ_1. All training sets are distributed as Sobol sequences and were trained on $N = 3686$ samples. The shaded areas denote the 1-sigma error bars across the assessed ground state properties.

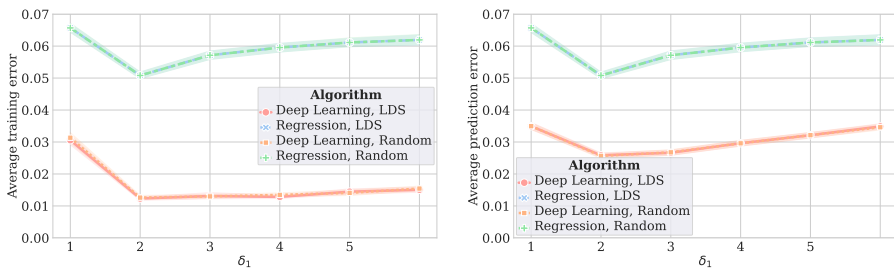

Figure 5.3: **Training/Prediction Error vs. Local Neighborhood Size.** This figure shows the scaling of the training (left) and prediction (right) RMSE with respect to the local neighborhood size δ_1. All training sets are of size $N = 3686$ with system size 9×5. The shaded areas denote the 1-sigma error bars across the assessed ground state properties.

from approximating the ground state property via local functions dominates the prediction error.

When increasing δ_1, we witness an increase in prediction error, especially for small training sets. This is consistent with Lemma C.10, which states that the bound on the prediction error is a combination of the training error and a term proportional to the star-discrepancy (and thus increases with the dimension of the domain of the local models). Our experimental results underline the balance which must be achieved between the two in order to obtain a small prediction error. This can clearly be observed in Figure 5.4. The training error decreases when increasing δ_1 and increases with the size of the training set. Meanwhile, the test error increases when increasing δ_1 and decreases with the size of the training set.

Another interesting observation is that the ML algorithm's performance on LDS seem to be almost the same as that of uniformly random points.

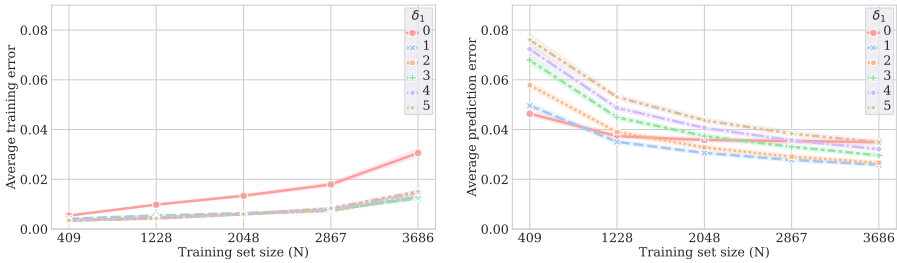

Figure 5.4: **Training/Prediction Error vs. Training Set Size.** This figure shows training (left) and prediction (right) RMSE with respect to training set size for different values of δ_1. All training sets are distributed as Sobol sequences and the grid size is 9×5. The shaded areas denote the 1-sigma error bars across the assessed ground state properties.

We believe this is due to the dominance of the local approximation error for small δ_1 and the drastic increase in dimensionality of the local models with increasing δ_1 outweighing the benefit of using LDS in practice. The dominance of approximation error is also a possible explanation for the slight decrease in prediction error with respect to the system size in Figure 4.2 (Left) and Figure 5.3. For our concrete choice of lattice shape and ground state properties, the local approximation error may be decreasing with respect to system size. However, we do not expect this to be the case in general.

D.3 Experiments with non-geometrically-local Hamiltonians

In this section, we assess the necessity of the geometric locality assumption by conducting numerical experiments for non-geometrically-local systems. We conclude that geometric locality is necessary for our theoretical results.

We conduct experiments on for a Hamiltonian given by

$$H = \sum_{j<i} J_{ij}(X_i X_j + Y_i Y_j + Z_i Z_j). \tag{D.4}$$

The difference between this Hamiltonian and Equation (D.1) is that the sites i and j are not required to be neighboring, thus violating the geomtric locality assumption needed for our rigorous guarantees. We predict the same ground state properties as in the previous section, i.e., two-body correlation functions on neighboring sites. Our ML model still uses the local coordinate set I_P from Equation (2.6). However, notice that the non-geometric-locality of the terms in the Hamiltonian impacts the number of parameters used. In other words, a larger number of parameters now affects a site in the neighborhood of each local Pauli. Furthermore, our adapted ML model assumes observables with

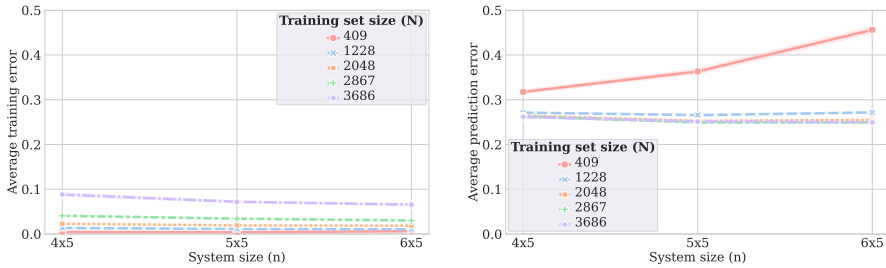

Figure 5.5: **Training/Prediction Error vs. System Size for Non-Geometrically-Local Systems.** This figure shows training (left) and prediction (right) RMSE with respect to the system size for the model given in Equation (D.4), which *violates* geometric locality. All training sets are of size $N = 3686$ and $\delta_1 = 0$. The shaded areas denote the 1-sigma error bars across the assessed ground state properties.

2-local terms[1]. Hence, the adapted ML model reads

$$\sum_{P \in S^{(2\text{-local})}} f_P^{\theta_P}. \tag{D.5}$$

Due to the lack of geometric locality and the larger number of terms of the Hamiltonian, the ground state properties are substantially harder to simulate, compared to the previous ones. We limit ourselves to uniformly random parameters and lattice shapes $4 \times 5, 5 \times 5$ and 6×5. The former two were simulated on Nvidia T4 GPUs and the latter on Nvidia A40 GPUs, using approximately $100 - 500$ hours per data set of size 4096. We also notice that the approximation error due to MPS may be larger in this dataset than in the previous one. As for the previous results, we trained the models for each ground state property in parallel, by optimizing the sum of their training objectives. For the local models $f_P^{\theta_P}$, we used fully connected neural networks with five hidden layers of width 100. This may not be optimal, but sufficient for the purpose of assessing the scaling of the prediciton error. We trained the models for different training set sizes using $\delta_1 = 0$. Since the adapted models consisted substantially more terms than the previous ones, training them for 500 epochs took between 5 and 35 hours on Nvidia T4 GPUs for lattice shapes 4×5 and 5×5 and on Nvidia A40 GPUs on a 6×5-lattice.

In Figure 5.5 (Right), we witness system size-dependent prediction error for the smallest training set size we investigate. Since the respective training error is very small, the respective prediction errors arise due to overfitting. This effect diminishes for larger training sets. This is what one would expect when directly applying the techniques of our theoretical results to this setting. Since the number of terms increases quadratically in system size, the norm of the weights in the final layer can not be bounded by a constant anymore. Furthermore,

[1] 2-local in the sense that P does not act on more than two not necessarily neighboring sites.

the properties of the local approximation do not hold true anymore. Hence, the predictive capabilities of a model with $\delta_1 = 0$ may be more limited here than in the geometrically local case. However, the prediction error may also be impacted by possible numerical errors in the training data, as well as the architecture of the local deep neural networks. Overall, these experiments illustrate the necessity of the geometric locality assumption in our theoretical results.

Paper II

Variational Quantum Optimization with Continuous Bandits

M. Wanner[1],J. Jonasson[1], Emil Carlsson, D. Dubhashi

Submitted, under review

The paper has been reformatted for uniformity.

https://arxiv.org/abs/2502.04021

[1] Equal contribution

Abstract

We introduce a novel approach to variational Quantum algorithms (VQA) via continuous bandits. VQA are a class of hybrid Quantum-classical algorithms where the parameters of Quantum circuits are optimized by classical algorithms. Previous work has used zero and first order gradient based methods, however such algorithms suffer from the barren plateau (BP) problem where gradients and loss differences are exponentially small. We introduce an approach using bandits methods which combine global exploration with local exploitation. We show how VQA can be formulated as a best arm identification problem in a continuous space of arms with Lipschitz smoothness. While regret minimization has been addressed in this setting, existing methods for pure exploration only cover discrete spaces. We give the first results for pure exploration in a continuous setting and derive a fixed-confidence, information-theoretic, instance specific lower bound. Under certain assumptions on the expected payoff, we derive a simple algorithm, which is near-optimal with respect to our lower bound. Finally, we apply our continuous bandit algorithm to two VQA schemes: a PQC and a QAOA quantum circuit, showing that we outperform or are competitive with state of the art methods based on finite difference schemes.

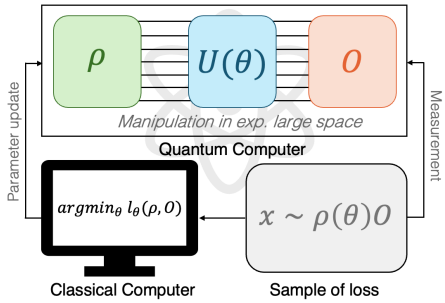

Figure 5.6: Illustration of VQA inspired by [9]

1 Introduction

In recent years, *variational quantum computing* has gathered momentum as a promising approach for quantum computers [1], [2], namely a *hybrid* classical-quantum framework which involves a quantum circuit with gates parameterized by continuous real variables (see Figure 5.6). Potential application areas range from Quantum chemistry [3], drug discovery [4] and material science [5] to Finance [6], supply chain management and manufacturing [7], where the Quantum circuit is used as an accelerator for specific domain-dependent problems. The circuit is specified by a set of real valued parameters which are tuned iteratively to optimal values by observing the circuit output using classical optimization algorithms. This approach has become popular in part to its flexibility, opening up applications in diverse areas in basic science and machine learning, and also because of the hope that it is more robust to the constraints of near-term quantum hardware in the NISQ (Noisy Intermediate Scale Quantum) era.

However, a big challenge for VQAs is that gradient based zero or first-order methods, such as COBYLA [8], become stuck because of the landscape of the optimization problem with increasing problem size: gradients or even loss differences become exponentially small which makes it difficult to identify local descent directions as the system size increases - exponentially many samples are provably necessary to identify descent directions. This is called the *barren plateau* (BP) phenomenon in Quantum computing [9] and affects various VQA, such as the Quantum Alternate Operator Ansatz [10], [11].

While significant effort has been dedicated to gradient based methods and trying to construct VQAs avoiding BPs (see Section 2), addressing the BP problem with other classical optimization techniques has received limited attention: Gradients are viewed as an indispensable component of VQAs, like they are for training classical neural networks. However, the evaluation of gradients or even the objective of a VQA may require many circuit evaluations, indicating that optimization based on individual VQA "shots", instead of the latter quantities, may be more natural.

Here we introduce a novel approach to address the BP problem which is theoretically well grounded and also easy to implement in practice. We argue that the optimization problem is very well suited to bandit methods which

combine global exploration and local exploitation. The optimal setting of the circuit parameters can be regarded as an optimization problem in a continuous bandit.

Black box optimization of noisy cost functions is a widely studied area with many applications [2], [12], [13]. One approach to this problem is via *bandit* algorithms, see [14] for a textbook treatment. In this work, we consider the best arm identification problem in bandits [15]. Formally, the typical problem setup consists of a set of distributions with means $\{\mu_a\}_{a=1}^{A}$, where each of them corresponds to an "arm" a. An agent is then tasked to identify the maximizing parameter, or "arm" a^*, with a given confidence by drawing a minimal amount of samples from the distributions.

For finite set of arms, this problem is well understood and can be solved optimally by e.g. *Track-and-Stop* [16] or game theoretic exploration schemes [17]. Further extensions provide optimal sampling strategies in settings where the arms reveal contextual information [18] or are subject to linear constraints [19]. An important property of these algorithms is their optimality with respect to the specific problem instance, which fundamentally differs from worst case optimality.

We consider the extension of the problem to the *continuous* setting, where the set of arms corresponds to the unit interval $[0, 1]$: thus there are uncountably many arms, each corresponding to a point in the unit interval: for every $x \in [0, 1]$, we have a distribution with mean $\mu(x)$. We assume the function $\mu : [0, 1] \to \mathbb{R}$, to be Lipschitz. Furthermore, the noise of the drawn samples is assumed to be sub-Gaussian. The task of the agent is to identify an arm x, which is ϵ close to the optimal arm x^* with high probability. The classical optimization problem in variational quantum algorithms corresponds exactly to this best arm identification problem in continuous bandit setting.

Continuous bandit optimization was addressed in previous work. Two very well known examples are the X Armed Bandits[20] and the Zooming algorithm [21]. These algorithms addressed the *regret* version of the bandit problem which is known to differ in important ways from the best arm identification problem. The former aims to accumulate as few sub-optimal samples as possible on an infinite time horizon, while the latter seeks to identify a probably approximatly correct (PAC) optimum under a minimal amount of samples. Here we address the best arm identification problem in continuous settings, which we consider a better representation for training VQA. Our bandit methods that combine global and local optimization are also applicable to better VQA designs and could be combined into hybrid schemes with purely local gradient based methods.

Specifically, we derive an instance specific lower bound for continuous, Lipschitz bandits on $[0, 1]$. Furthermore, we present a simple algorithm, whose sample complexity matches the lower bound up a log factor. Our experiments show an improvement in scaling over not instance specific methods and, due to its simplicity, substantially lower cost per iteration with respect to non-adaptive, instance optimal methods. Hence, our algorithm has a practical advantage over these methods, which turn out to be intractable when exceeding a certain number of arms. Our main contributions are:

- We introduce a novel approach to hybrid Quantum-classical algorithms

such as parametrized quantum algorithms (PQC) algorithms and QAOA by formulating it as a best arm identification problem in continuous bandits.

- We give an information-theoretic, instance specific lower bound for continuous best arm identification (in 1 dimension).
- We give an algorithm via adaptive partitioning that essentially meets the lower bound (up to a log factor) and linear computational complexity with respect to the number of samples.
- We give an algorithm that serves as a proxy for the multi-dimensional extension of the continuous bandit.
- We apply our bandit algorithm to PQC and QAOA problem instances, showing that it outperforms or is competitive with previous (finite difference) methods from literature. We also show synthetic examples where our method works while previous class of methods including SPSA fail catastrophically.

2 Related Work

Structured and Continuous Bandits Best arm identification for general bandit problems with a finite set of arms has been explored in both worst-case scenarios [22] and instance-specific settings [16], [17]. Subsequent research has extended this work to provide instance-specific bounds and algorithms for bandits incorporating contextual information, such as those with Linear or Lipschitz-continuous reward functions [18], as well as bandits subject to constraints [19]. Continuous bandits have primarily been studied in the context of regret minimization [21], [23], with further refinement in Adaptive-treed bandits [24], which, while focusing on cumulative regret, also offers PAC bounds. In the context of Quantum computing, regret minimization has been studied outside of VQA, where the task is to select an optimal observable from a finite or continuous set [25], [26], with regret defined as the expected measurement outcome, though the bounds in this setting are not instance-specific. Methods for pure exploration in continuous bandit settings, such as *MFDOO* [27], have been developed; however, they do not incorporate structural properties of the problem. Additionally, these methods rely on worst-case analysis and lack instance-dependent performance guarantees. There are no known bounds for continuous bandits with Lipschitz reward functions. Our work is the first to address this gap.

Mitigating BP in VQAs Recent research indicates a trade-off between the expressivity and trainability [28] of specific quantum circuit architectures, often referred to as *ansatz*. If an ansatz lacks sufficient expressivity, it may be incapable of representing the target function. Conversely, provably expressive ansatzes are typically susceptible to the BP phenomenon, which complicates the task of identifying the desired model within the represented model class. [9]

provides a summary of current techniques for mitigating BPs, a subset of which we briefly outline. One way to mitigate this issue is to find the best trade-off via adaptive structure search [29] and *ADAPT-VQE* [30]. Furthermore, most proofs of presence of BPs only apply to random parameter initialization. Therefore, employing alternative initialization strategies [31] can serve as an effective approach to mitigating this issue. Finally, certain architectures, such as noise-induced shallow circuits [32], are proven not to have BPs. However, recent work suggests that the absence of BPs implies classical simulability [33]. Even in the absence of BPs, the challenge of avoiding local minima remains. Alternative training methods for VQAs, such as those proposed in [34], [35], [36], [37], lack formal theoretical guarantees, weakening evidence of strong performance on larger problem instances. The same holds for applications of Bayesian optimization [38], which come with substantial computational complexity and potentially hard optimization of an acquisition function after each circuit evaluation. In this work, we introduce the application of frequentist bandit methods to this domain for the first time.

3 Preliminaries

This section introduces the general *stochastic bandit* model and its connection to VQAs.

3.1 Continuous Bandits

In its general form, a *stochastic bandit* is a pair $(\mathcal{X}, M)$ of a measurable space $\mathcal{X}$ and a set of random variables M. Each element $x \in \mathcal{X}$ is associated with a random variable $M(x) \in M$. Playing arm $x \in \mathcal{X}$ corresponds to observing an i.i.d. sample from $M(x)$. In this work, we consider *continuous bandits*, i.e., bandits with uncountably infinite set of arms $\mathcal{X} \subset \mathbb{R}^d$, whose expected rewards are denoted by $\mu(x) = \mathbb{E}[M(x)]$. Typically, $\mu(x)$ is assumed to be L-Lipschitz and the rewards $M(x)$ to be either sub-Gaussian or restricted to a bounded domain [23]. Stochastic bandits describe a setting where the only way of accessing μ is to observe samples of the respective rewards. Therefore, the continuous bandit model is directly applicable to VQA, which is outlined in the upcoming section.

3.2 Variational Quantum Optimization

As illustrated in Figure 5.6, in variational quantum computing, the process begins by initializing the quantum system to an n-qubit state ρ, which is then passed through a *parameterized quantum circuit* (PQC). Since all quantum circuits consist of unitary operations, a PQC can be expressed as a sequence of p parametrized unitaries $U(x) = \prod_{i=1}^{p} U_i(x_i)$, where $x = (x_1, ..., x_L)$ is a set of trainable parameters. After applying $U(x)$, the resulting state is measured with an observable O, yielding an outcome o_z, an eigenvalue of O associated with eigenvector $|z\rangle$. By the *Born rule*, the probability of measuring o_z is given by $\langle z| \rho(x) |z\rangle$, with $\rho(x) = U(x)\rho U(x)^\dagger$. The expected outcome is given by

$\mathrm{Tr}[\rho(x)O]$, which leads to the loss function $\ell_x(\rho, O) = \mathrm{Tr}[\rho(x)O]$. The expected loss is typically approximated by repeatedly running the circuit with the same parameters and computing the empirical average.
Reformulating this setup as measuring the initial state (possibly unknown) ρ with an observable from the set $\{\tilde{O}(x)\}_x$ with $\tilde{O}(x) = U^\dagger(x)OU(x)$ can be described as *multi-armed Quantum Bandits* [25]. Since the initial state ρ of a given PQC is known, optimizing a PQC reduces to a classical, continuous bandit problem. Explicitly, $\mu(x) = \mathrm{Tr}[\rho(x)O]$ and

$$P(M(x) = y) = \begin{cases} \langle z| \rho(x) |z\rangle & \text{if } y = o_z, \\ 0 & \text{otherwise.} \end{cases} \tag{3.1}$$

If the eigenvalues of O lie between 0 and 1, so does the reward distribution. This is easily achieved by linearly transforming O. Furthermore, general Unitaries $U(x_k)$ can be expressed as $e^{-ix_kV_k}$, where V_k is Hermitian with bounded spectral norm [28]. Therefore, μ's gradients are bounded by some constant L. Although our results only apply to 1-d continuous bandits, they can be used for multidimensional continuous bandits, for example in combination with Powell's method [39], which we will discuss in more detail in Section 5.

3.3 Problem statement

In Section 4 and Section 5, we consider a bandit problem $([0, 1], M(x))$, which satisfies the following assumptions:

(i) The rewards $M(x)$ are 1-sub-Gaussian.

(ii) The expected reward $\mu(x)$ is L-Lipschitz.

(iii) $\mu(x)$ has a unique optimum x^*.

(iv) There is a constant κ_0, such that μ is unimodal on every set $E \subseteq [x_i, x_i+\kappa]$ for all $\kappa < \kappa_0$.

The optimization algorithm aims to find an arm, which is close to the unique maximum
$x^* = \mathrm{argmax}_{x\in[0,1]}\mu(x)$ with high probability. Formally, we want the learner to recommend an arm $\hat{x}$, such that, given $\epsilon, \delta > 0$,

$$\Pr(\|\hat{x} - x^*\|_2 > \epsilon) \leq \delta. \tag{3.2}$$

An optimization algorithm that satisfies Equation (3.2) is often referred to as *δ-PAC learner*. Our goal is to design a PAC-learner, which finds an ϵ-optimal arm with probability at least $1 - \delta$ after a minimal number of steps, which we refer to by τ_δ^ϵ. In general, this stopping time is random so that we quantify the sample complexity of the algorithm by $\mathbb{E}[\tau_\delta^\epsilon]$. Note that assumptions (iii) and (iv) do not hold for general PQCs. In practice, this is not an issue, as these assumptions are not needed for our algorithm to find an arm with reward $\mu(x) \leq \mu(x^*) + \epsilon$.

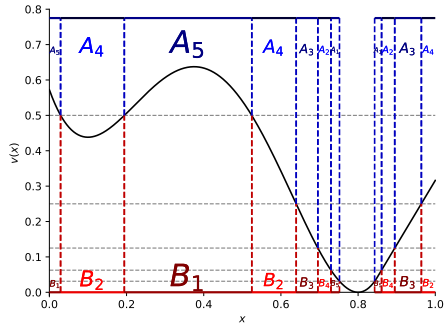

Figure 5.7: Example with $\epsilon = 2^{-5}, \kappa_0 = \frac{1}{4}, S = 3$.

Notation In the following, we introduce some notation, which will occur throughout Section 4 and Section 5. All other definitions will be introduced in the respective subsections. In the upcoming sections, we will consider the equivalent minimization problem for convenience. Therefore, let $v(x) \triangleq \mu(x^*) - \mu(x)$, such that v has its minimum where μ has its maximum and $v(x^*) = 0$. For clarity, we sometimes write $x^*(f) \triangleq \operatorname{argmax}_{x\in[0,1]} f(x)$ in order to refer to the maximizer of a specific function, which we omit for μ and v.

For convenience, we assume that $\epsilon = 2^{-D}$. Furthermore, we define the following level sets, i.e. sets of the form

$$v^{-1}[a, b] \triangleq \{x \in [0, 1] : a \leq v(x) \leq b\}.$$

Furthermore, we define $A_t = v^{-1}(2^{t-1}\epsilon, 2^t\epsilon]$ and
$B_t = v^{-1}(2^{-t}, 2^{-(t-1)}]$ for $t = 1, \ldots, D$. Note that $A_t = B_{D-t}$. This is illustrated in Figure 5.7. We use the Lebesgue measure, which we denote by $m(\cdot)$, to express the "length" of the sets.
Finally, we introduce two additional definitions.

Definition 3.1 (Covering-number)**.** *The* covering number $N_r(X)$ *is the smallest number of sets with diameter* r *required to cover a set* X*.*

The covering number gives rise to the *zooming dimension*, a concept, which is used in related literature, such as [21].

Definition 3.2 (Zooming-dimension)**.** *The* zooming dimension *of an instance* $\mu(x)$ *is defined as the smallest* β*, for which there is a constant* C*, such that* $N_{r/8}(X_r) = Cr^{-\beta}$ *for every* $1/2 > r > 0$ *and sets* $X_r \triangleq v^{-1}(r, 2r]$*.*

The zooming dimension is a convenient tool to write regret upper and lower bounds in concise form. Conceptually, it can be thought of as the limit of the "flatness" of μ around its maximum. In the 1-d setting, β takes values from 0 to 1.

4 Lower bound

In this section, we first establish an information-theoretic lower bound, following a similar approach to that of [16], [19]. Deriving a scheme analogous to *Track-and-Stop* directly from this bound is infeasible, as explicitly solving for w is intractable. Therefore, we derive a simplified, instance dependent lower bound. The proofs of all theoretical results in this section are provided in **??**.

4.1 Continuous lower bound

The methods to obtain a lower bound in the discrete, unstructured setting [16] can easily be adapted to the infinite-arm setting we consider. The main differences lie in the definition of the alternate set and the use of an integral instead of a sum, leading to a lower bound that closely resembles the one derived in the discrete setting. Denote the *alternate set* by

$$\mathrm{Alt}^\epsilon(\mu) \triangleq \{\lambda \ L\text{-Lipschitz} : \|x^*(\mu) - x^*(\lambda)\| > \epsilon\}. \tag{4.1}$$

For convenience, we only consider sample strategies, which have Riemann-integrable probability density and denote the respective set by $\mathcal{W}$. In the following, we present an information-theoretic lower bound for continuous bandits.

Theorem 4.1. *Let μ be a bandit continuous bandit on some set $\mathcal{X}$. For any (ϵ, δ)-PAC learner,*

$$\mathbb{E}[\tau_\delta^\epsilon] \geq \frac{\log(1/\delta)}{c^*(\mu)}, \tag{4.2}$$

where

$$c^*(\mu) = \sup_{w \in \mathcal{W}} \inf_{\lambda \in \mathrm{Alt}^\epsilon(\mu)} \int_{\mathcal{X}} w(x)(\mu(x) - \lambda(x))^2 dx. \tag{4.3}$$

The proof of Theorem 4.1 can be extended to alternative definitions of $\mathrm{Alt}^\epsilon(\mu)$ and holds for a general domain $\mathcal{X}$. It is a consequence of information theoretic properties of a PAC learner. Analogously to its discrete counterpart [17], Equation (4.3) has a game-theoretic interpretation: the w-player maximizes the value in Equation (4.3), to which the λ-player responds with λ that minimizes the objective for the given w. This perspective will allow us to derive a more tractable lower bound, as we will see in Section 4.2. Our choice of alternate set, i.e. restricting the location of the maximum of the confusing instance λ seems like the most natural extension of its discrete counterpart. However, in some cases, the objective may be to identify x, such that $\mu(x^*) - \mu(x) \leq \epsilon$. Consequently, one may consider alternate instances where $\max_x \mu(x) - \max_x \lambda(x) > \epsilon$. Though, this choice of alternate set does not properly reflect the optimization task and results in a strictly weaker, instance-independent lower bound.

Corollary 4.2. *Let $\mathcal{L}$ be the space of 1-Lipschitz functions on $\mathcal{X}$ and*

$$\mathrm{Alt}^\epsilon(\mu) \triangleq \{\lambda \in \mathcal{L} : \max_x \mu(x) - \max_x \lambda(x) > \epsilon\}. \tag{4.4}$$

Then,

$$c^*(\mu) = \epsilon^2. \tag{4.5}$$

This result supports our previous definition of the alternate set.

4.2 Simplified lower bound

In the following, we present an upper bound for $c^*(\mu) = c^*(v)$, which serves as a lower bound for $\mathbb{E}[\tau_\delta^\epsilon]$. We consider minimizing v for convenience and refer to $v'(x) = \lambda(x^*(\mu)) - \lambda(x)$ as the corresponding quantity for the alternate instance, denoted by λ in Equation (4.3). One can interpret $c^*(v)$ as a game, where two players compete against each other in alternating rounds: one player tries to minimize the expression by assigning values to v', while the other one tries to maximize it by applying values to w. In the beginning, the w-player may try to assign v', such that it equals v everywhere except in a small region around the minimum, ensuring that $v' \in \mathrm{Alt}^\epsilon(v)$. The w-player's approximately best response is to assign $w \equiv 0$ where $v' = v$ and uniform where $v' \neq v$. The v'-player may now be unable to play any $v' \in \mathrm{Alt}(v)$ which increases the objective, implying that the objective is close to c^*. However, v may be 'flat' in a certain area, where currently $w \equiv 0$. If this area is large enough, one can play a 1-Lipschitz $v' \in \mathrm{Alt}^\epsilon(v)$, which attains value $-\epsilon$ in that area and equals v in the remaining parts of the domain, yielding objective value 0. To prevent the v'-player from applying this strategy in other admissible parts of the domain, the w-player needs to respond with the uniform distribution over all such 'flat' areas (properly reweighed, such that none of them is more advantageous than others). This gives rise to our second result, the discrete lower bound.

Theorem 4.3. *Let $v : [0,1] \to \mathbb{R}$ satisfy the assumptions from Section 3. Then,*

$$\mathbb{E}[\tau_\delta^\epsilon] \geq \frac{\log(1/\delta)}{80\epsilon^3/L} \sum_{t=1}^{D} \frac{m(B_t)}{8^{D-t}}. \tag{4.6}$$

Theorem 4.3 is still instance dependent and holds for any $\kappa_0 > \epsilon > 0$. When $\epsilon \to 0$, one can show that the instance dependence reduces to the *zooming dimension* of μ.

Corollary 4.4. *Let $v : [0,1] \to \mathbb{R}$ with zooming dimension β satisfy the assumptions from Section 3. Then,*

$$\mathbb{E}[\tau_\delta^\epsilon] \geq \mathbf{\Theta}(\log(1/\delta)\epsilon^{-2+\beta}), \tag{4.7}$$

when $\epsilon \to 0$.

Theorem 2 from [24] states that for any strategy for tree-armed bandits, the regret $r_T \gtrsim T^{-1/(\beta+2)}$ at large enough round T. When setting $\epsilon \geq r_T$ and solving for T, one can observe that this coincides with Corollary 4.4. Note however that our result holds for any strategy.

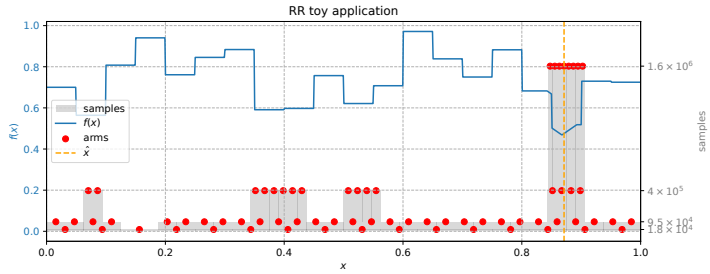

Figure 5.8: Toy function **(blue)** and example run of Algorithm 1 with D rounds. The red dots indicate the *arms* sampled from and the gray area the number of samples for each arm. The minimum estimated by the algorithm $\hat{x}$ is depicted by the orange dashed line.

5 Algorithm

In this section, we introduce a simple algorithm which matches the lower bound of the previous section up to a logarithmic factor. We briefly outline the algorithm and provide a convergence analysis. Finally, we outline a simple extension to multi-dimensional domains. The proofs of the theoretical results can be found in Section B.

5.1 Algorithm outline

The algorithm follows the following procedure: In each round $1 \leq t \leq D$, it draws sample of fixed size from points x on a uniform grid on $[0, 1]$. Then, we construct confidence intervals around $\mathbb{E}[v(x)]$ which, in combination with the Lipschitz property, allow us to exclude x and their neighbourhood, when the estimated mean of $v(x)$ is below some threshold. The number of grid points is doubled after each step, and we only consider points, which do not belong to the neighbourhood of excluded points. We stop when all points, except for an interval of length at most ϵ, have been excluded. This is illustrated in Figure 5.8.

In the following, denote the set of grid points in round t by

$$H_t = \left\{ \frac{1}{\lceil L \rceil} \left(\frac{k}{2^{t+3}} - \frac{1}{2^{t+4}} \right) \middle| 1 \leq k \leq \lceil L \rceil \cdot 2^{t+3} \right\}. \tag{5.1}$$

Furthermore, let $E_t \subseteq [0, 1]$ be the parts of the domain exclude in the end of round t and G_t the parts of the domain, which are left in the beginning of round t, such that $G_t = G_{t-1} \setminus E_t$. These quantities give rise to Algorithm 1.

The runtime of Algorithm 1 adheres to the following worst case-guarantee.

Theorem 5.1. *Let $v : [0, 1] \to \mathbb{R}$ satisfy the assumptions from Section 3. Then Algorithm 1 terminates after $D \triangleq \log_2(1/\epsilon)$ rounds and the number of*

Algorithm 1: Reject and Refine (RR)

Input: Inverted bandit $v(\cdot)$, constants L, ϵ.
Initialize $G_0 = [0,1]$, $t = 1$.
repeat
  for all $h \in G_{t-1} \cap H_t$ **do**
    Draw n_t samples from $v(h)$.
    Compute $\hat{v}(h)$
    Construct $1 - \frac{\delta}{|H_t| 2^t}$ CI of length $\frac{1}{2^{t+3}}$.
    $a_t^* \leftarrow \operatorname{argmin}_{h \in H_1} \hat{v}(h)$
    $E_t \leftarrow \bigcup_{h : \hat{v}(h) - \hat{v}(a_t^*) > \frac{12}{2^{t+4}}} \left[h - \frac{1}{2^{t+4}}, h + \frac{1}{2^{t+4}} \right]$
    $G_t \leftarrow G_{t-1} \setminus E_t$
    $t \leftarrow t + 1$
  end for
until $2^{-t} \leq \epsilon$
Output: $a^* = \operatorname{argmin}_{a_t^*} \hat{v}(a_t^*)$.

samples is bounded by

$$\tau_\delta^\epsilon \leq 2^{15} L \frac{\log_2(1/\epsilon) + \log(1/\delta)}{\epsilon^3} \sum_{t=1}^{D} \frac{m(B_t)}{8^{D-t}}. \tag{5.2}$$

The bound given in Theorem 5.1 matches the lower bound from Theorem 4.3 up to constant and $\log\left(\epsilon^{-1}\right)$ factors, indicating that Algorithm 1 is nearly optimal. Note that it holds for any ϵ irrespective of v being unimodal, allowing the final set $G_D \cap H_D$ to contain multiple arms with $v(a) \leq \epsilon$. This makes Algorithm 1 fundamentally different from running discrete best arm identification algorithms, as Frank-Wolfe sampling [18], on a sufficiently fine partition of $[0,1]$. Such a partition may contain multiple best arms, which would hinder methods of this kind from terminating.

Note that each refinement step comes with a cost: in the best case, one can always exclude half of the remaining domain, so that the number of arms remains constant. In the worst case, the number of arms doubles in each round. Since the confidence intervals of subsequent rounds require four times as many samples, the sample complexity increases by at least a factor four and at most eight.

The following result clarifies this and shows that the asymptotic performance of Algorithm 1 is at least as good *Adaptive-treed bandits.*

Corollary 5.2. *Consider $v : [0,1] \to \mathbb{R}$, which satisfies the assumptions from Section 3 and has* zooming dimension *β. Then*

$$\tau_\delta^\epsilon = \mathcal{O}\left((\log 1/\delta + \log 1/\epsilon) \epsilon^{-(\beta+2)} \right) \tag{5.3}$$

and when $\epsilon \to 0$, $\mathbb{E}[\tau_\delta^\epsilon]$ matches the lower bound from Corollary 4.4.

A key insight from Corollary 4.4 and Corollary 5.2 is that, in the limit $\epsilon \to 0$, continuous best arm identification and regret minimization are nearly equivalent.

Computational complexity Computations of Algorithm 1 besides drawing samples involve estimating the means $\hat{v}(h)$ and their comparison. Since each sample is included in exactly one empirical mean, the respective cost is τ_δ^ϵ. After each refinement step, the minimal arm is identified, with which the other arms are then compared, amounting to two comparisons per arm present. Clearly, we always draw at least one sample per arm, yielding an overall sample complexity linear in the number of samples, i.e. $\mathcal{O}(\tau_\delta^\epsilon)$. Hence, our approach is significantly more computationally efficient than Bayesian optimization methods, such as those used in [38], which require solving an increasingly large linear system after each sample.

Application to multi-dimensional parameter optimization In order to overcome the limitation of Algorithm 1 to a 1-dimensional parameter space, we employ a variety of schemes to map it to higher dimensional parameter spaces. At the heart of each strategy lies the following idea: For some function $f : [0,1]^d \to \mathbb{R}$, pick a point p and direction u, which give rise to the 1-dimensional function

$$g(s) = f((p + us)_{[0,1]^d}), \tag{5.4}$$

reducing the problem to computing

$$s^* = \text{argmin}_{s \in [0,1]}\ g(s) \tag{5.5}$$

with Lipschitz constant $L\|u\|_2$. By subscript $[0,1]^d$, we indicate "modulo", i.e. a periodic parameter domain, which applies to the VQA-setting. Depending on the application, the line may be truncated to $[0,1]^d$ instead.
Powells method [39] is a well-established, gradient-free optimization method, which incorporates a scheme of this kind and it can be shown that Powell's method efficiently minimizes functions of quadratic form. Typically, Brent's parabolic interpolation method [40] is used to find s^*, however, any gradient free optimizer can be applied, which makes Powell's method a well-suited extension to Algorithm 1. Another simple extension that falls into this category is to always choose u_k at random. We discuss the respective practical implementations in Section C.1.

6 Experiments

We compare the empirical convergence of Algorithm 1 and finite difference based methods on a 1-d toy example and two VQAs (see Figure 5.9). Our toy experiment conducted on a function with flat regions, local minima and a linear "wedge" around the optimum (see Figure 5.8 and Section C.2) illustrates how our algorithm profits from global context. As predicted by Theorem 5.1, the minima $\hat{x}_t$ computed by Algorithm 1 become consistently closer to the true x^*

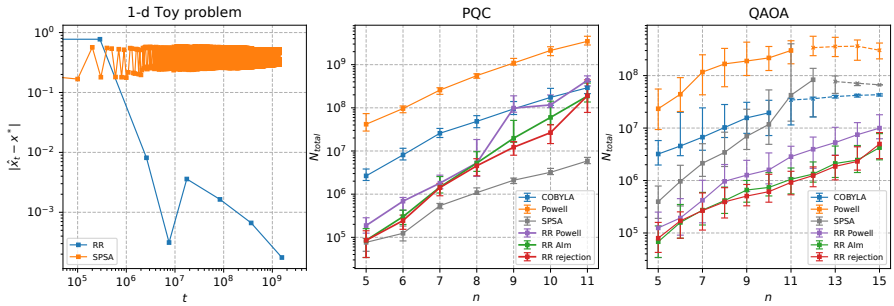

Figure 5.9: (1): Distance $|\hat{x}_t - x^*|$ to the true optimum at sample t for runs of Algorithm 1 and SPSA **(left)**. (2): Median and $(0.25, 0.75)$-quantiles of the sample complexity N_{total} for optimizing a PQC **(middle)** and MaxCut-QAOA **(right)** below the thresholds $C = 0.4$ and $C = 0.2$, respectively. N_{total} refers to the sample complexity and n to the number of qubits. The medians are computed over 20 and 100 simulations, respectively. Failure of convergence is indicated by a dashed line.

while SPSA [41] remains stuck in a local minimum when not initialized close to x^*. After a constant number of samples, Algorithm 1 samples exclusively from points near the wedge, indicating that the flat regions no longer influence convergence beyond this point.

The second part of our experiments examines the sample complexity of training VQAs to reach a target loss threshold, where we compare COBYLA [8], Powell's method [39] and SPSA with three implementations of the multidimensional proxy for Algorithm 1, outlined in Section C.1. The first example reproduces the experimental setup from [42], which aims to train a PQC with respect to an objective that exhibits barren plateaus. Our second example is a standard application of QAOA [43] to the MaxCut problem on random graphs, where the approximation ratio was optimized. For a detailed overview over the experimental setup we refer to Section C.3.

Although SPSA remains dominant on the PQC example, our algorithms achieve better sample complexity than Powell's method and COBYLA, and effectively outperform all finite difference based optimizers in the QAOA setting. Moreover, Powell's method experiences substantial improvement when equipped with Algorithm 1 as 1-d optimization routine. Our experiments show that our simple, unrefined method can compete with state-of-the-art baselines. While we do not consider our multidimensional proxy to constitute a new state-of-the-art approach, our experiments highlight the power of global context and the effectiveness of the bandit-framework we propose, especially in combination with other methods. This may include local methods. One might argue that finite difference strategies could eventually converge when probing enough initial points: bandit algorithms provide a principled strategy for selecting which points to probe.

7 Conclusion and Outlook

Here, we introduced a novel approach to VQA optimization based on continuous bandit techniques for exploration. Our work builds on and extends the existing bandit literature in several key ways. First, we generalize best-arm identification to continuous, Lipschitz bandits, formally motivating our choice of alternate set and deriving an instance-specific sample-complexity lower bound. We complement this bound with an algorithm that is provably near-optimal on the unit interval. Moreover, we show that, in the limit, our bounds coincide with those known for regret-minimization methods on an infinite time horizon, a result, which to our knowledge has not been known previously.
We also present simple extensions of our algorithm to higher dimensions, enabling us to evaluate the framework on VQA instances that exhibit barren plateaus. Notably, these straightforward schemes achieve competitive sample complexity on the considered instances. Further work is required to develop bandit algorithms that provide more general, non-asymptotic, and practically significant improvement in sample complexity. However, the framework we propose offers a scalable alternative that leverages global information and information-theoretic principles, while remaining inherently robust to noise. Unlike finite-difference–based methods, which become provably infeasible in the presence of barren plateaus.

Algorithm 1 suffers from a relatively large constant cost which could be reduced via more efficient construction of confidence intervals either by further exploiting the Lipschitz property, as [44] or more effective concentration laws, as e.g. [45]. On the theoretical side, one natural future direction is the extension of our algorithm to higher dimensions. This comes with the challenge that the algorithm not only needs to choose which parts of the domain it refines, but also along which direction - an extension of this kind for our algorithm requires additional ideas. Moreover, it may be possible to circumvent working with an approximate upper bound and formulate an algorithm in the style of *Track-And-Stop*, which proposes new parameters based on direct approximations of the continuous lower bound. Finally, our theoretical upper bounds reveal a new insight about the effect of BPs on trainability in 1-d, which we conjecture to also hold for the multi-dimensional setting: in the asymptotic sense, flat regions only affect the performance of our algorithms when the respective function value is close to the optimal value of the VQA. Therefore, even if BPs cannot be avoided, bandit methods have the potential to mitigate their effects.

Bibliography

[1] A. Abbas, A. Ambainis, Augustino and B. et al., "Challenges and opportunities in quantum optimization," *Nat Rev Phys*, vol. 6, pp. 718–735, 2024 (cit. on p. 2 (II)).

[2] M. Cerezo et al., "Variational quantum algorithms," *Nature Reviews Physics*, vol. 3, no. 9, pp. 625–644, 2021, ISSN: 2522-5820. DOI: 10.1038/s42254-021-00348-9. [Online]. Available: https://doi.org/10.1038/s42254-021-00348-9 (cit. on pp. 2 (II), 3 (II)).

[3] Y. Cao et al., "Quantum chemistry in the age of quantum computing," *Chemical Reviews*, vol. 119, no. 19, pp. 10 856–10 915, 2019, ISSN: 0009-2665. DOI: 10.1021/acs.chemrev.8b00803. [Online]. Available: https://doi.org/10.1021/acs.chemrev.8b00803 (cit. on p. 2 (II)).

[4] N. S. Blunt et al., "Perspective on the current state-of-the-art of quantum computing for drug discovery applications," *Journal of Chemical Theory and Computation*, vol. 18, no. 12, pp. 7001–7023, 2022, ISSN: 1549-9618. DOI: 10.1021/acs.jctc.2c00574. [Online]. Available: https://doi.org/10.1021/acs.jctc.2c00574 (cit. on p. 2 (II)).

[5] V. Lordi and J. M. Nichol, "Advances and opportunities in materials science for scalable quantum computing," *MRS Bulletin*, vol. 46, no. 7, pp. 589–595, 2021, ISSN: 1938-1425. DOI: 10.1557/s43577-021-00133-0. [Online]. Available: https://doi.org/10.1557/s43577-021-00133-0 (cit. on p. 2 (II)).

[6] D. Herman et al., "Quantum computing for finance," *Nature Reviews Physics*, vol. 5, no. 8, pp. 450–465, 2023, ISSN: 2522-5820. DOI: 10.1038/s42254-023-00603-1. [Online]. Available: https://doi.org/10.1038/s42254-023-00603-1 (cit. on p. 2 (II)).

[7] QUTAC et al., "Industry quantum computing applications," *EPJ Quantum Technol.*, vol. 8, no. 1, p. 25, 2021. DOI: 10.1140/epjqt/s40507-021-00114-x. [Online]. Available: https://doi.org/10.1140/epjqt/s40507-021-00114-x (cit. on p. 2 (II)).

[8] M. J. D. Powell, "A direct search optimization method that models the objective and constraint functions by linear interpolation," in *Advances in Optimization and Numerical Analysis*, S. Gomez and J.-P. Hennart, Eds. Dordrecht: Springer Netherlands, 1994, pp. 51–67, ISBN: 978-94-015-8330-5. DOI: 10.1007/978-94-015-8330-5_4. [Online]. Available:

https://doi.org/10.1007/978-94-015-8330-5_4 (cit. on pp. 2 (II), 13 (II)).

[9] M. Larocca et al., "A review of barren plateaus in variational quantum computing," *CoRR*, vol. abs/2405.00781, 2024 (cit. on pp. 2 (II), 2 (II), 4 (II)).

[10] S. Hadfield, Z. Wang, B. O,ÄôGorman, E. G. Rieffel, D. Venturelli and R. Biswas, "From the quantum approximate optimization algorithm to a quantum alternating operator ansatz," *Algorithms*, vol. 12, no. 2, 2019, ISSN: 1999-4893. DOI: 10.3390/a12020034. [Online]. Available: https://www.mdpi.com/1999-4893/12/2/34 (cit. on p. 2 (II)).

[11] E. Fontana et al., "Characterizing barren plateaus in quantum ans√§tze with the adjoint representation," *Nature Communications*, vol. 15, no. 1, p. 7171, 2024, ISSN: 2041-1723. DOI: 10.1038/s41467-024-49910-w. [Online]. Available: https://doi.org/10.1038/s41467-024-49910-w (cit. on p. 2 (II)).

[12] D. Bouneffouf and I. Rish, *A survey on practical applications of multi-armed and contextual bandits*, 2019. arXiv: 1904.10040 [cs.LG]. [Online]. Available: https://arxiv.org/abs/1904.10040 (cit. on p. 3 (II)).

[13] S. Maghsudi and E. Hossain, "Multi-armed bandits with application to 5g small cells," *IEEE Wireless Communications*, vol. 23, no. 3, pp. 64–73, 2016. DOI: 10.1109/MWC.2016.7498076 (cit. on p. 3 (II)).

[14] T. Lattimore and C. Szepesvári, *Bandit algorithms*. Cambridge University Press, 2020 (cit. on pp. 3 (II), 21 (II)).

[15] J.-Y. Audibert and S. Bubeck, "Best Arm Identification in Multi-Armed Bandits," in *COLT - 23th Conference on Learning Theory - 2010*, Haifa, Israel, Jun. 2010, 13 p. [Online]. Available: https://enpc.hal.science/hal-00654404 (cit. on p. 3 (II)).

[16] A. Garivier and E. Kaufmann, "Optimal best arm identification with fixed confidence," in *29th Annual Conference on Learning Theory*, V. Feldman, A. Rakhlin and O. Shamir, Eds., ser. Proceedings of Machine Learning Research, vol. 49, Columbia University, USA: PMLR, 2016, pp. 998–1027. [Online]. Available: https://proceedings.mlr.press/v49/garivier16a.html (cit. on pp. 3 (II), 4 (II), 8 (II), 8 (II)).

[17] R. Degenne, W. M. Koolen and P. Ménard, "Non-asymptotic pure exploration by solving games," in *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox and R. Garnett, Eds., vol. 32, Curran Associates, Inc., 2019. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2019/file/8d1de7457fa769ece8d93a13a59c8552-Paper.pdf (cit. on pp. 3 (II), 4 (II), 8 (II)).

[18] P.-A. Wang, R.-C. Tzeng and A. Proutiere, "Fast pure exploration via frank-wolfe," in *Advances in Neural Information Processing Systems*, M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang and J. W. Vaughan, Eds., vol. 34, Curran Associates, Inc., 2021, pp. 5810–5821. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2021/file/2dffbc474aa176b6dc957938c15d0c8b-Paper.pdf (cit. on pp. 3 (II), 4 (II), 11 (II)).

[19] E. Carlsson, D. Basu, F. Johansson and D. Dubhashi, "Pure exploration in bandits with linear constraints," in *Proceedings of The 27th International Conference on Artificial Intelligence and Statistics*, S. Dasgupta, S. Mandt and Y. Li, Eds., ser. Proceedings of Machine Learning Research, vol. 238, PMLR, 2024, pp. 334–342. [Online]. Available: https://proceedings.mlr.press/v238/carlsson24a.html (cit. on pp. 3 (II), 4 (II), 8 (II)).

[20] S. Bubeck, R. Munos, G. Stoltz and C. Szepesvári, "X-armed bandits," *J. Mach. Learn. Res.*, vol. 12, pp. 1655–1695, 2011. DOI: 10.5555/1953048.2021053. [Online]. Available: https://dl.acm.org/doi/10.5555/1953048.2021053 (cit. on p. 3 (II)).

[21] R. Kleinberg, A. Slivkins and E. Upfal, "Bandits and experts in metric spaces," *J. ACM*, vol. 66, no. 4, 30:1–30:77, 2019. DOI: 10.1145/3299873. [Online]. Available: https://doi.org/10.1145/3299873 (cit. on pp. 3 (II), 4 (II), 7 (II)).

[22] S. Bubeck, R. Munos and G. Stoltz, "Pure exploration in multi-armed bandits problems," in *Algorithmic Learning Theory*, R. Gavaldà, G. Lugosi, T. Zeugmann and S. Zilles, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 23–37, ISBN: 978-3-642-04414-4 (cit. on p. 4 (II)).

[23] S. Bubeck, R. Munos, G. Stoltz and C. Szepesvári, "X-armed bandits.," *Journal of Machine Learning Research*, vol. 12, no. 5, 2011 (cit. on pp. 4 (II), 5 (II)).

[24] A. D. Bull, "Adaptive-treed bandits," *Bernoulli*, vol. 21, no. 4, pp. 2289–2307, 2015. DOI: 10.3150/14-BEJ644. [Online]. Available: https://doi.org/10.3150/14-BEJ644 (cit. on pp. 4 (II), 9 (II)).

[25] J. Lumbreras, E. Haapasalo and M. Tomamichel, "Multi-armed quantum bandits: Exploration versus exploitation when learning properties of quantum states," *Quantum*, vol. 6, p. 749, Jun. 2022, ISSN: 2521-327X. DOI: 10.22331/q-2022-06-29-749. [Online]. Available: https://doi.org/10.22331/q-2022-06-29-749 (cit. on pp. 4 (II), 6 (II)).

[26] J. Lumbreras, M. Terekhov and M. Tomamichel, *Learning pure quantum states (almost) without regret*, 2024. arXiv: 2406.18370 [quant-ph]. [Online]. Available: https://arxiv.org/abs/2406.18370 (cit. on p. 4 (II)).

[27] E. de Montbrun and S. Gerchinovitz, "Certified multifidelity zeroth-order optimization," *SIAM/ASA Journal on Uncertainty Quantification*, vol. 12, no. 4, pp. 1135–1164, 2024. DOI: 10.1137/23M1591086. eprint: https://doi.org/10.1137/23M1591086. [Online]. Available: https://doi.org/10.1137/23M1591086 (cit. on p. 4 (II)).

[28] Z. Holmes, K. Sharma, M. Cerezo and P. J. Coles, "Connecting ansatz expressibility to gradient magnitudes and barren plateaus," *PRX Quantum*, vol. 3, no. 1, Jan. 2022, ISSN: 2691-3399. DOI: 10.1103/prxquantum.3.010313. [Online]. Available: http://dx.doi.org/10.1103/PRXQuantum.3.010313 (cit. on pp. 4 (II), 6 (II)).

[29] Y. Du, T. Huang, S. You, M.-H. Hsieh and D. Tao, "Quantum circuit architecture search for variational quantum algorithms," *npj Quantum Information*, vol. 8, no. 1, p. 62, 2022, ISSN: 2056-6387. DOI: 10.1038/s41534-022-00570-y. [Online]. Available: https://doi.org/10.1038/s41534-022-00570-y (cit. on p. 5 (II)).

[30] H. R. Grimsley, G. S. Barron, E. Barnes, S. E. Economou and N. J. Mayhall, "Adaptive, problem-tailored variational quantum eigensolver mitigates rough parameter landscapes and barren plateaus," *npj Quantum Information*, vol. 9, no. 1, p. 19, 2023, ISSN: 2056-6387. DOI: 10.1038/s41534-023-00681-0. [Online]. Available: https://doi.org/10.1038/s41534-023-00681-0 (cit. on p. 5 (II)).

[31] K. Zhang, L. Liu, M.-H. Hsieh and D. Tao, "Escaping from the barren plateau via gaussian initializations in deep variational quantum circuits," in *Advances in Neural Information Processing Systems*, S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho and A. Oh, Eds., vol. 35, Curran Associates, Inc., 2022, pp. 18 612–18 627. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2022/file/7611a3cb5d733e628081431445cb01fd-Paper-Conference.pdf (cit. on p. 5 (II)).

[32] A. A. Mele et al., *Noise-induced shallow circuits and absence of barren plateaus*, 2024. arXiv: 2403.13927 [quant-ph]. [Online]. Available: https://arxiv.org/abs/2403.13927 (cit. on p. 5 (II)).

[33] M. Cerezo et al., *Does provable absence of barren plateaus imply classical simulability? or, why we need to rethink variational quantum computing*, 2024. arXiv: 2312.09121 [quant-ph]. [Online]. Available: https://arxiv.org/abs/2312.09121 (cit. on p. 5 (II)).

[34] Z.-C. Yang, A. Rahmani, A. Shabani, H. Neven and C. Chamon, "Optimizing variational quantum algorithms using pontryagin's minimum principle," *Phys. Rev. X*, vol. 7, p. 021 027, 2 2017. DOI: 10.1103/PhysRevX.7.021027. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevX.7.021027 (cit. on p. 5 (II)).

[35] X. Bonet-Monroig et al., "Performance comparison of optimization methods on variational quantum algorithms," *Phys. Rev. A*, vol. 107, p. 032 407, 3 2023. DOI: 10.1103/PhysRevA.107.032407. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevA.107.032407 (cit. on p. 5 (II)).

[36] A. Agirre, E. van Nieuwenburg and M. M. Wauters, "A monte carlo tree search approach to qaoa: Finding a needle in the haystack," *New Journal of Physics*, vol. 27, no. 4, p. 043 014, Apr. 2025, ISSN: 1367-2630. DOI: 10.1088/1367-2630/adc765. [Online]. Available: http://dx.doi.org/10.1088/1367-2630/adc765 (cit. on p. 5 (II)).

[37] K. M. Nakanishi, K. Fujii and S. Todo, "Sequential minimal optimization for quantum-classical hybrid algorithms," *Phys. Rev. Res.*, vol. 2, p. 043 158, 4 2020. DOI: 10.1103/PhysRevResearch.2.043158. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevResearch.2.043158 (cit. on p. 5 (II)).

[38] K. A. Nicoli et al., *Physics-informed bayesian optimization of variational quantum circuits*, 2024. arXiv: 2406.06150 [cs.LG]. [Online]. Available: https://arxiv.org/abs/2406.06150 (cit. on pp. 5 (II), 12 (II)).

[39] M. J. D. Powell, "An efficient method for finding the minimum of a function of several variables without calculating derivatives," *The Computer Journal*, vol. 7, no. 2, pp. 155–162, Jan. 1964, ISSN: 0010-4620. DOI: 10.1093/comjnl/7.2.155. eprint: https://academic.oup.com/comjnl/article-pdf/7/2/155/959784/070155.pdf. [Online]. Available: https://doi.org/10.1093/comjnl/7.2.155 (cit. on pp. 6 (II), 12 (II), 13 (II)).

[40] R. Brent, *Algorithms for Minimization Without Derivatives* (Dover Books on Mathematics). Dover Publications, 2013, ISBN: 9780486143682. [Online]. Available: https://books.google.se/books?id=AITCAgAAQBAJ (cit. on p. 12 (II)).

[41] J. C. Spall, "An overview of the simultaneous perturbation method for efficient optimization," *Johns Hopkins apl technical digest*, vol. 19, no. 4, pp. 482–492, 1998 (cit. on p. 13 (II)).

[42] A. Arrasmith, M. Cerezo1, P. Czarnik1, L. Cincio and P. J. Coles, "Effect of barren plateaus on gradient-free optimization," *Quantum*, vol. 5, 2021 (cit. on pp. 13 (II), 31 (II), 31 (II), 32 (II)).

[43] E. Farhi, J. Goldstone and S. Gutmann, *A quantum approximate optimization algorithm*, 2014. arXiv: 1411.4028 [quant-ph]. [Online]. Available: https://arxiv.org/abs/1411.4028 (cit. on pp. 13 (II), 31 (II)).

[44] S. Magureanu, R. Combes and A. Proutiere, "Lipschitz bandits: Regret lower bound and optimal algorithms," in *Proceedings of The 27th Conference on Learning Theory*, M. F. Balcan, V. Feldman and C. Szepesvári, Eds., ser. Proceedings of Machine Learning Research, vol. 35, Barcelona, Spain: PMLR, 2014, pp. 975–999. [Online]. Available: https://proceedings.mlr.press/v35/magureanu14.html (cit. on p. 14 (II)).

[45] I. Waudby-Smith and A. Ramdas, "Estimating means of bounded random variables by betting," *Journal of the Royal Statistical Society Series B: Statistical Methodology*, vol. 86, no. 1, pp. 1–27, Feb. 2023, ISSN: 1369-7412. DOI: 10.1093/jrsssb/qkad009. eprint: https://academic.oup.com/jrsssb/article-pdf/86/1/1/56961777/qkad009.pdf. [Online].

Available: https://doi.org/10.1093/jrsssb/qkad009 (cit. on p. 14 (II)).

[46] E. Kaufmann, "Contributions to the Optimal Solution of Several Bandit Problems," Habilitation à diriger des recherches, Université de Lille, Nov. 2020. [Online]. Available: https://theses.hal.science/tel-03825097 (cit. on pp. 21 (II), 21 (II)).

[47] P. Erdős and A. Rényi, "On the strength of connectedness of a random graph," *Acta Mathematica Hungarica*, vol. 12, no. 1, pp. 261–267, 1961 (cit. on p. 31 (II)).

Appendix

A Proofs of Section 4

In this section, we prove the theoretical results from Section 4. We begin with the proof of the continuous lower bound given in Theorem 4.1.

Proof of Theorem 4.1. Consider a continuous bandit as defined in Section 3 with expected reward $\mu(x)$ on domain $\mathcal{X}$ and some policy, which proposes an *action*, i.e. a point $x_t \in \mathcal{X}$ with probability density $\pi(X_t = x_t | Z_t = z_t)$ in round t. By $Z_t = (R_{t-1}, X_{t-1}, \dots, R_1, X_1)$, we denote the history of previously observed rewards R_k and actions X_k. We assume that the reward distributions of the bandit and the policy have a probability density, which is true in most cases (see e.g. chapter 4.7 in [14]. For simplicity, we further only consider policies with Riemann-integrable densities. One can rewrite the following proof in terms of Lebesgue integrals, however, in the context of 1-sub-Gaussian, Lipschitz reward distributions, we do not expect any gain from the additional generality. Now consider a bandit with reward distribution $\Lambda(x)$ and expected reward $\lambda \in \mathrm{Alt}^\epsilon(\mu)$. We now proceed similarly as in the proof of Theorem 4.1 in [46]. Let $Z_t^\mu \sim P_\mu(Z_t), Z_t^\lambda \sim P_\lambda(Z_t)$ denote the History of samples under policy π and reward distributions M, Λ, respectively. On the one hand, consider $\mathcal{E}$ to be the event that the policy returns $x_{\tau_\delta^\epsilon}$, such that $|x_{\tau_\delta^\epsilon} - x^*| \leq \epsilon$. If π δ-PAC, then we get

$$\mathrm{kl}(H_{\tau_\delta^\epsilon}, H_{\tau_\delta^\epsilon}) \geq \mathrm{kl}(\mathbb{P}_\mu(\mathcal{E}), \mathbb{P}_\lambda(\mathcal{E})) \geq \mathrm{kl}(1-\delta, \delta) \tag{A.1}$$

by Lemma 0.1 in [46], where we use the same definition for $\mathbb{P}_\mu(\mathcal{E}), \mathbb{P}_\lambda(\mathcal{E})$, i.e. the expected exponents of the log-likelihood ratios. Note that the proof only makes use of the data processing inequality, which does not require a discrete action space. In the following, assume without loss of generality that $\Lambda(x)$ dominates $M(x)$ on all $x \in \mathcal{X}$. On the other hand note that

$$P_\mu(Z_t) = \prod_{i=1}^{t} P_\mu(R_i|X_i)\pi(X_i|Z_{i-1}) \tag{A.2}$$

and

$$P_\lambda(Z_t) = \prod_{i=1}^{t} P_\lambda(R_i|X_i)\pi(X_i|Z_{i-1}) \tag{A.3}$$

and therefore

$$\mathop{\mathbb{E}}_{P_\mu, \boldsymbol{X}}\left[\log \frac{P_\mu(Z_t)}{P_\lambda(Z_t)}\right] = \mathop{\mathbb{E}}_{P_\mu, \boldsymbol{X}}\left[\sum_{i=1}^{t} \log \frac{P_\mu(R_i|X_i)}{P_\lambda(R_i|X_i)}\right] \tag{A.4}$$

$$= \sum_{i=1}^{t} \mathop{\mathbb{E}}_{P_\mu, X_i}\left[\log \frac{P_\mu(R_i|X_i)}{P_\lambda(R_i|X_i)}\right] \tag{A.5}$$

$$= \sum_{i=1}^{t} \int_{\mathcal{X}\times\mathbb{R}} w_i(x,r) \log \frac{P_\mu(R_i = r|X_i = x)}{P_\lambda(R_i = r|X_i = x)} d(x,r), \tag{A.6}$$

where $w_i(x, r)$ is the joint probability density of action and reward for π in round i. Letting $w_i(x)$ the corresponding marginal density of the action, we obtain

$$= \sum_{i=1}^{t} \int_{\mathcal{X}\times\mathbb{R}} w_i(x) \int_{\mathbb{R}} P_\mu(R_i = r|X_i = x) \log \frac{P_\mu(R_i = r|X_i = x)}{P_\lambda(R_i = r|X_i = x)} dr dx \quad \text{(A.7)}$$

$$= \sum_{i=1}^{t} \int_{\mathcal{X}\times\mathbb{R}} w_i(x) \mathrm{kl}(M(x), \Lambda(x)) dx \quad \text{(A.8)}$$

$$= \int_{\mathcal{X}\times\mathbb{R}} \left(\sum_{i=1}^{t} w_i(x) \right) \mathrm{kl}(M(x), \Lambda(x)) dx. \quad \text{(A.9)}$$

By Equation (A.1), we now get

$$\int_{\mathcal{X}\times\mathbb{R}} \left(\sum_{i=1}^{t} w_i(x) \right) \mathrm{kl}(M(x), \Lambda(x)) dx \geq \mathrm{kl}(1-\delta, \delta) \quad \text{(A.10)}$$

$$\Rightarrow t \int_{\mathcal{X}\times\mathbb{R}} \left(\sum_{i=1}^{t} \frac{w_i(x)}{t} \right) \mathrm{kl}(M(x), \Lambda(x)) dx \geq \mathrm{kl}(1-\delta, \delta). \quad \text{(A.11)}$$

Since the inequality holds for all Λ according to our assumption, we can restate this as

$$t \inf_{\Lambda:\lambda(x)\in \mathrm{Alt}^\epsilon(\mu)} \int_{\mathcal{X}\times\mathbb{R}} \left(\sum_{i=1}^{t} \frac{w_i(x)}{t} \right) \mathrm{kl}(M(x), \Lambda(x)) dx \geq \mathrm{kl}(1-\delta, \delta) \quad \text{(A.12)}$$

and since w_i are arbitrary and $w_i \in \mathcal{W}$,

$$t \sup_{w\in\mathcal{W}} \inf_{\Lambda:\lambda(x)\in \mathrm{Alt}^\epsilon(\mu)} \int_{\mathcal{X}\times\mathbb{R}} w(x) \mathrm{kl}(M(x), \Lambda(x)) dx \geq \mathrm{kl}(1-\delta, \delta) \quad \text{(A.13)}$$

and by M, Λ sub-Gaussian,

$$t \sup_{w\in\mathcal{W}} \inf_{\Lambda:\lambda(x)\in \mathrm{Alt}^\epsilon(\mu)} \int_{\mathcal{X}\times\mathbb{R}} w(x) \mathrm{kl}(M(x), \Lambda(x)) dx \quad \text{(A.14)}$$

$$\leq t \sup_{w\in\mathcal{W}} \inf_{\lambda(x)\in \mathrm{Alt}^\epsilon(\mu)} \int_{\mathcal{X}\times\mathbb{R}} w(x) \mathrm{kl}(\mu(x), \lambda(x)) dx. \quad \text{(A.15)}$$

The statement follows directly. □

Note that the above proof works independently of the definition of $\mathrm{Alt}^\epsilon(\mu)$, given that the respective integrals remain well-defined. Hence, one may be interested in the impact of the choice of $\mathrm{Alt}^\epsilon(\mu)$. Indeed, as stated in Corollary 4.2, defining

$$\mathrm{Alt}^\epsilon(\mu) \triangleq \{\lambda(x) \text{ 1-Lipschitz} : \max_x \mu(x) - \max_x \lambda(x) \geq \epsilon\} \quad \text{(A.16)}$$

makes the lower bound much weaker, which we show in the next proof.

Proof of Corollary 4.2. Consider

$$\mathrm{Alt}^{\epsilon}_{\eta}(\mu) \triangleq \{\lambda(x)\,1\text{-Lipschitz} : \max_x \mu(x) - \max_x \lambda(x) \geq \epsilon + \eta\} \tag{A.17}$$

for some $\eta > 0$ and note that $\mathrm{Alt}^{\epsilon}_{\eta}(\mu) \xrightarrow{\eta \to 0} \mathrm{Alt}^{\epsilon}(\mu)$. First, we show that $c^*(\mu) \leq \epsilon$.
By the function class considered in $\mathrm{Alt}^{\epsilon}(\mu)$, we can choose $\lambda(x)$, such that $(\lambda(x) - \mu(x))^2 \leq (\epsilon + \eta)^2$ at any point x. Therefore,

$$\begin{aligned} c^*(\mu) &= \sup_{w \in \mathcal{W}} \inf_{\lambda \in \mathrm{Alt}^{\epsilon}_{\eta}(\mu)} \int_{\mathcal{X}} w(x)(\mu(x) - \lambda(x))^2 dx && \text{(A.18)} \\ &\leq \sup_{w \in \mathcal{W}} \int_{\mathcal{X}} w(x)(\epsilon + \eta)^2 dx && \text{(A.19)} \\ &= (\epsilon + \eta)^2. && \text{(A.20)} \end{aligned}$$

For $\eta \to 0$, the results follows.
Next, we show that $c^*(\mu) \geq \epsilon$.
If we choose $w(x) = \delta_{x^*}(x)$, which is 1 if and only if $x = x^*$, we get

$$\begin{aligned} c^*(\mu) &= \sup_{w \in \mathcal{W}} \inf_{\lambda \in \mathrm{Alt}^{\epsilon}_{\eta}(\mu)} \int_{\mathcal{X}} w(x)(\mu(x) - \lambda(x))^2 dx && \text{(A.21)} \\ &\geq \inf_{\lambda \in \mathrm{Alt}^{\epsilon}_{\eta}(\mu)} (\mu(x^*) - \lambda(x^*))^2 dx && \text{(A.22)} \\ &\geq (\epsilon + \eta)^2, && \text{(A.23)} \end{aligned}$$

again by definition of $\mathrm{Alt}^{\epsilon}_{\eta}(\mu)$. Letting $\eta \to 0$ yields the desired result, which concludes the proof. □

Now we proceed to proving Theorem 4.3. Recall that for convenience, we assume that $\epsilon = 2^{-D}$. Also for convenience, assume that also κ_0 is an integer power of 2, such that $\epsilon = 2^{-S}\kappa_0$ for some $S \in \mathbb{N}$. First, we present the arguments from Section 4.2 in a rigorous way. With slight abuse of notation, let

$$\mathrm{Alt}^{\epsilon}(v) \triangleq \{v'(x)\,L\text{-Lipschitz} : \left| \left(\mathrm{argmin}_{x \in [0,1]} v'(x) \right) - x^* \right| > \epsilon\}. \tag{A.24}$$

In the following, let $A_0 \triangleq v^{-1}[0, \epsilon)$ and $A_{\leq s} \triangleq \bigcup_{t \leq s} A_t$. Consider some interval $[a, b]$ contained in some level set set A_t. If we require $v(a) = v'(a)$ and $v(b) = v'(b)$, v', the Lipschitz-constraint prevents v' from obtaining the value $-\epsilon$ when the interval is shorter than $\frac{2}{L}(2^t + 2)\epsilon$. Therefore, we only need to consider intervals $[a, b] \subseteq A_t$ with $b - a \geq \frac{2}{L}(2^t + 2)\epsilon$.
By the assumption in Section 3, when $2^t\epsilon \lesssim \kappa_0$, A_t must be the union of exactly two intervals $[a_1, b_1], [a_2, b_2]$. Let A^l_t be the largest interval among the two. Furthermore, let

$$R(v) = \left\{ A^l_t : 0 \leq t \leq S \wedge m(A^l_t) \geq \frac{2}{L}(2^t + 2)\epsilon \right\} \tag{A.25}$$

denote the set of indices with feasible level set. If $R(v) = \emptyset$, v' can only attain some trivial solution, which is given by

$$v_0 \triangleq \begin{cases} 3\epsilon + \frac{1}{L}|x - m| & x \in [\frac{m-\epsilon}{L}, m + \frac{\epsilon}{L}] \\ 4\epsilon & x \in A_{\leq 2} \setminus [\frac{m-\epsilon}{L}, m + \frac{\epsilon}{L}] \\ v(x) & \text{otherwise,} \end{cases} \tag{A.26}$$

where $m \in A_{\leq 2}$ is chosen, such that it has distance ϵ from its boundary, i.e. $\inf_{x \in \partial A_{\leq 2}} \|m - x\|_2 = \frac{\epsilon}{L}$. This function equals v on all higher level sets, and is constant on $A_{\leq 2}$ except for a small bump, which touches the boundary of $A_{\leq 2}$ and represents a reasonable "approximate" infimum with respect to v' when the latter can not have a minimum outside of $[x^* - \frac{\epsilon}{L}, x^* + \frac{\epsilon}{L}]$ with value smaller than 0.
In the following, we restrict $\mathrm{Alt}^\epsilon(v)$ to a set of functions, which equal v everywhere except on some $A_t^l \in R(v)$, where they have one wedge, i.e. of the form $-\epsilon\frac{1}{L}|x - m|$ for some m.

Lemma A.1.

$$\frac{2}{L} \sup_{w \in \mathcal{W}} \inf_{v' \in \mathrm{Alt}^\epsilon(v)} \int_0^1 w(x)(v'(x) - v(x))^2 dx \leq \frac{2\epsilon^3}{\sum_{A_t \in R(v)} \frac{m(A_t)}{(2^t+2)^3}}. \tag{A.27}$$

Proof. If $R(v) = \emptyset$, Equation (A.27) is trivially satisfied. Otherwise, we restrict v' to have a wedge on some $A_t^l \in R(v)$, which attains $v'(x^*(v')) = -\epsilon$ is equal to v everywhere else. We denote the restricted alternate set by $\overline{\mathrm{Alt}}^\epsilon(v)$. Observe that $(v(x) - v'(x))^2 \leq (2^t + 2)^2\epsilon^2$, which it does on a set $F_t \subseteq A_t$ of length $m(F_t) \leq \frac{2}{L}(2^t + 2)\epsilon$. Therefore,

$$\sup_{w \in \mathcal{W}} \inf_{v' \in \overline{\mathrm{Alt}}^\epsilon(v))} \int_0^1 w(x)(v'(x) - v(x))^2 dx \tag{A.28}$$

$$\leq \sup_{w \in \mathcal{W}} \min_t \inf_{\substack{F_t \subseteq A_t^l \\ m(F_t) = \frac{2}{L}(2^t+2)\epsilon}} \int_{A_t^l} w(x)(2^t + 2)^2\epsilon^2 \mathbb{1}_{F_t} dx \tag{A.29}$$

$$\leq \sup_{w \in \mathcal{W}} \min_t \inf_{\substack{F_t \subseteq A_t^l \\ m(F_t) = \frac{2}{L}(2^t+2)\epsilon}} (2^t + 2)^2\epsilon^2 \int_{F_t} w(x) dx. \tag{A.30}$$

For w to maximize Equation (A.30), w needs to be uniform over A_t^l, i.e. $\frac{w_t}{m(A_t^l)}$ for constants $w_t > 0$. Plugging this in yields

$$\sup_{w \in \mathcal{W}} \min_t \inf_{\substack{F_t \subseteq A_t^l \\ m(F_t) = \frac{2}{L}(2^t+2)\epsilon}} (2^t + 2)^2\epsilon^2 \int_{F_t} w(x) dx \tag{A.31}$$

$$\leq \sup_{\substack{w_t A_t \in R(v) \\ \sum_t w_t = 1}} \min_t \frac{2}{L}(2^t + 2)^3\epsilon^3 \frac{w_t}{m(A_t^l)}. \tag{A.32}$$

To minimize this expression, we need to choose w_t, such that $\frac{2}{L}(2^t+2)^3\epsilon^3\frac{w_t}{m(A_t^l)}$ is the same for all t. Hence, the optimal w for the upper bound on $c^*(v)$ reads

$$w_s^*(x) = \begin{cases} \frac{m(A_s^l)/(2^s+2)^3}{\sum_{A_t\in R(v)} m(A_t^l)/(2^t+2)^3} & x \in A_s^l \cap R(v) \\ 0 & \text{otherwise.} \end{cases} \tag{A.33}$$

Note that the factors $\frac{2}{L}$ cancel out. Plugging this into the above expression, we get

$$\sup_{w\in\mathcal{W}} \inf_{v'\in \mathrm{Alt}^\epsilon(v)} \int_0^1 w(x)(v'(x)-v(x))^2 dx \le \sup_{w\in\mathcal{W}} \int_0^1 w(x)(v'(x)-v(x))^2 dx \tag{A.34}$$

$$\le \frac{\epsilon^3}{\sum_{A_t\in R(v)} \frac{m(A_t^l)}{(2^t+2)^3}} \tag{A.35}$$

$$\le \frac{2\epsilon^3}{\sum_{A_t\in R(v)} \frac{m(A_t)}{(2^t+2)^3}}, \tag{A.36}$$

where the last inequality follows from $m(A_t) \le 2m(A_t^l)$ for $A_t \in R(v)$. □

Lemma A.2. *Let $v : [0,1] \to \mathbb{R}$ satisfy the assumptions from Section 3. Then,*

$$\sum_{t=1}^{S} \frac{m(A_t)}{(2^t+2)^3} \le \max\left(2\sum_{A_t\in R(v)} \frac{m(A_t^l)}{(2^t+2)^3}, \frac{\epsilon}{4}\right). \tag{A.37}$$

Proof. If the right hand side of Equation (A.27) is less than $16\epsilon^2$, the v^*-player takes $v^* = v'$ and otherwise she takes $v^* = v_0$.

In case $R(v)$ is empty, the v^*-player sets $v^* = v_0$.

Note now that

$$\sum_{A_t\notin R(v), t\le S} \frac{m(A_t)}{(2^t+2)^3} \le \frac{\epsilon}{(2^t+2)^2} < \frac{\epsilon}{8}. \tag{A.38}$$

and if the right hand side of Equation (A.27) is less than $16\epsilon^2$,

$$\sum_{A_t\in R(v)} \frac{m(A_t)}{(2^t+2)^3} > \frac{\epsilon}{8}. \tag{A.39}$$

Hence whenever $v^* \neq v_0$,

$$\sum_{t=1}^{S} \frac{m(A_t)}{(2^t+2)^3} < 2\sum_{k\in R(v)} \frac{m(A_t)}{(2^t+2)^3} \tag{A.40}$$

On the other hand if the right hand side is at least $16\epsilon^2$, then

$$\sum_{A_t\in R(v)} \frac{m(A_t)}{(2^t+2)^3} \le \frac{\epsilon}{8} \tag{A.41}$$

and so

$$\sum_{t=1}^{S} \frac{m(A_t)}{(2^t+2)^3} < \frac{\epsilon}{4}. \tag{A.42}$$

□

Corollary A.3. *Let $v : [0,1] \to \mathbb{R}$ satisfy the assumptions from Section 3. Then*

$$\sup_{w \in \mathcal{W}} \int_0^1 w(x)(v'(x) - v(x))^2 dx \leq \frac{4\epsilon^3}{\sum_{t=1}^{S} \frac{m(A_t)}{(2^t+2)^3}}. \tag{A.43}$$

Proof. By Lemma A.2, we have that either

$$\sum_{t=1}^{S} \frac{m(A_t)}{(2^t+2)^3} \leq 2 \sum_{A_t \in R(v)} \frac{m(A_t^l)}{(2^t+2)^3} \tag{A.44}$$

or

$$\sum_{t=1}^{S} \frac{m(A_t)}{(2^t+2)^3} \leq \frac{\epsilon}{4}. \tag{A.45}$$

In the former case, the statement follows directly from Lemma A.1. In the latter case, we get

$$16\epsilon^2 = \frac{4\epsilon^3}{\epsilon/4} \leq \frac{4\epsilon^3}{\sum_{t=1}^{S} \frac{m(A_t)}{(2^t+2)^3}}, \tag{A.46}$$

and again, the desired statement follows from Lemma A.1. □

Proof of Theorem 4.3. By Corollary A.3

$$\frac{2}{L} c_\epsilon^*(v) \leq \sup_{w \in \mathcal{W}} \int_0^1 w(x)(v'(x) - v(x))^2 dx \leq \frac{4\epsilon^3}{\sum_{t=1}^{S} \frac{m(A_t)}{(2^t+2)^3}}. \tag{A.47}$$

Since κ_0 only depends on v and not ϵ, we then have for ϵ sufficiently small that

$$\frac{2}{L} c_\epsilon^*(v) \leq \sup_{w \in \mathcal{W}} \int_0^1 w(x)(v^*(x) - v(x))^2 dx \leq \frac{5\epsilon^3}{\sum_{t=1}^{T} \frac{m(A_t)}{(2^t+2)^3}}. \tag{A.48}$$

This finally leads to

$$\mathbb{E}[\tau_\delta^\epsilon] \geq \frac{\log(1/\delta)}{10\epsilon^3/L} \sum_{t=1}^{D} m(A_t)/(2^t+2)^3 \geq \frac{\log(1/\delta)}{80\epsilon^3/L} \sum_{t=1}^{D} \frac{m(A_t)}{8^t}. \tag{A.49}$$

□

Proof of Corollary 4.4. Let C, β, such that we require $Cr^{-\beta}$ sets of diameter $r/8$ to cover the sets $X_r = \{x \in [0,1] : r \leq \mu(x^*) - \mu(x) \leq r\}$ for any $r \in [0, 1/2]$. Furthermore, let $\eta(\epsilon)$ be the largest $0 < \eta < \epsilon$, such that $N_\eta(X_\eta) \geq C' r^{-\beta}$. By C', we denote the smallest constant, such that $\eta(\epsilon)$ exists for all $\kappa_0 > \epsilon > 0$. Note that C', must exist and be independent of ϵ. Otherwise, we could construct constant C'' and $\beta' < \beta$, such that we can cover all X_r with at most $C'' r^{-\beta'}$ sets of diameter $r/8$, which contradicts the definition of the zooming dimension. Now let $D(\epsilon) = \lfloor \log_2 \epsilon \rfloor$. Then, by unimodality of v, $B_{t-1} \cup B_t$ consists of at most two intervals and therefore

$$2^t N_{2^t}(B_{t-1} \cup B_t) \leq 2 \cdot (B_{t-1} \cup B_t). \tag{A.50}$$

Furthermore, note that

$$X_r \subset B_{t-1} \cup B_t \quad \forall\ 2^{-t} \leq r \leq 2^{-(t-1)}. \tag{A.51}$$

Thus, for $\epsilon < \kappa_0$,

$$\begin{aligned}
\sum_{t=1}^{D(\epsilon)} 8^t m(B_t) &\geq \frac{1}{2 \cdot 8} \sum_{t=2}^{D(\epsilon)} 8^t m(B_{t-1} \cup B_t) && \text{(A.52)}\\
&\geq \frac{4^{D(\epsilon)}}{32} N_{2^{D(\epsilon)}}(B_{D(\epsilon)-1} \cup B_{D(\epsilon)}) && \text{(A.53)}\\
&\geq \frac{4^{D(\epsilon)}}{32} \sup_{2^{-D(\epsilon)} \leq r \leq 2^{-(D(\epsilon)-1)}} N_r(X_r). && \text{(A.54)}
\end{aligned}$$

Since $\eta(\epsilon) = \epsilon$ in the limit, we can conclude that

$$\begin{aligned}
\lim_{\epsilon \to 0} \sup_{2^{-D(\epsilon)} \leq r \leq 2^{-(D(\epsilon)-1)}} N_r(X_r) &= \lim_{\epsilon \to 0} \sup_{2^{-D(\epsilon)} \leq r \leq 2^{-(D(\epsilon)-1)}} N_r(X_r) && \text{(A.55)}\\
&\geq \lim_{\epsilon \to 0} C' 2^{\beta D(\eta(\epsilon))} && \text{(A.56)}\\
&= \lim_{\epsilon \to 0} \epsilon^{-\beta}. && \text{(A.57)}
\end{aligned}$$

When plugging this back into Equation (A.52), the statement follows directly. □

B Proofs of Section 5

For illustration purposes, we start by analyzing Algorithm 1 for $L = 1$. Let $G_0 \triangleq [0,1]$. The goal for this round is to exclude all points x, for which $v(x) \geq \frac{1}{2}$, i.e. obtain $E_1 \supseteq B_1$, with probability at least $1 - \frac{\delta}{2^1}$. Therefore, we pull each of the $2^{1+3} = 16$ arms in H_1 sufficiently many times to create symmetric confidence intervals for $v(h)$, $h \in H_1$, of length at most $1/2^{1+3} = 1/16$, i.e. of the form

$$v(h) = \hat{v}(h) \pm \frac{1}{2^{1+4}} = \hat{v}(h) \pm \frac{1}{32} =: [\underline{v}(h), \overline{v}(h)],$$

at confidence level $1 - \delta/2^{2 \cdot 1+3} = 1 - \delta/32$, making the multiple confidence level at least $1 - \delta/2$. When constructing these confidence intervals, we do

this independently for each $h \in H_1$, i.e. without using any structure of v to make inference from pulls of nearby arms. Let E_1 be the set of points $x \in G_0$ for which we can conclude after these pulls that $v(x) > 9/32$, given that all confidence intervals are correct and let $B_{\geq t}$ for $\cup_{s \geq t} B_s$.
Assuming that all the confidence intervals indeed cover the true value $v(h)$, the Lipschitz property tells us that the true function value $v(a^*)$ of the arm a^* with the smallest empirical mean, i.e. $a^* = \operatorname{argmin}_{h \in H_1} \hat{v}(h)$, is at most $1/32$. This implies that

$$\overline{v}(a_1) \leq \frac{1}{32} + \frac{1}{16} = \frac{3}{32}.$$

Suppose now that $x \in B_1$, i.e. that $v(x) > 1/2$. Then there is an $h \in H_1$ at distance at most $1/32$ from x, which by the Lipschitz property must satisfy $v(h) > 15/32$ and hence the lower end of the confidence interval for $v(h)$ satisfies

$$\underline{v}(h) > \frac{13}{32},$$

from which we can conclude that $v(x) > 12/32$ for all $x \in [h - 1/32, h + 1/32]$. Taken together, the above allows us to drop the simplifying assumption that $v(x^*) = 0$ and leads to the conclusion

$$v(x) - v(a^*) > \frac{9}{32} \qquad \forall x \in [h - 1/32, h + 1/32].$$

In particular, this is the case when $\hat{v}(h) - \hat{v}(a^*) > \frac{12}{32}$. Hence $x \in E_1$ and since x was arbitrary, $E_1 \supseteq B_{\geq 2}$ with probability at least $1 - \frac{\delta}{2}$. For $G_1 = G_0 \setminus E_1$, we immediately get that $G_1 \subseteq B_{\geq 2}$. Moreover, as $1/2^2 < 9/32$, we get $B_{\geq 3} \subseteq G_1 \subseteq B_{\geq 2}$.
Continuing this procedure for $t = 2, \ldots, D$ with confidence levels $1 - \frac{\delta}{2^t}$ yields Algorithm 1.

Proof of Theorem 5.1. The algorithm works as follows for rounds $t = 2, 3, \ldots, D$. Pull each of the arms of $H_t \cap G_{t-1}$ sufficiently many times to obtain symmetric confidence intervals

$$v(h) = \hat{v}(h) \pm \frac{1}{2^{t+4}} \tag{B.1}$$

at confidence level $1 - \delta/(2^t |H_t|$. Let E_t be the set of points in G_{t-1}, for which can conclude that $v(x) - v(a_t) > 9/2^{t+4}$, provided that all confidence intervals are correct, and set $G_t = G_{t-1} \setminus E_t$. This yields the sets $G_2, G_3, \ldots, G_D$. By the union bound, this results in a multiple confidence level of at least $1 - \delta/2^t$.

Recall that for constructing a symmetric confidence interval on confidence level 1-α of length 2ℓ for the mean of a Gaussian distribution with unit variance, it suffices that the sample size N satisfies

$$N \geq \frac{2 \log(2/\alpha)}{\ell^2}. \tag{B.2}$$

Hence, the number of pulls per arm is $2^{2t+9} \log\big(2^{2t+4}/\delta\big)$ and so the total number of pulls in this round becomes

$$2^{2t+9} |H_t| \mu(G_{t-1}) \log\big(2^t |H_t|/\delta\big) = L 2^{3t+12} \mu(G_{t-1}) \log\big(L 2^{2t+4}/\delta\big). \tag{B.3}$$

Let

$$a_t = \text{argmin}_{h \in H_t} v(h). \tag{B.4}$$

Then $v(a_t) \leq 1/2^{t+4}$ and thus

$$\overline{v}(a_t) \leq \frac{3}{2^{t+4}}. \tag{B.5}$$

Consider an $x \in G_{t-1}$ for which $v(x) > 1/2^t$, if such x exists. Then there exists and $h \in H_t$ such that $v(h) > 15/2^{t+4}$. If $h \in G_{t-1}$, so that the h-arm is actually pulled in this round, this gives

$$\underline{v}(h) > \frac{13}{2^{t+4}}, \tag{B.6}$$

from which we can conclude that $v(x) > 12/2^{t+4}$ and hence $v(x) - v(a_t) > 9/2^{t+4}$ and thus $x \in E_t$ so that $x \notin G_t$. If on the other hand $h \notin G_{t-1}$, we already know from the previous round that $v(h) - v(a_{t-1}) > 9/2^{t+3}$ and so $v(h) - v(a_t) > 9/2^{t+3}$ and we can conclude that $v(x) - v(a_t) > 9/2^{t+3} - 2^{t+4} > 9/2^{t+4}$ and thus again $x \in E_t$. Thus provided that all confidence intervals are correct, $B_{\geq t+2} \subseteq G_t \subseteq B_{\geq t+1}$.

Now run the above for $t = 1, \ldots, D$. After this, we have $B_{\geq D+2} \subseteq G_D \subseteq B_{\geq D+1}$, i.e. $v^{-1}[0, \epsilon/2] \subseteq G_D \cap v^{-1}[0, \epsilon]$. Hence G_D is non-empty and all elements $x \in G$ are ϵ-optimal arms, so any arm in $G_D \cap H_D$ is ϵ-optimal.

The complexity of the algorithm is upper bounded by

$$\sum_{t=0}^{D} L 2^{3t+12} \mu(G_{t-1}) \log\big(L 2^{2t+4}/\delta\big) \leq 2^{14} L \sum_{t=1}^{D} 8^t \mu(B_{\geq t})(t + \log(/\delta)). \tag{B.7}$$

Since $\sum_{j=1}^{t} j\, 8^j < 2t \cdot 8^t$ and $\sum_{j=1}^{t} 8^j < 2 \cdot 8^t$, the right hand side is bounded by

$$2^{15} L \sum_{t=1}^{D} (t + \log(1/\delta)) 8^t \mu(B_t) < 2^{15} L (D + \log(1/\delta)) \sum_{t=1}^{D} 8^t \mu(B_t). \tag{B.8}$$

The expression on the right equals

$$2^{15} L \frac{\log_2(1/\epsilon) + \log(1/\delta)}{\epsilon^3} \sum_{t=1}^{D} \mu(B_t)/8^{D-t}. \tag{B.9}$$

So, in summary, the above defines an algorithm $\mathcal{A}$ with

$$\tau_{\mathcal{A}} \leq 2^{15} L \frac{\log_2(1/\epsilon) + \log(1/\delta)}{\epsilon^3} \sum_{t=1}^{D} \frac{\mu(B_t)}{8^{D-t}} = 2^{15} L (D + \log(1/\delta)) \sum_{t=1}^{D} 8^t \mu(B_t). \tag{B.10}$$

□

Finally, we present the proof of Corollary 5.2.

Proof of Corollary 5.2. For convenience, we rewrite

$$\frac{1}{\epsilon^3}\sum_{t=1}^{D}\frac{m(B_t)}{8^{D-t}} = \sum_{t=1}^{D} 8^t m(B_t). \tag{B.11}$$

For the number $N_{2^{-t}/8}(B_t)$ of sets with radius $2^{-t}/8$ required to cover B_t, it holds that $\frac{m(B_t)}{2^{-t}/4} \leq N_{2^{-t}/8}(B_t) \leq C2^{-\beta t}$ and hence $m(B_t) \leq C2^{t(\beta-1)}$ for some constant C. Using the formula for finite geometric sums and $\epsilon = 2^{-D}$, we get

$$\sum_{t=1}^{D} 8^t m(B_t) \leq \sum_{t=1}^{D} C2^{(2+\beta)t} \leq C2^{2+\beta)D} = \mathcal{O}(\epsilon^{-2+\beta}). \tag{B.12}$$

The proof of the second part can be done analogously to the proof of Corollary 4.4. □

C Details on the numerical experiments

In this section, we give a detailed overview over the implementation and experimental setup used in Section 6.

C.1 Notes on practical implementation

The extensions of Algorithm 1 to higher dimensions we consider may require a large number of 1-d optimizations. To control the sample complexity, we restrict the maximal depth of Algorithm 1 to a a threshold $D_{\max}$. Since most lines $g(s)$ may not contain the optimum, we terminate the procedure at depth 1 when the values observed so far suggest that significant improvement at higher depth is unlikely. The resulting computation yields $g_k(\hat{s}_k^*) \leq g_k(s_k^*) + \epsilon$ in step k, where

$$\hat{s}_k = \operatorname*{argmin}_{x \in \boldsymbol{H}_k} g_k(x) \quad \text{and} \quad \boldsymbol{H}_k = \bigcup_{t=1}^{D_{\max}} H_t.$$

Observe that $g_k(0) = \hat{g}_{k-1}(\hat{s}_{k-1}^*)$ in steps $k > 1$, yielding $s_k = 0$ as additional observation.

When using Algorithm 1 in combination with Powell's method (RR Powell), we incorporate this by selecting the minimizer of $\boldsymbol{H}_k \cup \{0\}$ as $\hat{s}_k^*$: we only *accept* $\hat{s}_k$ when it improves on the current minimum and choose $\hat{s}_k^* = 0$ otherwise, such that $p_{k+1} = p_k$. In the approach, where sample u_k uniformly at random, we update p_{k+1} to $p_k + \hat{s}_k u_k$

(i) with some acceptance probability a_k, depending on $g_k(\hat{s}_k)$ and $g_k(0)$ (reject)

(ii) if $g_t(t_k^*) < g_{t-1}(t_k^*)$ (AIm),

and $p_{k+1} = p_k$ otherwise. We consider $a_k = e^{-q(\hat{s}_k^* - \hat{s}_{k-1}^*)}$ a reasonable acceptance probability. Finally, δ may also be chosen in a favourable way, since the practical performance may suffer from overly conservative multiple confidence estimates. In summary, this results in q, D_{max}, δ and L as hyperparameters.

C.2 Details on Toy problem

Our toy example, illustrated in Figure 5.8, is a piecewise constant version of the function

$$f(x) = 1 - ((\sin(13x)\sin(27x) + 1)/4)$$

of the form

$$\tilde{f}(x) = f(i/20) \qquad \text{if} \quad \frac{i}{20} - \frac{1}{40} \leq x < \frac{i}{20} + \frac{1}{40}, \quad x = 1, \ldots, 20. \tag{C.1}$$

and a wedge with slope 2 around the minimum of f, which is at $x^* \approx 0.8675$. Note that this wedge causes f to have zooming dimension 0, which enables the convergence depicted in Figure 5.9. In the case of SPSA we evaluated the empirical mean of 10^5 samples at each evaluation in order to obtain a comparable number of samples. For illustration purposes, it was initialized at $x = 0.5$ and remained stuck in that region.
This experiments was conducted on a laptop and took roughly 5 minutes.

C.3 Details on experimental setups for VQAs

Implementation of PQC In the example from [42], the goal is to train the parameters in an n-qubit circuit $V(\theta)$, such that $V(\theta)\,|0\rangle = |0\rangle$, with the all-zero state and optimizing following local cost function

$$C(\theta) = \mathrm{Tr}[O_L V(\theta)\,|0\rangle\,\langle 0|\,V^T(\theta)], \tag{C.2}$$

where

$$O_L = \mathbb{1} - \frac{1}{n}\sum_{i=1}^{n} |0_i\rangle\,\langle 0_i|\,. \tag{C.3}$$

We trained the same PQC as in [42] for $n = 5, \ldots, 11$ qubits with $p = n$ layers, ensuring that the objective function exhibits barren plateaus. For each n, we conducted 20 simulations with initial parameters sampled uniformly at random. The optimization was terminated when either a threshold of $C = 0.4$ was reached or the respective algorithm converged. For all versions of RR, we found $q = 400$, $D_{\max} = 1$, $\delta = 20$ and $L = 0.5$ to be optimal. We tested COBYLA and Powell's method with the default settings from the respective SciPy functions and used $N = 10^3, 10^4, 10^5$ circuit evaluations to approximate the objective in each iteration. The latter two only converged for $N = 10^5$ on all examples. SPSA was implemented using *pennylane* with default settings. We found $N = 10^3$ to work best.

Implementation of QAOA Our second example demonstrates the application of (vanilla) QAOA [43] to the MaxCut problem on graphs with $n = 5, \ldots, 15$ vertices. The graphs were at random according to the Erdős-Rényi model [47] with edge probability 0.5. As objective, we used $1 - R_a$, where R_a denotes the approximation ratio. For each n, we performed 100 simulations, each utilizing a randomly generated graph and initial parameters sampled uniformly at random.

For RR, we used the same hyperparameters as in the previous experiments. For the other methods, we tested $N = 10^3, 10^4, 10^5, 10^6$ and found that even for $N = 10^6$, less than half of them converged below the threshold of $C = 0.2$ for $n > 11$. Their success rate of optimization varied between 1% and 30% for $n = 15$ across all tested values of N. In Figure 5.9, we depict runs with $N = 10^5$ for COBYLA and Powell's method and $N = 10^4$ for SPSA, which were the smallest number of shots for which we observed reasonable convergence rates. We terminated the SPSA experiments when they roughly exceeded 10^8 samples. The $40\%, 25\%$ and 10% of the depicted runs for $n = 13, 14, 15$, which did reach the threshold $C = 0.2$ did so after at least 3×10^7 samples.

Further notes The simulations of both experiments were carried out separately for each seed on Intel Xeon Gold 6130 cpus with 16 CPU cores. Each simulation included all system sizes n and took between 4 and 16 hours. Finally, it is worth noting that the sample complexity of these two methods is significantly smaller than that reported by [42], which our algorithm outperforms by several orders of magnitude. A possible explanation for this discrepancy is that we employed a different number of shots to approximate the objective.