# Reinforcement Learning and Variational Quantum Algorithms

by

Jiahao Yao

A dissertation submitted in partial satisfaction of the

requirements for the degree of

Doctor of Philosophy

in

Applied Mathematics

in the

Graduate Division

of the

University of California, Berkeley

Committee in charge:

Professor Lin Lin, Chair
Professor Pieter Abbeel
Professor Per-Olof Person

Fall 2023

Reinforcement Learning and Variational Quantum Algorithms

Abstract

Reinforcement Learning and Variational Quantum Algorithms

by

Jiahao Yao

Doctor of Philosophy in Applied Mathematics

University of California, Berkeley

Professor Lin Lin, Chair

In recent years, the realms of deep learning and variational quantum algorithms have undergone significant advancements. These innovative algorithms have proven to be exceptionally efficient and robust in addressing complex problems within quantum chemistry, condensed matter physics, and quantum field theory simulations, surpassing the capabilities of traditional classical algorithms. A key factor driving this progress is the development of hybrid quantum algorithms, which blend quantum and classical computational techniques.

Prominent examples of these hybrid algorithms include the Quantum Approximate Optimization Algorithm (QAOA), the Variational Quantum Eigensolver (VQE), and various Variational Quantum Algorithms (VQAs). These methods enable the construction of parameterized quantum circuits (PQCs), which are central to the operation of these algorithms. By employing PQCs, these algorithms leverage the unique properties of quantum computing, such as superposition and entanglement, to explore solution spaces more comprehensively than classical methods.

Furthermore, the optimization process in these hybrid algorithms involves a sophisticated interplay between quantum and classical computing resources. The quantum computer is used to evaluate the performance of the quantum circuit for given parameters, and classical optimization techniques are then applied to refine these parameters iteratively. This synergistic approach enhances the efficiency and effectiveness of the optimization process, making it particularly suitable for problems that are intractable for classical computers alone.

We primarily concentrate on a specific issue: the preparation of ground states. A

notable challenge in this process is the noise originating from measurements or the device itself. It's crucial to consider this noise when preparing ground states. To address this, we need to develop algorithms that are robust to noise. Our approach involves the development of variational quantum algorithms, which allow for parameter updates during iterative processes. Effectively preparing the ground state is vital, as it has significant applications in subsequent downstream tasks.

In addressing the ground state preparation challenge, our objective is to generate the ground state, defined as the lowest eigenstate of the Hamiltonian $H$. Our exploration is two-pronged: firstly, we investigate various parametrization methods for the variational circuits, aiming to enhance the flexibility and efficiency of the quantum circuits. Secondly, we scrutinize different optimization strategies. This includes examining policy gradients and incorporating optimization challenges within the framework of reinforcement learning, thereby expanding the scope and capability of our optimization methodologies.

In evaluating the optimization process, we utilize two critical metrics: fidelity and ground state energy. Fidelity measures the overlap between the target quantum states and the evolved quantum states from the quantum circuit, serving as an indicator of the precision in achieving the desired quantum state. Ground state energy, conversely, relates to observables that can be measured in experimental settings, offering valuable insights into the physical characteristics of the quantum system under investigation.

The algorithms we discuss are specifically engineered to operate effectively in environments where quantum computer measurements are subject to noise. Demonstrating robustness against such measurement noise, these optimization algorithms efficiently identify optimal parameters for the variational quantum circuits. This efficiency and resilience are pivotal in advancing the field of quantum computing, particularly in the context of practical, noisy quantum systems.

Chapter 1 introduces the background knowledge and overview of deep learning techniques and optimization algorithms, quantum circuits, and variational quantum algorithms the basic problem setup and provides an overview of the results in this paper.

Chapter 2 introduces a policy gradient approach to the Quantum Approximate Optimization Algorithm (QAOA) using methods. Chapter 3 presents reinforcement learning techniques for the preparation of many-body ground states in quantum systems. It specifically leverages counter-diabatic driving, a method that guides the system adiabatically to avoid non-equilibrium excitations, thus ensuring more reliable

ground state preparation. Chapter 4 presents a noise-robust, deep autoregressive policy networks based end-to-end quantum control framework as to the challenge of noise in quantum systems. Chapter 5 presents another approach which integrates MCTS with quantum circuit optimization, aiming to enhance the efficiency and effectiveness of the circuit design and operation. Chapter 6 presents a random coordinate descent method as a straightforward yet effective technique for optimizing parameterized quantum circuits.

Please note that Part 2 is based on [Yao, J., Bukov, M., & Lin, L. Mathematical and Scientific Machine Learning (pp. 605-634). PMLR.] (joint work with Marin Bukov, Lin Lin), Part 3 is based on [Yao, J., Lin, L., & Bukov, M. (2021). Physical Review X, 11(3), 031070.] (joint work with Marin Bukov, Lin Lin), Part 3 is based on [Yao, J., Kottering, P., Gundlach, H., Lin, L., & Bukov, M. Mathematical and Scientific Machine Learning (pp. 1044-1081). PMLR.] (joint work with Paul Kottering, Hans Gundlach, Lin Lin, Marin Bukov), and Part 5 is based on [Yao, J., Li, H., Bukov, M., Lin, L., & Ying, L. Mathematical and Scientific Machine Learning (pp. 49-64). PMLR.] (joint work with Haoya Li, Marin Bukov, Lin Lin, Lexing Ying). Finally, Part 6 is based on a joint work in preparation with Zhiyan Ding, Taehee Ko, Lin Lin, Xiantao Li).

# Contents

**Contents** **i**

**1 Introduction** **1**
1.1 Deep Learning . . . 1
1.2 Unsupervised Learning . . . 3
1.3 Reinforcement Learning . . . 9
1.4 Deep Autoregressive model . . . 25
1.5 Black box optimization . . . 29
1.6 AI for Science: Bridging Deep Learning and Quantum Physics . . . 34
1.7 Quantum circuit . . . 34
1.8 Mathematical Framework for Ground State Determination . . . 37
1.9 Quantum Approximate Optimization Algorithm (QAOA) . . . 38
1.10 Variational Quantum Eigensolver (VQE) . . . 40
1.11 Variational Quantum Algorithms . . . 41
1.12 Counter-Diabatic Driving in Variational Quantum Algorithms . . . 43
1.13 Barren Plateaus in Quantum Algorithms . . . 44
1.14 Outlook . . . 45

**2 Policy Gradient based Quantum Approximate Optimization Algorithm** **47**
2.1 Introduction . . . 48
2.2 Preliminaries . . . 50
2.3 Policy gradient based QAOA (PG-QAOA) . . . 52
2.4 Quantum Qubit Models . . . 57
2.5 Numerical Experiments and Results . . . 58
2.6 Conclusion and Outlook . . . 66

**3 Counterdiabatic Driving inspired Reinforcement Learning based variational quantum algorithms** **68**

3.1 Introduction . . . 69
3.2 Generalized Continuous-Discrete Quantum Approximate Optimization Ansatz . . . 70
3.3 Variational State Preparation inspired by Counter-Diabatic Driving . 73
3.4 Many-Body Ground State Preparation . . . 74
3.5 Comparison with Counter-Diabatic Driving . . . 85
3.6 Transfer Learning and Generalization of the RL Algorithm to different System Sizes . . . 89
3.7 Discussion/Outlook . . . 89
3.8 High-level optimization: Policy Gradient using Deep Autoregressive Networks . . . 91
3.9 Low-level optimization: finding optimal protocol time steps $\alpha_j$ . . . 105
3.10 Scaling with the number of particles $N$, the protocol duration $T$, and the circuit depth $q$ . . . 106
3.11 Many-Body Control Landscape . . . 109
3.12 Variational Gauge Potentials . . . 111
3.13 CD-QAOA for Many-Body State Preparation . . . 120

**4 Noise-Robust Deep Autoregressive Policy Networks based variational algorithms 139**
4.1 Introduction . . . 140
4.2 Preliminaries . . . 141
4.3 Mixed Discrete-Continuous Policy Gradient using Deep Autoregressive Networks . . . 148
4.4 Application: Quantum Ising Model in the Presence of Noise . . . 154
4.5 Numerical Experiments and Results . . . 156
4.6 Conclusion and Outlook . . . 157
4.7 Pseudocode and Algorithm Hyperparameters . . . 159
4.8 A comparison of compactly supported distributions defining continuous actions . . . 161
4.9 Choosing the protocol duration $T$ . . . 163

**5 Monte Carlo Tree Search based variational quantum algorithms 165**
5.1 Introduction . . . 165
5.2 Generalized QAOA ansatz . . . 168
5.3 Reinforcement learning setup . . . 169
5.4 Monte Carlo tree search with improved policy gradient solver . . . 170
5.5 Numerical experiments . . . 175
5.6 Conclusion and discussions . . . 182

5.7 Related works . . . 183
5.8 Setup of physical models . . . 185
5.9 Noise models . . . 187
5.10 Details for the natural policy gradient with entropy regularization . . 188
5.11 Additional experiment results . . . 190
5.12 Additional numerical results on the energy landscape . . . 192
5.13 Physical models with the identity action . . . 192

**6 Random Coordinate Descent for optimizing parameterized quantum circuit 195**
6.1 Introduction . . . 195
6.2 Preliminaries and main results . . . 203
6.3 Proof of main results . . . 208
6.4 Numerical results . . . 213
6.5 Conclusion . . . 225
6.6 Stochastic stability of noisy GD . . . 226
6.7 Stochastic stability of noisy RCD . . . 228
6.8 The proofs of theorem 7 and theorem 8 . . . 229
6.9 Parameterized Circuit for the VQE . . . 233
6.10 Parameterized Circuit for the VQE in QUBO experiments . . . 234
6.11 Additional histograms of partial derivative estimates . . . 234
6.12 Cost function for the TSP . . . 234
6.13 Technique used in quantum factoring . . . 238

**Bibliography 239**

# Acknowledgments

I would like to express my deepest gratitude to my advisor Lin Lin for his unwavering support throughout my journey at UC Berkeley. A fortuitous oversight, where the pivotal Berkeley email found its way into my spam folder, was serendipitously corrected, thanks to Lin Lin's persuasive discussion over Skype, which convinced me to embark on my academic path at this esteemed institution. My five years at UC Berkeley have been nothing short of transformative, both academically and personally, shaping what I believe to be the best years of my life. Lin Lin's mentorship has been invaluable; without his guidance, my achievements during these graduate years would not have been possible.

I extend my heartfelt thanks to Marin Bukov, whom I met at a mathematics seminar. Our ensuing discussions and collaborative journey have blossomed into a deep and enduring friendship. I cherish our times spent in cafés and at the LeConte office, engaging in pair programming and brainstorming sessions. Marin's nonjudgmental support, despite my initial lack of extensive physics background, enabled us to successfully undertake numerous research projects together, marking one of the most beautiful coincidences in my life.

My appreciation also goes to Michael Lindsey, one of the pioneering students in Lin's group. Our extensive discussions on the latest machine learning research, translated seamlessly into the language of physics, have been incredibly enlightening. Michael's exemplary role as a student and researcher at UC Berkeley has been a guiding light, shaping my own journey as a graduate student.

I am deeply thankful to esteemed professors Pieter Abbeel, Sergey Levine, and Ion Stoica at UC Berkeley. Their cutting-edge work and exemplary roles as academic researchers have profoundly influenced my academic aspirations and provided a benchmark for what it means to be a distinguished researcher. Every interaction with them has been a learning experience, constantly reminding me of the vast knowledge yet to be acquired.

I am equally thankful to my research collaborators, including Lexing Ying and again, Marin Bukov, among others. Their insights and contributions have been instrumental in my academic growth.

I extend a special mention to my friends for their unwavering support and camaraderie throughout my academic endeavors. Their constant encouragement and steadfast presence have been invaluable to me, and I am deeply grateful for their significant contribution to my journey. In particular, I would like to express my heartfelt thanks

to Michael Luo. Our shared experiences, ranging from collaborating on research projects to presenting at the ICLR conference, have not only been professionally enriching but have also cemented a strong and enduring friendship. I am profoundly thankful for the role he has played in both my personal and professional growth.

I am also immensely grateful to the staff in the Mathematics Department, particularly Vicky Lee, Jon, and Jason, for their invaluable assistance and support.

Finally, I must express my profound gratitude to my family. Their unceasing encouragement and support have been the foundation of my journey, making it possible for me to pursue my studies in the United States. Their sacrifices and belief in me have been the cornerstone of my achievements.

# Chapter 1

# Introduction

The convergence of deep learning and quantum computing represents a significant milestone in the evolution of computational technologies. This synergy is opening up new horizons, particularly in areas where traditional computing methods encounter limitations. The recent advancements in both fields have set the stage for groundbreaking developments, especially in the domain of variational quantum algorithms.

Particularly noteworthy is the integration of deep learning methods into hybrid variational algorithms within the quantum computing sphere. This integration symbolizes a promising direction for leveraging the strengths of both fields. Deep learning's robust data processing capabilities, combined with quantum computing's unparalleled computational power, have the potential to solve complex problems more efficiently than ever before.

This work aims to explore the depths of this integration, focusing on the ways in which deep learning methodologies can enhance and refine the performance of variational quantum algorithms. The goal is to unravel the complexities and potentials of this synergy, providing insights into how these advanced technologies can be harnessed to push the boundaries of computational science and technology.

## 1.1 Deep Learning

Deep Learning represents one of the most significant advancements in the field of artificial intelligence in recent years. At its foundation, deep learning is grounded in approximation theory, striving to use data for constructing a 'universal functional approximator' [144] – a concept widely recognized for its ability to model complex

functions [116]. This approach employs deep neural networks to learn and approximate functions that were previously challenging, marking a paradigm shift in computational modeling.

This innovative methodology has achieved remarkable success in various domains, notably in computer vision [179], natural language processing [353], and robotics [192]. Deep learning has proven to be exceptionally beneficial in the realm of scientific machine learning, a field where traditional methods often rely heavily on grid-based systems or are constrained to linear regression models [33]. Previously, the introduction of nonlinearity was limited to the use of power terms. However, deep networks offer a significant leap forward, effectively addressing this limitation by replacing human-designed functions with a data-driven approach [188]. This shift not only enhances the capacity for learning and approximation but also provides a more robust and versatile tool for tackling complex real-world problems.

In the realm of deep learning, supervised learning is a foundational paradigm. The process involves learning a function $f$ from given data $x_i \in \mathbb{R}^n$ and corresponding labels $y_i \in \mathbb{R}$, with the goal of approximating $f(x_i) \approx y_i$. This task is mathematically similar to interpolation. In one-dimensional cases, traditional methods like Lagrange interpolation using polynomial bases are straightforward. However, the complexity significantly increases in higher dimensions. Here, neural networks emerge as an effective alternative to traditional basis functions. Unlike methods relying on Taylor expansion, where expressiveness is derived from additive properties, neural networks gain their expressiveness from the compositional nature of their layered structure [327].

The architecture of a neural network is defined by its trainable parameters, denoted as $f_\theta$. The central task in training these networks involves optimization to find the optimal parameters $\theta$, formally expressed as $\theta = \arg\min_{\theta \in \mathbb{R}^d} L(\theta) = \frac{1}{N} \sum_{i=1}^{N} L(f_\theta(x_i), y_i)$. Among various loss functions, the $L_2$ loss, defined as $L(\hat{y}, y) = (\hat{y} - y)^2$, and the logistic loss function, $L(\hat{y}, y) = -\left(y \log(\hat{y}) + (1 - y) \log(1 - \hat{y})\right)$, are frequently employed [116]. The logistic loss is particularly useful for binary classification tasks where $y_i \in \{0, 1\}$.

In scientific computing, deep learning frameworks have shown great promise in tackling inverse problems. Consider a complex forward process that can be executed to generate specific outputs from given inputs. These input-output pairs become invaluable training data for deep neural networks. By utilizing this data, the networks are trained to approximate the function $f_\theta$, thus becoming powerful tools for modeling the underlying process and facilitating the resolution of inverse problems [157].

An essential insight from the concept of Learning Fast Approximations of Sparse Coding [187] is its ability to integrate multiple iterations into a single forward function, thereby significantly reducing the iteration count. This methodology proves particularly advantageous in contexts such as the MCMC (Markov Chain Monte Carlo) chain, where it facilitates the learning of mappings that span multiple Markovian steps. Ultimately, this approach can drastically cut down the mixing time, offering a more efficient alternative to performing multiple MCMC runs.

Similar strategies can be applied across various domains of computational science and machine learning. For instance, in the field of optimization, similar techniques could accelerate convergence in iterative algorithms. In deep learning, this approach might lead to more efficient training of neural networks, especially in scenarios requiring complex or time-consuming iterative processes. Additionally, in the realm of signal processing and image reconstruction, such methodologies could significantly expedite algorithms, contributing to faster and more efficient processing.

This concept of reducing iteration counts and optimizing computational steps has broader implications, potentially transforming the way complex problems are approached and solved. By applying this principle, we can achieve more efficient, faster, and potentially more accurate solutions across a multitude of computational tasks.

## 1.2 Unsupervised Learning

One of the primary challenges in supervised learning is its heavy reliance on extensive volumes of labeled data. Acquiring such labels often proves to be labor-intensive and costly, posing a significant obstacle in many scenarios. In the real world, while there is frequently an abundance of data, corresponding labels may be scarce or entirely absent. This gap underscores the growing relevance of unsupervised learning, where learning algorithms infer patterns and structures directly from unlabeled data.

Within this context, self-supervised learning emerges as a notable subset of unsupervised learning. It involves the creation of a loss function directly from the data itself, circumventing the need for human-generated labels. This method typically entails formulating a prediction problem using only the available data. For example, a neural network might be trained by hiding parts of the input data, aiming to predict these masked portions in the output. As demonstrated in works like those by Vincent et al. [316] and Zhang et al. [355], such strategies strive to develop tasks solely from the data at hand. This approach facilitates the learning of insightful data representations without the dependency on explicit labeling.

The subsequent sections delve into a series of experiments highlighting advancements and applications in the field of self-supervised learning.

## Autoencoder

The autoencoder [21], a distinctive model in unsupervised learning, operates independently of labeled data. An autoencoder is constructed using two key functions: the encoder function $f$ and the decoder function $g$. The encoder $f : \mathbb{R}^d \mapsto \mathbb{R}^m$ transforms the data into a latent space, while the decoder $g : \mathbb{R}^m \mapsto \mathbb{R}^d$ aims to reconstruct the original data from this latent representation. Here, $m$ denotes the dimensionality of the latent space. Typically, these functions are realized through separate neural networks. The primary goal in training an autoencoder is to minimize the difference between the reconstructed data and the original input, thereby enabling the model to learn meaningful data representations without reliance on external labels. The optimization problem can be expressed as:

$$\min_{\theta,\varphi} \frac{1}{N} \sum_{i=1}^{N} \|g_\varphi(f_\theta(x_i)) - x_i\|^2$$

This approach allows the autoencoder to use the data itself as a target for learning, capturing essential features and patterns in an unsupervised manner.

## Variational Autoencoder

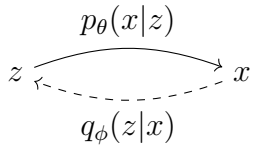



Figure 1.1: Schematic representation of a Variational Autoencoder (VAE). The solid arrow represents the generative process $p_\theta(x|z)$, mapping the latent variable $z$ to the observed data $x$. The dashed arrow denotes the inference process $q_\phi(z|x)$, estimating the latent variable from the observed data.

The Variational Autoencoder (VAE) is a fundamental model in unsupervised learning, renowned for treating the encoding and decoding processes probabilistically. It represents data $x$ and latent representations $z$ through probability distributions:

$p_\theta(x|z)$ for the likelihood of data given the latent representation, and $q_\phi(z|x)$ for the distribution of latent embedding given the data.

Central to VAEs is the concept of the Variational Lower Bound, or Evidence Lower Bound (ELBO). This is derived using Jensen's Inequality:

$$\log p_\theta(x) \geq \mathbb{E}_{z\sim q_\phi(z|x)}[\log p_\theta(x|z)] - D_{KL}[q_\phi(z|x)\|p(z)]$$

The goal in training VAEs is to maximize this ELBO, which involves optimizing the expected log likelihood of the data given the latent variables, minus the Kullback-Leibler (KL) divergence between the encoder's distribution and the prior distribution over latent variables.

Expanding on the KL divergence, we have:

$$\begin{aligned} D_{KL}(q_x(z)\|p(z|x)) &= \mathbb{E}_{q_x(z)}[\log q_x(z) - \log p(z|x)] \\ &= \mathbb{E}_{q_x(z)}[\log q_x(z) - \log p(z) - \log p(x|z) + \log p(x)] \end{aligned}$$

This leads to the lower bound of the log likelihood:

$$\begin{aligned} \log p(x) &= \mathbb{E}_{q_x(z)}[\log p(x|z)] - D_{KL}(q_x(z)\|p(z)) + D_{KL}(q_\phi(z|x)\|p(z)) \\ &= \mathbb{E}_{\epsilon\sim\mathcal{N}(0,1)}[\log p_\theta(x|z)] - D_{KL}(q_\phi(z|x)\|p(z|x)) \end{aligned}$$

In practice, $q_\phi(z|x)$ is often parameterized as $\mathcal{N}(\mu_\phi(x), \sigma^2_\phi(x))$. The ELBO thus comprises two parts: the reconstruction cost, which maximizes the conditional log probability of reconstructing data $x$ from latent code $z$, and the regularization cost, $D_{KL}(q_\phi(z|x)\|p(z))$, which regularizes the encoder distribution to be as close as possible to the prior distribution.

Overall, the VAE approach aims to maximize data likelihood by optimizing the variational lower bound, learning the encoding and decoding functions that define the probabilistic relationship between data and latent representations [165, 95].

## Autoregressive Model

Autoregressive models are pivotal in the domain of generative modeling, with substantial influence in the field, especially in the advancements of transformer architectures.

These models are based on a simple yet powerful principle and are often seen as specialized instances within the broader autoregressive framework.

A key implementation of autoregressive models is the MADE (Masked Auto-encoder for Distribution Estimation) model [111]. MADE represents a nuanced variation of Multilayer Perceptrons (MLPs), incorporating unique architectural modifications. It employs specific masking within the neural network layers to ensure that the autoregressive property is maintained, where future elements in a sequence do not influence the prediction of current elements.

The probability model in MADE can be generalized for a sequence of $n$ elements as follows:

$$p_\theta(x_1, x_2, \ldots, x_n) = \prod_{i=1}^{n} p_\theta(x_i | x_1, x_2, \ldots, x_{i-1})$$

This model is adept at learning conditional probabilities, enabling the autoregressive generation of subsequent values in the sequence $x_i$.

Another significant autoregressive model is PixelCNN, where convolutional layers are masked to ensure conditional generation of pixels, based on the spatial location of each pixel in an image [239, 312].

## Normalizing flow

Normalizing flow models, a notable category of generative models, stand out for their capacity to sample and evaluate probability densities. Originating from a simple distribution, such as Gaussian, these models employ a series of invertible functions to transform it. This unique approach facilitates the direct computation of probability density transformations.

Mathematically, the fundamental operation of normalizing flows is represented as:

$$z = f_\theta(x), \quad p_\theta(x)\, dx = p(z)\, dz, \quad p_\theta(x) = p(f_\theta(x)) \left| \frac{\partial f_\theta(x)}{\partial x} \right|$$

In this framework, $f_\theta$ is a composition of multiple invertible functions, typically implemented through neural networks. By stacking these functions, the expressiveness of the model is significantly enhanced:

$$x \to f_1 \to f_2 \to \ldots \to f_k \to z$$

Figure 1.2: Transformation sequence in a normalizing flow model, where the input $x$ is progressively transformed through a series of invertible functions $f_1, f_2, \ldots, f_k$, culminating in the output $z$. Each $f_i$ represents a key step in the flow, contributing to the comprehensive transformation from initial input to final output.

$$\begin{aligned} z &= f_k \circ f_{k-1} \circ \cdots \circ f_1(x) \\ x &= f_1^{-1} \circ f_2^{-1} \circ \cdots \circ f_k^{-1}(z) \end{aligned}$$

The model's probability density is given by the following expression:

$$\log p_\theta(x) = \log p(z) + \sum_{i=1}^{k} \log \det \left| \frac{\partial f^i}{\partial f^{i-1}} \right|$$

A pivotal strategy in constructing these invertible functions involves splitting the input $x$ into two components, $x = (x_1, x_2)$, and transforming them into $y = (y_1, y_2)$ using a combination of simple invertible functions and more complex neural networks. This approach, known as 'coupling flows', exemplifies the flexibility and innovation in normalizing flow design [92].

Moreover, models like NICE (Non-linear Independent Components Estimation) and RealNVP (Real-valued Non-Volume Preserving) utilize affine coupling to build these invertible transformations. The procedure is as follows:

$$\begin{aligned} z_{1:d/2} &= x_{1:d/2} \\ z_{d/2:d} &= x_{d/2:d} \cdot s_\theta \left( x_{1:d/2} \right) + t_\theta \left( x_{1:d/2} \right) \\ \frac{\partial z}{\partial x} &= \begin{pmatrix} I & 0 \\ \frac{\partial z_{d/2:d}}{\partial x_{1:d/2}} & \operatorname{diag} \left( s_\theta \left( x_{1:d/2} \right) \right) \end{pmatrix} \end{aligned} \tag{1.2.1}$$

The Jacobian matrix's upper triangular structure facilitates the computation of its determinant. The non-zero function $s_\theta$, often implemented using the exponential function to ensure positivity, is crucial for maintaining the invertibility that is central to normalizing flows [91, 92].

## Autoregressive flow

Autoregressive flow [145] represents an advanced topic in the field of generative modeling, integrating the principles of autoregressive models with those of normalizing flows, as implied by its name.

In autoregressive flow, the modeling process is as follows:

$$x_1 \sim p_\theta(x_1), \quad x_2 \sim p_\theta(x_2 \mid x_1), \quad x_3 \sim p_\theta(x_3 \mid x_1, x_2)$$

This model leverages the capabilities of normalizing flows in an autoregressive manner for sampling:

$$x_1 = f_\theta^{-1}(z_1), \quad x_2 = f_\theta^{-1}(z_2 \mid x_1), \quad x_3 = f_\theta^{-1}(z_3 \mid x_1, x_2)$$

However, one limitation of this approach is its inherently sequential nature, which precludes parallel processing during sampling.

To address this limitation, the concept of 'Inverse Autoregressive Flow' (IAF) was introduced. In IAF, the sampling process does not directly condition on the previous samples but instead on the preceding latent codes. The sampling formula for IAF is as follows:

$$x_1 = f_\theta(z_1), \quad x_2 = f_\theta(z_2 \mid z_1), \quad x_3 = f_\theta(z_3 \mid z_1, z_2)$$

This modification allows for more efficient sampling while still adhering to the autoregressive structure [164].

## Implicit Model — GAN (Generative Adversarial Networks)

Generative Adversarial Networks (GANs) [117] consist of two distinct networks: a generator and a discriminator. These networks have opposing goals: the generator aims to produce images that deceive the discriminator, while the discriminator strives to differentiate between generated and real images. From a game theory perspective, the objective of this adversarial process can be formulated as:

$$\min_G \max_D \mathbb{E}_{x\sim\text{data}}[\log D(x)] + \mathbb{E}_{z\sim p(z)}[\log(1 - D(G(z)))] \tag{1.2.2}$$

In this equation, the generator $G$ seeks to minimize the objective, whereas the discriminator $D$ aims to maximize it. The first term encourages the discriminator

to recognize real data accurately, while the second term pushes the discriminator to identify fake data created by the generator.

There are several variants of GANs, one of which is the Wasserstein GAN (W-GAN) [15]. This variant introduces the concept of the Wasserstein distance, also known as Kantorovich-Rubinstein duality:

$$W(p_r, p_g) = \sup_{\|f\|_L \leq 1} \mathbb{E}_{x \sim p_r}[f(x)] - \mathbb{E}_{x \sim p_g}[f(x)] \tag{1.2.3}$$

$$\min_G \max_D \mathbb{E}_{x \sim p_r}[D(x)] - \mathbb{E}_{x \sim p_g}[D(G(x))] \tag{1.2.4}$$

The first equation defines the Wasserstein-1 distance between the real data distribution $p_r$ and the generated data distribution $p_g$, using a supremum over all 1-Lipschitz functions $f$. The second equation presents the optimization objective for W-GANs, with $G$ as the generator and $D$ as the discriminator.

Another notable variant is the Wasserstein GAN with Gradient Penalty (WGAN-GP). This model introduces a penalty term to constrain the gradients, aiming to maintain them close to unity:

$$\lambda \mathbb{E}_{\hat{x} \sim p_{\hat{x}}} \left( \|\nabla_{\hat{x}} D(\hat{x}) - I\| \right)^2$$

Here, $\hat{x}$ represents a linear interpolation between samples $x$ and $\tilde{x}$, weighted by $\epsilon$. This penalty term helps to stabilize the training of WGANs by regularizing the discriminator's gradients [123].

## 1.3 Reinforcement Learning

Reinforcement Learning (RL) distinguishes itself as a notably flexible approach in the machine learning landscape, particularly when contrasted with traditional methods. It is fundamentally structured around the concept of a Markov Decision Process (MDP), which includes key components like state space, action space, and reward mechanisms. RL's mathematical underpinnings are closely related to optimal control problems, typically focusing on discrete decision-making processes.

In the classical RL setup, an agent dynamically interacts with its environment. At each decision point, or state, the agent selects an action from a set of possible actions. The environment, in response, provides a reward based on the action's outcome and

transitions to a new state. The agent's primary objective is to develop a policy that maximizes cumulative rewards or returns over time.

The field of RL is rich with extensive research and development. Two fundamental algorithms that form the basis of many advanced RL techniques are policy gradient and Q-learning. Policy gradient methods focus on directly optimizing the policy to maximize the expected return. On the other hand, Q-learning leverages the Bellman equation to find optimal policy solutions, improving the learning process's efficiency and effectiveness [296, 324].

Reinforcement learning (RL) uniquely sets itself apart from supervised and unsupervised learning paradigms. Unlike supervised learning, which requires a dataset with corresponding labels, and unsupervised learning, which typically involves offline data processing without environment interaction, RL is fundamentally a time-series based process that necessitates active interaction with an environment.

Central to the concept of RL is the Markov Decision Process (MDP). An MDP is a mathematical framework for modeling decision-making situations where outcomes are partly random and partly under the control of a decision-maker. It is defined as $M = \{S, A, T, r\}$, where:

- $S$ represents the state space, comprising states $s \in S$ that can be either discrete or continuous.
- $A$ denotes the action space, containing actions $a \in A$ that can also be discrete or continuous.
- $T$ is the transition operator, described as a tensor that dictates the probabilities of moving from one state to another given a specific action.

Given these definitions, the probabilities of being in a state $s_t$ at time $t$ and taking an action $a_t$ are denoted as $\mu_{t,j} = p(s_t = j)$ and $\xi_{t,k} = p(a_t = k)$, respectively. The transition probability from state $j$ to state $i$ upon taking action $k$ is represented as $T_{i,j,k} = p(s_{t+1} = i | s_t = j, a_t = k)$.

These components interact as follows:

$$\mu_{t,i} = \sum_{j,k} T_{i,j,k} \mu_{t,j} \xi_{t,k}$$

This equation captures the essence of the MDP framework in RL, describing the dynamics of how states evolve over time based on the actions taken and the inherent probabilities of transitioning between states.

In the realm of stochastic decision-making and reinforcement learning, Markov Decision Processes (MDPs) and Partially Observed Markov Decision Processes (POMDPs) play pivotal roles. These models are integral in understanding and designing systems that interact with complex and potentially uncertain environments.

An MDP is typically defined as $M = \{S, A, T, r\}$, encompassing several key components. $S$ is the state space, consisting of states $s$ which can be either discrete or continuous. $A$ represents the action space, with actions $a$ that can also be discrete or continuous. $T$ is the transition operator, a tensor that describes the probabilities of transitioning from one state to another given a specific action. $r$ is the reward function, mapping state-action pairs to real numbers ($r : S \times A \to \mathbb{R}$), providing the reward for each state-action pair $(s_t, a_t)$.

These state transitions and interactions can be visualized as follows:

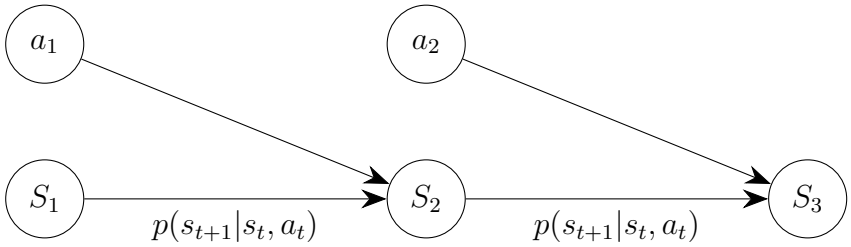



Figure 1.3: A schematic representation of state transitions in a Markov Decision Process, illustrating the dynamics between states ($S_1$, $S_2$, $S_3$) and actions ($a_1$, $a_2$) with probabilistic transitions.

Moving to a more complex scenario, we encounter the Partially Observed Markov Decision Process (POMDP), formally described by $M = \{S, A, O, T, \mathcal{E}, r\}$. In addition to the components found in an MDP, a POMDP introduces: 1) $O$, the observation space, with observations $o$ that can be discrete or continuous; 2) $\mathcal{E}$, the emission probability, which describes the likelihood $p(o_t|s_t)$ of obtaining an observation $o_t$ given the current state $s_t$.

The POMDP framework is particularly suited to scenarios where the agent does not have full observability of the state space, reflecting many real-world situations where decision-making is based on partial or noisy information [154, 296].

In the context of reinforcement learning, the probability of observing a particular trajectory under a policy parameterized by $\theta$ can be formally expressed as follows:

$$p_\theta(\tau) = p_\theta(s_1, a_1, \ldots, s_T, a_T) = p(s_1) \prod_{t=1}^{T} \pi_\theta(a_t|s_t) p(s_{t+1}|s_t, a_t)$$

In this expression, $\tau$ denotes a specific trajectory, while $s_t$ and $a_t$ represent the state and action at time step $t$, respectively. The term $\pi_\theta(a_t|s_t)$ is the policy, determined by parameter $\theta$, that defines the likelihood of taking action $a_t$ in state $s_t$. The transition probability $p(s_{t+1}|s_t, a_t)$ signifies the probability of moving from state $s_t$ to state $s_{t+1}$ upon taking action $a_t$. Additionally, $r(s_t, a_t)$ is the reward received for taking action $a_t$ in state $s_t$.

The objective in reinforcement learning is to find the optimal policy parameters $\theta^*$ that maximize the expected cumulative rewards over these trajectories. This is mathematically formulated as:

$$\theta^* = \arg\max_\theta \mathbb{E}_{\tau \sim p_\theta(\tau)} \left[ \sum_t r(s_t, a_t) \right]$$

The optimal policy $\theta^*$ is thus determined by maximizing the expected sum of rewards collected over the course of the trajectory.

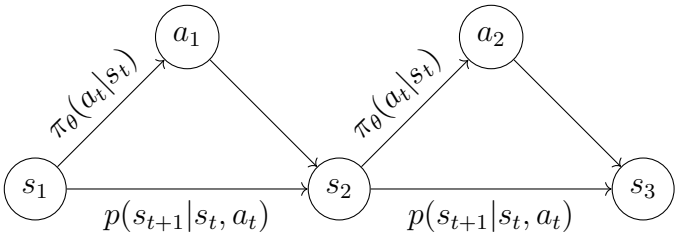



Figure 1.4: A representation of the state and action transitions within a reinforcement learning environment, illustrating the progression of states $s_t$ and actions $a_t$ along with the associated policy probabilities $\pi_\theta(a_t|s_t)$ and state transition probabilities $p(s_{t+1}|s_t, a_t)$.

In reinforcement learning, the objective function is often understood in terms of the Q-function and value function. The Q-function, defined as $Q^\pi(s_t, a_t) = \sum_{t'=t}^{T} \mathbb{E}_{\pi_\theta}[r(s', a')|s_t, a_t]$, represents the expected total reward for taking action $a_t$ in state $s_t$ and following policy $\pi_\theta$ thereafter. It quantifies the cumulative reward from state $s_t$ upon selecting action $a_t$ and adhering to the policy. Conversely, the value function, expressed as $V^\pi(s_t) = \sum_{t'=t}^{T} \mathbb{E}_{\pi_\theta}[r(s', a')|s_t]$, indicates the expected total reward starting from state $s_t$. This value is related to the Q-function and can be expressed

as $V^{\pi}(s_t) = \mathbb{E}_{a_t \sim \pi_\theta(a_t|s_t)}[Q^{\pi}(s_t, a_t)]$. The primary aim in reinforcement learning is to optimize the expected value from the initial state $s_1$, denoted as $\mathbb{E}_{s_1 \sim p(s_1)}[V^{T}(s_1)]$, thereby maximizing the overall return of the policy starting from the initial state in the environment.

Q-learning, a model-free reinforcement learning algorithm introduced by Watkins and Dayan [324], plays a vital role in the field. It enables the determination of the optimal policy by learning the Q-function without the need for a model of the environment. This algorithm is key in comparing the expected utility of different actions and has been foundational in advancing the applications of reinforcement learning.

Reinforcement learning algorithms can be broadly categorized into three distinct types based on their approach to policy and value function:

1. **Value Function-Based Algorithms**[324]: These algorithms focus on estimating the value function $V(s)$ or the Q function $Q(s, a)$. Once the model is fitted to ascertain these values, the policy is defined as $\pi(s) = \arg\max_a Q(s, a)$. This strategy outlines an optimal policy $\pi(s)$ for each state $s$, where the action $a$ is chosen to maximize the Q-value $Q(s, a)$ for that state. Essentially, it dictates that the best action in state $s$ is the one offering the highest expected total reward. This approach is exemplified in Q-learning, a model-free reinforcement learning algorithm, where the agent seeks to maximize cumulative rewards over time.

2. **Direct Policy Gradients**[297]: This method employs a more straightforward approach. It first estimates the total expected returns $R_t = \sum_t r(s_t, a_t)$ using available data. Following this, it applies policy gradients to directly update the model parameters: $\theta \leftarrow \theta + \alpha \nabla_\theta \mathbb{E}[\sum_t r(s_t, a_t)]$. This process involves adjusting the policy parameters $\theta$ directly based on the gradient of the expected rewards.

3. **Actor-Critic Algorithms**[175]: These algorithms attempt to blend the strengths of both value function-based and direct policy gradient methods. They utilize data to fit the value function or Q function, similar to value function-based algorithms. However, like direct policy gradients, they then take the gradient of the Q function to update the policy parameters. This hybrid approach leverages the advantages of both value estimation and direct policy optimization, aiming to balance and improve the overall learning process.

Each of these algorithm types offers unique advantages and is suited to different kinds of reinforcement learning problems, making them critical tools in the arsenal of machine learning methodologies.

## Imitation learning

Imitation learning, often referred to as behavior cloning, is a technique in machine learning where a model is trained to emulate expert behavior. The fundamental approach involves collecting a dataset comprising pairs of actions and observations, denoted as $a_t$ and $o_t$ respectively. This dataset is then used to train a supervised learning model to derive a policy $\pi_\theta$, which maps observations to actions, formalized as $\pi_\theta(a_t|o_t)$.

One significant challenge in imitation learning is the accumulation of errors leading to a deviation of the learned policy's rollout from the expert's distribution. This phenomenon is known as covariate shift or distribution mismatch. To address this issue, the DAgger [267] (Dataset Aggregation) algorithm was introduced. The objective of DAgger is to iteratively refine the training dataset to better match the distribution encountered during the policy's rollout.

In practice, DAgger involves executing rollouts of the learned policy, collecting the resulting data, and then augmenting the original training dataset with these new observations annotated with expert actions. This enriched dataset is subsequently used to retrain the policy. This process is repeated iteratively until the policy converges to satisfactory performance levels, effectively aligning the policy's distribution with that of the expert's.

This iterative training approach, where the model is continuously refined with data sampled from its own distribution, helps in mitigating the issue of covariate shift, resulting in more robust and reliable policies.

## Policy gradient algorithms

Policy gradient methods in reinforcement learning focus on directly optimizing the policy function by adjusting its parameters to maximize expected cumulative rewards. The objective function for policy gradient can be written as follows:

$$\theta^* = \arg\max_\theta \mathbb{E}_{\tau \sim p_\theta(\tau)} \left[ \sum_t r(s_t, a_t) \right] \tag{1.3.1}$$

where $\tau$ denotes a trajectory, $s_t$ and $a_t$ are the state and action at time $t$ respectively, and $r(s_t, a_t)$ is the reward function. The expectation $\mathbb{E}$ is taken over the distribution of trajectories induced by the policy $p_\theta$.

Depending on the nature of the problem, particularly whether the horizon is infinite or finite, the objective function can be adapted. For an infinite horizon, it is defined as:

$$\theta^* = \arg\max_\theta \mathbb{E}_{(s,a)\sim p_\theta(s,a)}[r(s,a)] \tag{1.3.2}$$

In the finite horizon case, the objective is:

$$\theta^* = \arg\max_\theta \sum_{t=1}^{T} \mathbb{E}_{(s_t,a_t)\sim p_\theta(s_t,a_t)}[r(s_t,a_t)] \tag{1.3.3}$$

The policy gradient, often implemented through the REINFORCE algorithm [334], is a direct method of policy optimization. This algorithm involves sampling trajectories $\tau^i$ from the policy $\pi_\theta(a_t|s_t)$, computing the policy gradient using on-policy data:

$$\nabla_\theta J(\theta) \approx \sum_i \left(\sum_t \nabla \log \pi(a_t^i|s_t^i)\right)\left(\sum_t r(s_t^i,a_t^i)\right) \tag{1.3.4}$$

and then updating the parameters using gradient ascent:

$$\theta \leftarrow \theta + \alpha \nabla_\theta J(\theta) \tag{1.3.5}$$

where $\alpha$ is the learning rate.

In policy gradient methods, a common challenge is the high variance of gradient estimates, which can impede efficient learning. Two techniques are commonly employed to mitigate this issue: the introduction of "reward to go" and the use of a baseline.

In reinforcement learning, particularly in the context of policy gradient methods, the concept of "reward to go" and the use of a baseline are two critical techniques for enhancing the effectiveness of learning algorithms.

Traditionally, the policy gradient is formulated around the idea of total expected returns. However, this can be modified to include the concept of "reward to go," which refines the standard policy gradient formulation. In this approach, instead of using the total cumulative reward, the gradient is calculated based on the rewards accumulated from the current time step $t$ to the end of the episode. Mathematically, this is represented as:

$$\nabla_\theta J(\theta) \approx \frac{1}{N} \sum_{i=1}^{N} \left( \sum_{t=1}^{T} \nabla \log \pi_\theta(a_{i,t}|s_{i,t}) \right) \left( \sum_{t'=t}^{T} r(s_{i,t'}, a_{i,t'}) \right)$$

This method introduces the principle of causality, which asserts that future events cannot influence the past. By focusing only on rewards from the current timestep onwards, the "reward to go" reduces the variance in the gradient estimate, resulting in more stable and efficient learning.

Another significant technique is the introduction of a baseline to further reduce variance. A baseline is a term subtracted from the reward, which, while not affecting the expected value of the gradient, can greatly reduce its variance. The underlying principle is that subtracting a constant from the reward does not change the expectation of the gradient:

$$E[\nabla_\theta \log \pi_\theta(\tau) b] = b \nabla_\theta 1 = 0$$

Consequently, the policy gradient can be reformulated to include a baseline $b$, such as the average reward or the value function, leading to:

$$\nabla_\theta J(\theta) \approx \frac{1}{N} \sum_{i=1}^{N} \nabla_\theta \log \pi_\theta(\tau)[r(\tau) - b]$$

Incorporating these techniques — "reward to go" and baseline subtraction — into policy gradient methods substantially enhances the learning process, making it more efficient and robust against the inherent variances of training in complex environments.

These techniques are fundamental in the field of reinforcement learning, as they improve the efficiency and stability of policy gradient methods.

## Actor Critor

The Actor-Critic framework [174] is a prominent approach in reinforcement learning that combines policy-based and value-based methods. In the Actor-Critic architecture, the actor represents the policy function, and the critic estimates the value function.

The standard policy gradient formulation is refined in Actor-Critic methods by leveraging the Q-function for the expected reward-to-go:

$$Q(s_t, a_t) = \sum_{t'=t}^{T} \mathbb{E}_{\pi_\theta}[r(s_{t'}, a_{t'})|s_t, a_t] \tag{1.3.6}$$

This function represents the true expected cumulative reward starting from state $s_t$ and taking action $a_t$. To further reduce the variance in the policy gradient estimate, a baseline, typically the value function $V(s_t)$, is subtracted:

$$\nabla_\theta J(\theta) \approx \frac{1}{N} \sum_{i=1}^{N} \sum_{t=1}^{T} \nabla_\theta \log \pi_\theta(a_{i,t}|s_{i,t}) \left(Q^{\pi_\theta}(s_{i,t}, a_{i,t}) - V(s_{i,t})\right) \tag{1.3.7}$$

where the value function is defined as:

$$V(s_t) = \mathbb{E}_{a_t \sim \pi_\theta(a_t|s_t)}[Q(s_t, a_t)] \tag{1.3.8}$$

The difference $Q^\pi(s, a) - V^\pi(s)$, known as the advantage function $A^\pi(s, a)$, measures the relative benefit of choosing action $a$ in state $s$ compared to the average outcome from that state.

The Actor-Critic algorithm involves three primary steps:

- **Sample Generation**: Execute the current policy to collect data, including state transitions, actions, and rewards.
- **Model Fitting for $Q^\pi$ and $V^\pi$**: The Q function is updated using the formula $Q^\pi(s_t, a_t) = r(s_t, a_t) + V^\pi(s_{t+1})$, and the advantage function is computed as $A^\pi(s_t, a_t) = r(s_t, a_t) + V^\pi(s_{t+1}) - V^\pi(s_t)$.
- **Value Function Training**: Apply supervised regression to train the value function network. The training data consists of state and cumulative reward pairs. The loss function $L(\phi)$ for the value function is defined as:

$$L(\phi) = \frac{1}{2} \sum_i \|V_\phi^\pi(s_i) - y_i\|^2 \tag{1.3.9}$$

  where $y_i$ represents the bootstrapped target, calculated as $y_{i,t} = r(s_{i,t}, a_{i,t}) + \gamma V_\phi^\pi(s_{i,t+1})$, with $\gamma$ being the discount factor.

Bootstrapping improves the algorithm by incorporating the current estimate of the subsequent state's value:

$$y_{i,t} = r(s_{i,t}, a_{i,t}) + V_\phi^\pi(s_{i,t+1})$$

This bootstrapped target combines observed rewards with estimates of future values, leading to faster convergence, more stable updates, and direct use of previously fitted value functions. The discount version uses:

$$y_{i,t} \approx r(s_{i,t}, a_{i,t}) + \gamma V_\phi^\pi(s_{i,t+1})$$

where $\gamma$ is the discount factor.

## Q Learning

Q-Learning is a cornerstone of reinforcement learning, evolving from the concepts of policy iteration and value iteration.

Policy Iteration consists of two phases: policy evaluation and policy update. The policy evaluation is performed as follows:

$$V^\pi(s) \leftarrow r(s, \pi(s)) + \gamma \mathbb{E}_{s' \sim p(\cdot|s,\pi(s))}[V^\pi(s')] \tag{1.3.10}$$

The policy is then updated using:

$$\pi'(a_t|s_t) = \begin{cases} 1 & \text{if } a_t = \arg\max_a A^\pi(s_t, a_t) \\ 0 & \text{otherwise} \end{cases} \tag{1.3.11}$$

Value Iteration also involves two steps: updating the Q function and updating the value function. The Q function is updated using:

$$Q(s, a) \leftarrow r(s, a) + \gamma \mathbb{E}[V(s')] \tag{1.3.12}$$

and the value function is updated by:

$$V(s) \leftarrow \max_a Q(s, a) \tag{1.3.13}$$

After convergence, the policy is derived as:

$$\pi'(a_t|s_t) = \begin{cases} 1 & \text{if } a_t = \arg\max_a Q^\pi(s_t, a_t) \\ 0 & \text{otherwise} \end{cases} \tag{1.3.14}$$

where $A^\pi(s,a) = Q^\pi(s,a) - V^\pi(s)$.

Deep Q Learning, an extension of traditional Q Learning as described in Mnih et al. (2015) [220], involves several key steps. Initially, data collection is required, where a dataset is gathered, which may be independent of the current policy. This is followed by setting targets for the Q network. The target is set as $y_i \leftarrow r(s_i, a_i) + \gamma \max_{a'_i} Q_\phi(s'_i, a'_i)$. Subsequently, the parameters of the Q network are updated through regression. This involves minimizing the expression $\frac{1}{2}\sum_i \|Q_\phi(s_i, a_i) - y_i\|^2$. This method is categorized as off-policy, indicating that the data used need not originate from the current policy.

Enhancements to Q Learning have been introduced to address its limitations. One such enhancement is the use of a Replay Buffer, which significantly improves sample efficiency by allowing for multiple gradient descents per data point. Another notable enhancement is the implementation of a Target Network. This involves using a separate network for estimating values, defined as $y_i \leftarrow r(s_i, a_i) + \gamma \max_{a'_i} Q_{\phi'}(s'_i, a'_i)$. Furthermore, to maintain stability and efficiency in the learning process, Polyak averaging is applied to the parameters of the network, following the formula $\phi' \leftarrow \tau\phi' + (1-\tau)\phi$, with $\tau$ typically set at 0.999. These enhancements contribute to a more robust and efficient learning process in Q Learning algorithms.

## Natural Gradient

The concept of the natural gradient[7] arises in the context of optimization in machine learning, particularly within the realm of neural networks and reinforcement learning. It represents an advanced method for adjusting the parameters of a model in a way that accounts for the curvature of the parameter space. This stands in contrast to the standard gradient descent method, which does not take this curvature into account.

In traditional gradient descent, we update the parameters of our model by moving in the direction opposite to the gradient of the loss function. However, this approach assumes that the parameter space is Euclidean, where all directions are equally scaled, and straight-line distances are meaningful. But in many machine learning models, especially those involving probability distributions (like those in reinforcement learning), the space is not Euclidean and has a complex, curved geometry.

The natural gradient aims to address this by adjusting the update direction to be more appropriate for the geometry of the parameter space. When we speak of geometry in

parameter space, we're often referring to the information geometry, which considers how probability distributions change as we move through the space of parameters.

In reinforcement learning, especially in policy gradient methods, we want to update our policy parameters in such a way that we move to better policies, but without making too large of a step. A large step could lead to a drastically different policy that performs poorly, which is undesirable.

The natural gradient takes into account the Fisher information matrix, which represents the curvature of the space and provides a metric for measuring distances between probability distributions (such as policies). When we use the natural gradient, we effectively perform a more informed update that respects the structure of the policy space.

Mathematically, the natural gradient is defined as:

$$\tilde{\nabla}_\theta J(\theta) = F(\theta)^{-1} \nabla_\theta J(\theta)$$

where $\tilde{\nabla}_\theta J(\theta)$ is the natural gradient of the objective function $J$ with respect to the parameters $\theta$, $F(\theta)$ is the Fisher information matrix at $\theta$ and $\nabla_\theta J(\theta)$ is the standard gradient of $J$.

Using the natural gradient instead of the standard gradient can lead to more efficient and stable learning, as it adjusts the step size automatically to account for the local geometry of the parameter space. This means it can take larger steps in flat directions (where parameters can be changed without significantly affecting the output) and smaller steps in steep directions (where changes have a large impact).

Algorithms like Natural Actor-Critic [245] and Trust Region Policy Optimization (TRPO) leverage the concept of the natural gradient to improve learning by ensuring that updates to the policy parameters do not change the behavior of the policy too drastically, which is often quantified by keeping the KL-divergence between consecutive policies below a certain threshold.

## Trust Region Policy Optimization (TRPO)

The Trust Region Policy Optimization (TRPO) [279] is a sophisticated approach for optimizing policies in reinforcement learning. At its core, TRPO aims to improve the policy by maximizing an objective function, which typically involves expected returns. However, unlike standard policy gradient methods that can lead to destructive large

updates, TRPO operates within a trust region—a constrained optimization space—to ensure stability and reliability of learning.

This trust region is defined by the Kullback-Leibler (KL) divergence, which measures how one probability distribution diverges from a second, reference probability distribution. In the context of TRPO, it ensures that the updated policy does not deviate too significantly from the current policy, maintaining a certain level of 'closeness'. This constraint is crucial for avoiding the pitfalls of large policy updates, which can lead to performance collapse.

The objective function in TRPO is typically augmented with a penalty term or is subject to a constraint that limits the KL divergence between the new policy and the old policy. The algorithm then uses this constraint to form a trust region, which defines the boundaries within which the policy can be updated. By doing so, TRPO seeks to take the largest possible improvement step on the policy without violating the trust region—hence optimizing policy performance while maintaining a degree of safety.

Mathematically, the optimization problem TRPO solves can be expressed as:

$$\max_{\theta} \mathbb{E}_{s,a\sim\pi_{\theta_{old}}} \left[ \frac{\pi_\theta(a|s)}{\pi_{\theta_{old}}(a|s)} A^{\pi_{\theta_{old}}}(s,a) \right]$$

subject to:

$$\mathbb{E}_{s\sim\pi_{\theta_{old}}} \left[ D_{KL}(\pi_{\theta_{old}}(\cdot|s) \| \pi_\theta(\cdot|s)) \right] \leq \delta$$

Here, $\theta$ represents the policy parameters, $\pi$ denotes the policy, $A$ stands for the advantage function, and $\delta$ is a predefined threshold for the KL divergence that determines the size of the trust region.

TRPO's effectiveness lies in its balance between exploration and exploitation, enabled by its trust region approach. This balance is particularly useful in environments with high-dimensional state and action spaces, where it is essential to make consistent progress while avoiding drastic policy changes that could lead to unpredictable or suboptimal behavior.

## Proximal Policy Optimization (PPO)

Proximal Policy Optimization (PPO) [278] is an algorithm that builds upon the ideas of TRPO, aiming to simplify the implementation while preserving the core principle

of stable policy updates. PPO has become one of the most popular algorithms in reinforcement learning due to its effectiveness and ease of use.

The primary innovation of PPO is the introduction of a clipped surrogate objective function. This function limits the amount of change to the policy by clipping the policy ratio, which prevents the new policy from being too far from the old policy. The clipping mechanism serves a similar purpose to the trust region in TRPO, but it is easier to compute and implement.

The clipped surrogate objective function of PPO is as follows:

$$L^{CLIP}(\theta) = \hat{\mathbb{E}}_t \left[ \min(r_t(\theta)\hat{A}_t, \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t) \right]$$

where $r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)}$ is the probability ratio of the action under the new and old policies, $\hat{A}_t$ is an estimator of the advantage function at time $t$, $\epsilon$ is a hyperparameter, typically a small value like 0.1 or 0.2, that defines the clip range to limit the ratio $r_t(\theta)$ and the min operation ensures that the objective is the lesser of the unclipped and clipped objectives, which bounds the policy update.

PPO retains the advantage of TRPO in that it constrains policy updates to avoid large and potentially harmful deviations. However, PPO uses a first-order optimization method, which is computationally less expensive and simpler than the second-order methods used in TRPO. This makes PPO more practical for a wider range of problems and more accessible for implementation.

Additionally, PPO is characterized by alternating between sampling data through interaction with the environment and optimizing the clipped surrogate objective. This alternating process is typically repeated for a fixed number of epochs, contrasting with TRPO's single large update. This procedure allows PPO to efficiently use the collected data to refine the policy iteratively.

In practice, PPO has demonstrated robust performance across a variety of domains, from simulated robotic control to complex strategy games. Its balance of sample efficiency, ease of implementation, and stability has established it as a go-to algorithm for many reinforcement learning practitioners.

## Soft Actor-Critic (SAC)

Soft Actor-Critic (SAC) [126] is a state-of-the-art algorithm in the field of reinforcement learning that stands out due to its sample efficiency and stability. It is an off-policy actor-critic method that incorporates the maximum entropy framework,

which encourages exploration by not only seeking to maximize expected return but also maximizing entropy. In other words, SAC aims for a policy that is both high-performing and as random as possible.

Soft Actor-Critic (SAC) is a reinforcement learning algorithm with several key components. Firstly, it employs an actor-critic architecture. In this setup, the actor suggests actions based on the given state, while the critic evaluates these actions by estimating their Q-values. SAC uses two Q-value functions to mitigate the positive bias in the policy improvement step, a known issue that can degrade the performance of value-based methods.

Another essential component of SAC is its soft policy update strategy. The algorithm updates policies by considering the entropy of the policy within the optimization objective, leading to more exploratory behavior. This approach is particularly advantageous in complex environments where efficient exploration is crucial for finding effective policies.

Additionally, SAC operates on an off-policy basis. It learns from old experiences stored in a replay buffer, making it more sample-efficient than on-policy algorithms. This efficiency is due to the ability to use each sample multiple times for learning.

Finally, SAC incorporates the maximum entropy framework. This framework encourages the agent to act as randomly as possible, to maximize entropy, while still achieving success in the task at hand. The result is the development of robust and versatile policies that can adapt to environmental changes, enhancing the overall effectiveness of the algorithm.

The objective function of SAC is designed to optimize both the expected return and the entropy of the policy:

$$J(\pi) = \mathbb{E}_{(s_t,a_t)\sim\rho_\pi}\left[\sum_t \gamma^t \left(R(s_t,a_t) + \alpha\mathcal{H}(\pi(\cdot|s_t))\right)\right]$$

where $R(s_t, a_t)$ is the reward at timestep $t$, $\gamma$ is the discount factor, $\alpha$ is the temperature parameter that determines the relative importance of the entropy term against the reward (with higher values favoring exploration) and $\mathcal{H}$ denotes the entropy of the policy $\pi$.

The algorithm typically includes an automatic entropy adjustment mechanism, where the temperature parameter $\alpha$ is learned adaptively rather than being set manually. This allows the algorithm to automatically balance exploration and exploitation based on the specific environment it is interacting with.

SAC has shown superior performance in a variety of continuous control tasks, particularly those with high-dimensional action spaces. Its ability to learn effective policies while maintaining a high degree of exploration makes it a powerful tool for solving complex reinforcement learning problems.

## Reinforcement Learning for Quantum Applications

The integration of reinforcement learning (RL) into the realms of quantum computing and quantum control marks a significant stride in contemporary research. This interdisciplinary fusion holds immense promise for revolutionizing quantum technologies through innovative optimization strategies for quantum systems.

Leveraging the strength of RL algorithms, which excel in learning optimal strategies through trial and error, this approach adeptly adapts to the unique intricacies of quantum mechanics. Since the pioneering efforts of researchers like Marin Bukov[48], who initially applied RL techniques to quantum control, the field has seen remarkable advancements. Bukov's work set a solid foundation for understanding how complex quantum systems can be manipulated and optimized using reinforcement learning.

Subsequent research has diversified the application of RL in quantum systems. Researchers have crafted various methods to tackle different challenges within quantum computing and control. These methodologies span a broad spectrum, including optimizing quantum circuit designs, enhancing the efficiency of quantum simulations, and refining the accuracy of quantum measurements.

A notable achievement in this domain is the application of RL in quantum error correction[107]. RL algorithms have proven to be effective in devising strategies to reduce errors in quantum computers, a critical milestone in achieving reliable and scalable quantum computing.

Dynamic quantum control is another area where RL has made significant inroads. Here, RL algorithms are trained to adjust quantum systems in real-time, learning to manipulate quantum states and processes to achieve specific outcomes. This kind of dynamic control is vital for executing tasks like quantum gate implementation and state preparation.

Beyond theoretical exploration, RL's integration into quantum technology has also seen practical applications. Experimental setups have demonstrated the feasibility of RL-guided quantum control under lab conditions, reinforcing the potential of RL as a crucial tool in the quantum scientist's arsenal.

In conclusion, the confluence of reinforcement learning and quantum technology is

a rapidly growing field with enormous potential. The foundational work of Marin Bukov and the continuous evolution of this research highlight the significant role RL could play in advancing quantum computing and control, paving the way for transformative developments in quantum technologies.

## 1.4 Deep Autoregressive model

Deep autoregressive models are integral to deep learning, particularly in the realm of sequential data processing. These models are primarily used in sequence-to-sequence tasks and have significantly contributed to advancements in fields such as natural language processing and time-series analysis.

Deep autoregressive models function by predicting future elements in a sequence based on learned dependencies from previous elements. Their autoregressive nature means each output is conditioned on preceding outputs, creating a dependency chain within the sequence. This approach allows the models to effectively capture and utilize the context inherent in sequential data.

The core mechanism of deep autoregressive models involves factorizing the probability of a sequence into a product of conditional probabilities. Mathematically, for a sequence $x_1, x_2, ..., x_n$, the joint probability is expressed as:

$$P(x_1, x_2, ..., x_n) = \prod_{t=1}^{n} P(x_t | x_1, ..., x_{t-1})$$

To model these conditional probabilities, deep learning architectures such as Recurrent Neural Networks (RNNs) [119], Long Short-Term Memory networks (LSTMs) [142], and Gated Recurrent Units (GRUs) [70] are commonly used.

In terms of applications, deep autoregressive models have shown remarkable utility and versatility. They are extensively used in natural language processing tasks, including text generation, machine translation, and speech recognition. Beyond language processing, these models are also applied in image generation and time-series forecasting, demonstrating their broad applicability in handling various types of sequential data.

While the concept of deep autoregressive models is foundational, the introduction of Transformers has further revolutionized the field. Transformers, with their attention mechanisms [314], have provided an alternative and often more effective approach to

dealing with sequential data, particularly in tasks where long-range dependencies are crucial.

## Transformer

The Transformer model, introduced by Vaswani et al. [314] in their groundbreaking paper "Attention Is All You Need" in 2017, has revolutionized the field of sequence modeling, especially in natural language processing (NLP). This model marked a significant departure from traditional recurrent architectures, like LSTMs and GRUs, by relying entirely on attention mechanisms, particularly self-attention, to capture global dependencies between input and output.

The architecture of the Transformer consists of two main components: an encoder and a decoder. Each layer of the encoder contains two sub-layers: a multi-head self-attention mechanism and a position-wise feed-forward network, both surrounded by layer normalization and residual connections. The decoder mirrors this structure but adds a third sub-layer that performs multi-head attention over the encoder's output.

One of the key features of Transformers is their ability to process entire sequences simultaneously, unlike autoregressive models that process data sequentially. This parallel processing capability significantly enhances efficiency and scalability. Additionally, positional encodings are added to the input to provide the model with information about the position of each element in the sequence, compensating for the absence of recurrence.

Transformers have dramatically impacted various fields, achieving state-of-the-art results in numerous NLP tasks. They form the backbone of prominent models like BERT [87], GPT [253], and T5 [254]. Moreover, the success of Transformers has spurred research into applying their architecture beyond NLP, including in areas like computer vision and audio processing.

In comparison to deep autoregressive models, which focus on capturing sequential dependencies through recurrence, Transformers handle long-range dependencies more efficiently through attention mechanisms and parallel processing. While autoregressive models are interpretable and effective in certain contexts, they often struggle with longer sequences due to issues like vanishing gradients. Transformers, by contrast, address many of these limitations, offering a scalable and potent solution for processing large amounts of sequential data.

## Large Language Models (LLMs)

Large Language Models (LLMs) have become a pivotal topic in deep learning, particularly in the evolution and application of natural language processing technologies. These models, which are primarily based on the transformer architecture, have revolutionized our approach to processing and generating natural language.

LLMs like GPT (Generative Pre-trained Transformer), BERT (Bidirectional Encoder Representations from Transformers), and others represent a significant leap in neural network-based models. They are characterized by their immense size in terms of the number of parameters and the scale of the training data. These models are trained on extensive text corpora, learning complex tasks such as predicting the next word in a sentence, generating coherent text, or understanding the context of language. This training typically involves unsupervised learning on large datasets, followed by fine-tuning for specific applications.

In terms of capabilities, LLMs excel in a variety of language-related tasks including text generation, translation, summarization, question answering, and more. They have been instrumental in advancing areas like conversational AI, content creation, and information extraction. Many LLMs, especially those used for text generation like the GPT series, are autoregressive, generating text one word at a time and basing each new word on the sequence generated so far. This autoregressive nature allows them to produce text that is coherent and contextually relevant over long sequences.

The backbone of LLMs' success is the transformer architecture, renowned for its ability to process sequences in parallel and effectively capture long-range dependencies, making it ideally suited for the complexities of natural language. The attention mechanisms within transformers enable LLMs to focus on relevant parts of the input when performing various language tasks.

However, LLMs are not without challenges. Issues such as bias in training data, ethical considerations, and the computational cost of training and deploying such large models are significant areas of concern. Current research is directed towards making these models more efficient, ethical, and robust.

The evolution from deep autoregressive models to transformers, and now to Large Language Models, marks a significant advancement in deep learning and natural language processing. While autoregressive models laid the initial groundwork for understanding sequential data, transformers provided a more efficient and effective framework, which was then leveraged to develop LLMs. These models have expanded the horizons of language understanding and generation, signaling a new era in AI's capabilities in human language interaction. As the field advances, LLMs are

expected to bring even more sophisticated applications and interactions, potentially transforming numerous facets of technology and society.

## Applications of Autoregressive Models in Physics

Autoregressive models are not only widely recognized for their applications in statistics and machine learning but also hold significant value in the realm of physics. These models provide a robust framework for understanding and predicting a range of physical phenomena, leveraging the concept that the future state of a system can often be modeled as a function of its past states.

In the field of physics, many systems exhibit temporal or spatial dependencies, and it is here that autoregressive models particularly excel. By utilizing past observations of various physical quantities — such as temperature, pressure, or more complex variables like quantum states or field distributions — these models are adept at generating predictions about future states. This predictive capability is invaluable, not only for theoretical studies but also for practical applications across disciplines like climate modeling, astrophysics, and materials science.

For example, in climate science, autoregressive models are used to forecast weather patterns or long-term climate changes, drawing on historical data. By analyzing the evolution of variables like temperature and humidity over time, these models can provide crucial insights into future climatic conditions. In astrophysics, they assist in predicting the behavior of celestial bodies or the evolution of cosmic events based on past observations.

The realm of quantum mechanics and particle physics, where traditional deterministic models often fall short, is another area where autoregressive models prove their mettle. They offer a probabilistic approach to understanding the dynamics of particles and fields, predicting the probability distributions of various quantum states[54], or modeling the behavior of subatomic particles in accelerators.

Autoregressive models' flexibility and adaptability make them highly suitable for a broad spectrum of applications in physics. They excel at capturing the inherent dependencies in physical systems and extrapolating from known data, thus serving as a powerful tool for both theoretical exploration and practical problem-solving in the physical sciences. As the collection and analysis of vast data sets in various physics fields continue to grow, the importance of autoregressive models becomes ever more pronounced. They offer a blend of simplicity and effectiveness, crucial for modeling the complexities of the physical world.

## 1.5 Black box optimization

Black box optimization algorithms are pivotal in optimizing functions where the internal mechanisms are either unknown or inaccessible. These algorithms are diverse, each with unique strengths and suitable for various problem types. Notable algorithms in black box optimization include:

Several advanced optimization algorithms have been developed, each with unique characteristics and applications. Genetic Algorithms (GA)[143], inspired by natural evolution, utilize selection, crossover, and mutation processes to optimize solutions. Particle Swarm Optimization (PSO)[158] mimics the behavior of swarms, where particles traverse the solution space to find optimal solutions. Simulated Annealing (SA)[167], drawing parallels from metallurgical annealing, probabilistically navigates the solution space, aiding in escaping local minima.

Bayesian Optimization[258] employs Bayesian methods to model the objective function, enhancing the sampling process's efficiency. Differential Evolution (DE) is particularly effective for multi-dimensional real-valued functions, involving strategies like mutation and crossover. The Covariance Matrix Adaptation Evolution Strategy (CMA-ES)[132]is a more sophisticated algorithm, designed to address complex non-linear, non-convex black-box optimization challenges.

Additionally, the Nelder-Mead Method [229], which uses a simplex-based approach, is suited for multidimensional unconstrained optimization. Lastly, Random Search, while straightforward, can be surprisingly effective. This method relies on randomly sampling the solution space, providing a baseline comparison for more complex algorithms. Each of these algorithms offers distinct advantages and is suited for specific types of optimization problems.

The choice of a specific algorithm depends on factors like problem dimensionality, noise level, and available computational resources. These algorithms have found applications in various fields, including machine learning, operations research, and engineering.

In a dissertation on black-box optimization, a comprehensive and critical approach is crucial. An effective review should cover the definition and importance of black-box optimization, classify the types of problems it addresses, and provide an overview of key methods and algorithms, along with their strengths, weaknesses, and suitability for different problem types. It should also discuss common challenges like dimensionality and local minima, incorporate recent research findings, and provide case studies and practical implementations.

Emerging trends and interdisciplinary applications, as well as a critical evaluation of the current state of research, are also essential components of such a dissertation. Concluding thoughts on the potential evolution of black-box optimization and its future impact are equally important.

One research line in black-box optimization contrasts with gradient-based and second-order (Hessian) methods. The advantage of black-box optimization lies in its independence from the function's gradient, which is beneficial when the gradient is hard to obtain or computationally expensive. Instead, function values can be directly used to optimize parameters, making it a versatile tool in scenarios where traditional methods are impractical.

Black box optimization encompasses a wide range of techniques, each designed to tackle optimization problems where the internal workings of the function are not fully known. Notable among these are genetic algorithms, Bayesian optimization often used in hyperparameter optimization, and probability-based methods. In probability-based optimization, a common strategy is to reparameterize the problem so that, although the function itself may not be differentiable, it becomes differentiable with respect to the parameters of the probability distribution after reparameterization.

Some popular methods in this category include the Cross-Entropy Method (CEM) and Covariance Matrix Adaptation Evolution Strategy (CMA-ES), which are forms of iterative stochastic optimization. Another significant method is Monte Carlo Tree Search (MCTS)[172], widely applied in various domains such as gaming and other machine learning applications.

## Monte Carlo Tree Search (MCTS)

MCTS operates on four fundamental concepts: selection, expansion, simulation, and backpropagation. In the selection phase, beginning at the root node, MCTS selects child nodes based on a policy that balances exploration (trying new moves) and exploitation (choosing known effective moves), often utilizing the Upper Confidence Bound applied to Trees (UCT) strategy. During expansion, if a leaf node has unexplored moves, the tree expands by adding child nodes representing these moves. In the simulation phase, Monte Carlo simulations are run from these new nodes, estimating the potential success of moves from that node. Finally, the backpropagation phase involves updating node values based on simulation outcomes, thereby enhancing the accuracy of move estimations.

The UCT score is integral to decision-making in tree-based algorithms. It is mathematically expressed as:

$$\text{Score}(s_t) = \frac{Q(s_t)}{N(s_t)} + 2C\sqrt{\frac{2\ln N(s_{t-1})}{N(s_t)}}$$

In this formula, $Q(s_t)$ represents the value of state $s_t$, and $N(s_t)$ is the number of visits to state $s_t$. The term $2C\sqrt{\frac{2\ln N(s_{t-1})}{N(s_t)}}$ introduces an exploration factor to balance exploration and exploitation. Here, $N(\cdot)$ indicates the visit count function, playing a crucial role in the algorithm's decision-making process.

## Cross Entropy Method (CEM)

The Cross Entropy Method (CEM) [83] is an optimization technique extensively used in artificial intelligence, machine learning, operations research, and other fields requiring solutions in complex spaces. This method is adept at solving problems where the objective is to find the best parameters to optimize a specific function. Here's a breakdown of how CEM works:

The Cross-Entropy Method (CEM) is a structured approach to optimization and is composed of several sequential steps. It begins with the initialization of a random probability distribution $D$ over the potential solution space, often a simple distribution such as a Gaussian with predefined mean and standard deviation.

Following initialization, CEM generates a set of samples from this distribution, with each sample representing a possible solution to the optimization problem. These samples are then evaluated using the objective function to determine their effectiveness.

Based on the results of this evaluation, CEM selects the top-performing samples. The selection criteria could be based on a fixed number of top performers or a percentile cutoff.

After selecting the best samples, the algorithm updates the parameters of the distribution $D$, such as recalculating the mean and standard deviation. This update shifts the focus towards areas of the search space that have shown better performance.

Finally, this process of generating, evaluating, selecting, and updating samples is repeated iteratively. This cycle continues until the algorithm meets predefined convergence criteria, indicating that an optimal or sufficiently good solution has been found.

In CEM, samples $A_1, A_2, \ldots, A_N$ are initially drawn from a distribution $p(A)$. Each sample $A_i$ is evaluated by a performance measure $J(A_i)$. The top-performing samples, known as 'elites' $A_{i_1}, \ldots, A_{i_M}$, are chosen based on their high $J$ values, with $M < N$.

The distribution $p(A)$ is then refined to these elites, enhancing the effectiveness of future samples.

CEM is a zero-order optimization algorithm, meaning it does not require gradient information. This contrasts with methods like Stochastic Gradient Descent (SGD), which rely on gradients. Additionally, higher-order methods such as the Gauss-Newton method and Kronecker-Factored Approximate Curvature (K-FAC) are vital, especially in optimizing the wavefunction within neural network ansatzes.

## Natural Evolution Strategies (NES)

Natural Evolution Strategies (NES)[333] represent a powerful optimization algorithm in various fields, particularly noted for its application in complex optimization scenarios. The fundamental concept behind NES is to utilize search gradients for updating the parameters of a search distribution. The distribution can vary, but commonly used ones include Gaussian mixture models and the Cauchy distribution, chosen for their easily computable derivatives of log-density.

In NES, let $\theta$ denote the parameters of density $\pi_\theta(z)$ and $f(z)$ represent the fitness function for samples $z$. The expected fitness under the search distribution is given by

$$J(\theta) = \mathbb{E}_\pi[f(z)] = \int f(z)\pi_\theta(z)dz.$$

The gradient of $J(\theta)$ can be computed using the 'log-likelihood trick', expressed as

$$\begin{aligned}
\nabla_\theta J(\theta) &= \nabla_\theta \int f(z)\pi_\theta(z)dz \\
&= \int f(z)\nabla_\theta \pi_\theta(z)dz \\
&= \int f(z)\nabla_\theta \left(\frac{\pi_\theta(z)}{\pi_\theta(z)}\right) dz \\
&= \int f(z)\nabla_\theta \log \pi_\theta(z)\pi_\theta(z)dz \\
&= \mathbb{E}_{\pi_\theta}[f(z)\nabla_\theta \log \pi_\theta(z)].
\end{aligned}$$

The search gradient can be estimated from samples $z_1 \dots z_\lambda$ (sampled according to $\pi_\theta$) as

$$\nabla_\theta J(\theta) \approx \frac{1}{\lambda} \sum_{k=1}^{\lambda} f(z_k) \nabla_\theta \log \pi_\theta(z_k),$$

and used in a gradient ascent approach to maximize the expected fitness function.

A key advantage of NES is that it does not require the fitness function $f$ to be differentiable. It works effectively even if $f$ is non-differentiable, as long as the search distribution's derivative is calculable. This approach is termed a probability-based algorithm, focusing on the function's expectation under a probability distribution rather than the original function itself.

One NES variant incorporates the concept of the natural gradient, introduced by Amari (1998) [7] and further developed by Amari and Douglas (1998) [6]. The natural gradient modifies the parameter update to be independent of the distribution's parameterization, using KL divergence as a measure of distance. This is achieved through a constrained optimization problem, leveraging the Fisher information matrix and techniques like Tikhonov regularization [307] or conjugate gradient methods for practical implementation.

$$\max_{\delta\theta} J(\theta + \delta\theta) \approx J(\theta) + \delta\theta^T \nabla_\theta J, \quad \text{s.t. } D(\theta + \delta\theta || \theta) \leq \epsilon, \tag{1.5.1}$$

When $\delta\theta \to 0$, we have the estimation $D(\theta + \delta\theta)||\theta) = \frac{1}{2}\delta\theta^T F(\theta)\delta\theta$,

where

$$\begin{aligned} F &= \int \pi_\theta(z) \nabla_\theta \log \pi_\theta(z) \nabla_\theta \log \pi_\theta(z)^T dz \\ &= \mathbb{E}\left[\nabla_\theta \log \pi_\theta(z) \nabla_\theta \log \pi_\theta(z)^T\right] \end{aligned}$$

We can get the the necessary condition using a Lagrangian multiplier (Peters, 2007)

$$F\delta\theta = \beta \nabla_\theta J$$

If F is invertible, the natural gradient amounts to $\widetilde{\nabla}_\theta J = F^{-1} \nabla_\theta J$. Then, we can use this adjusted gradient to update the parameters.

Note that the Fisher information matrix estimates may not always be invertible. To address this issue, techniques such as Tikhonov regularization can be employed, which

involves adding $\epsilon I$ to the matrix before inversion. Alternatively, iterative methods like the conjugate gradient can be used to compute the product of the inverse matrix with a vector without directly calculating the inverse.

## 1.6 AI for Science: Bridging Deep Learning and Quantum Physics

The intersection of deep learning and quantum physics in recent years has led to significant breakthroughs, marking a new era in scientific research known as "AI for Science." This convergence is exemplified in several groundbreaking studies.

A notable instance is DeepMind's development of FermiNet[246], an innovative neural network architecture specifically crafted for modeling quantum states in large electron systems. Their influential paper, "FermiNet: Quantum Physics and Chemistry from First Principles," demonstrates the power of deep learning in unraveling the intricate equations of quantum mechanics for practical applications. This work not only showcases deep learning's computational abilities but also its potential to transform our understanding of quantum systems.

Another remarkable contribution comes from Dian Wu, Lei Wang, and Pan Zhang with their paper "Solving Statistical Mechanics Using Variational Autoregressive Networks," [338] published in Physical Review Letters. This study introduces a unique method that utilizes autoregressive neural networks to solve problems in statistical mechanics. The approach surpasses traditional variational mean-field methods in various aspects, including the calculation of variational free energy, estimation of critical physical parameters like entropy, and generation of independent samples. Intriguingly, the paper integrates policy gradient techniques commonly used in reinforcement learning, highlighting the synergistic fusion of deep learning techniques with physics.

These pioneering works represent a growing trend where machine learning methodologies are increasingly applied to tackle intricate problems in physics. This blend of computational power and theoretical insight is significantly broadening the scope of scientific computation, opening up new possibilities for exploration and discovery in the realms of quantum physics and beyond.

## 1.7 Quantum circuit

Quantum circuits play a pivotal role in quantum computing, providing a mathematical and operational framework for quantum computation. A fundamental concept in

quantum circuits is the wavefunction, denoted $|\psi\rangle$, which resides in the complex Hilbert space $\mathbb{C}^N \approx (\mathbb{C}^2)^{\otimes n}$, where $N = 2^n$ and $n$ is the number of qubits. A key requirement for these wavefunctions is the normalization condition, ensuring that $\langle\psi|\psi\rangle = \sum_{j=0}^{N-1} |\psi_j|^2 = 1$.

In the realm of quantum gates, these are typically represented by unitary matrices $U \in \mathbb{C}^{N\times N}$. Quantum algorithms are essentially a series of large matrix-vector multiplications of the form $U_k \cdots U_1|\psi\rangle$. Measurement of some qubits is then performed, and the process is repeated $M$ times to produce classical output.

The basic language of quantum circuits includes the state vectors for one qubit in the computational basis, represented as:

$$|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \quad |1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

The Pauli matrices, $X$, $Y$, and $Z$, are also central to quantum computing:

$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}, \quad Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

These matrices serve as one-qubit gates (see Figure 1.5), with the $X$ gate acting as a bit-flip gate and the $Z$ gate changing the sign (phase) of $|1\rangle$.

$$|0\rangle \text{ —}\boxed{X}\text{— } |1\rangle \qquad |1\rangle \text{ —}\boxed{Z}\text{— } -|1\rangle$$

Figure 1.5: The illustration of the action of one-qubit quantum gates. On the left, the $X$ gate (bit-flip gate) is applied to the qubit initially in state $|0\rangle$, resulting in the state $|1\rangle$. On the right, the $Z$ gate (phase-flip gate) is applied to the qubit initially in state $|1\rangle$, leading to the state $-|1\rangle$. These gates exemplify fundamental quantum operations used in quantum computing for manipulating the state of qubits.

Another crucial gate is the Hadamard gate, represented by:

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

The Hadamard gate (see Figure 1.6) transforms the computational basis into the $|+\rangle$ and $|-\rangle$ basis.

$$|0\rangle \; —\boxed{H}— \; |+\rangle$$

Figure 1.6: The action of the Hadamard (H) gate on a quantum bit. When applied to a qubit initially in the $|0\rangle$ state, the Hadamard gate transforms it into the superposition state $|+\rangle$, which is an equal mix of the $|0\rangle$ and $|1\rangle$ states.

The tensor product is also a fundamental concept in quantum circuits, allowing for the construction of computational bases for multi-qubit systems. For instance, the basis states for a two-qubit system can be represented as:

$$|00\rangle = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \quad |01\rangle = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix}, \quad |10\rangle = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix}, \quad |11\rangle = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}$$

The CNOT gate (see Figure 1.7), an essential quantum gate, operates on two qubits and is represented by:

$$\text{CNOT} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

These gates serve as building blocks for constructing complex quantum circuits. The measurement operator in quantum circuits, crucial for collapsing a qubit's state to one of its basis states, is probabilistic, depending on the amplitudes of the qubit's state vector in the basis states. For a qubit in the state $\psi = \alpha|0\rangle + \beta|1\rangle$, measurement results in the qubit collapsing to $|0\rangle$ with probability $|\alpha|^2$ and to $|1\rangle$ with probability $|\beta|^2$. This process results in a loss of superposition information, mapping the qubit state to a classical bit.

In summary, quantum circuits (Figure 1.8 illustrates a typical quantum circuit), comprising gates and measurement operators, provide a foundation for quantum computing, allowing for the manipulation and measurement of qubits to perform complex computational tasks.

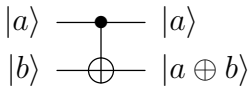



Figure 1.7: The demonstration of the operation of the Controlled NOT (CNOT) gate, a fundamental two-qubit gate in quantum computing. The CNOT gate flips the state of the second qubit (target) if the first qubit (control) is in the $|1\rangle$ state. In this illustration, the control qubit $|a\rangle$ remains unchanged, while the target qubit $|b\rangle$ is flipped to $|a \oplus b\rangle$, where $\oplus$ denotes the XOR operation.

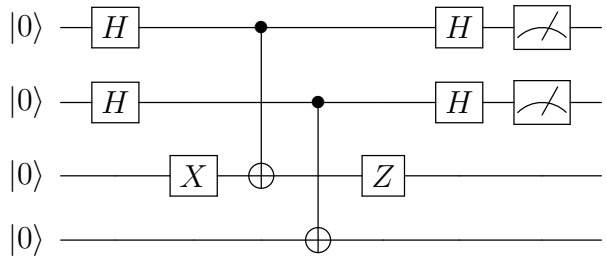



Figure 1.8: One typical quantum circuit, which is a fundamental construct in quantum computing. It consists of a series of quantum gates and measurement operators acting on qubits initially in the $|0\rangle$ state. The circuit begins with Hadamard (H) gates to create superposition states, followed by a combination of controlled NOT (CNOT) and Pauli-X, Pauli-Z gates to perform entanglement and phase manipulations. Finally, the circuit concludes with measurement operators, translating the qubit states into classical information. Such circuits enable complex computational tasks through the manipulation and measurement of qubits, showcasing the unique capabilities of quantum computing.

## 1.8 Mathematical Framework for Ground State Determination

The quintessential goal of variational quantum algorithms and other quantum control strategies is the precise determination of a system's ground state. This process is fundamentally about seeking the lowest eigenvalues ($\lambda_{\min}$) and their corresponding eigenvectors ($|\psi_{\min}\rangle$) of a quantum system's Hamiltonian ($H$). Mathematically, this is represented as:

$$H\,|\psi_{\min}\rangle = \lambda_{\min}\,|\psi_{\min}\rangle$$

In this pursuit, physicists rely on a repertoire of established models to elucidate various quantum phenomena. The transverse Ising model, represented as:

$$H_{\text{Ising}} = -J \sum_{\langle i,j \rangle} \sigma_i^z \sigma_j^z - h \sum_i \sigma_i^x$$

illuminates spin dynamics under an external magnetic field, where $\sigma_i^z$ and $\sigma_i^x$ are the Pauli matrices. The Heisenberg model, with its focus on exchange interactions, is encapsulated by:

$$H_{\text{Heisenberg}} = J \sum_{\langle i,j \rangle} (\sigma_i^x \sigma_j^x + \sigma_i^y \sigma_j^y + \sigma_i^z \sigma_j^z)$$

Lastly, the Hubbard model, pivotal in understanding electron correlations in a lattice, is expressed as:

$$H_{\text{Hubbard}} = -t \sum_{\langle i,j \rangle, \sigma} (c_{i,\sigma}^\dagger c_{j,\sigma} + \text{h.c.}) + U \sum_i n_{i,\uparrow} n_{i,\downarrow}$$

These models provide foundational structures for comprehending and manipulating quantum behaviors, pivotal in advancing quantum computing and simulation.

## 1.9 Quantum Approximate Optimization Algorithm (QAOA)

The Quantum Approximate Optimization Algorithm (QAOA) [102] is a promising quantum algorithm designed for solving combinatorial optimization problems within the realm of quantum computing. It represents a hybrid quantum-classical approach, utilizing both quantum and classical resources to find solutions more efficiently than classical algorithms alone. At its core, QAOA operates by encoding the optimization problem into a cost Hamiltonian, whose ground state corresponds to the optimal solution. The algorithm then applies a series of quantum gates, structured in a sequence of unitary operators, to prepare a superposition of states. These gates are parameterized, with their parameters optimized through a classical outer loop, typically using classical optimization techniques. This iterative process aims to approximate the ground state of the Hamiltonian, thus moving towards the optimal solution. QAOA stands out for its potential scalability and is particularly suited for near-term quantum computers, known as Noisy Intermediate-Scale Quantum (NISQ) devices, making it a focal point in current quantum computing research.

The Quantum Approximate Optimization Algorithm (QAOA) is underpinned by several fundamental mathematical concepts that are crucial for its understanding and implementation. At the heart of QAOA is the Cost Hamiltonian $H_c$, which encodes the optimization problem for a system with $n$ bits. This Hamiltonian acts as a Hermitian operator on a Hilbert space of dimension $2^n$. When applied to a computational basis string $z$, it results in $H_c|z\rangle = C(z)|z\rangle$, where $C(z)$ represents the cost function evaluated at $z$.

Complementing the Cost Hamiltonian is the Mixer Hamiltonian $H_m$, typically a transverse field Hamiltonian given by $H_m = -\sum_{i=1}^{n} X_i$, with $X_i$ being the Pauli-X operator on the $i$-th qubit. This Hamiltonian facilitates the creation of superpositions and enables quantum interference. The starting point of the algorithm is the preparation of the quantum state in an equal superposition of all possible states, represented by $|\psi_0\rangle = \frac{1}{\sqrt{2^n}} \sum_z |z\rangle$. This initial state is achieved by applying a Hadamard gate to each qubit in the $|0\rangle^{\otimes n}$ state.

QAOA utilizes a parameterized quantum circuit, defined by two sets of parameters: $\vec{\beta}$ and $\vec{\gamma}$, each corresponding to the circuit's depth $p$. The quantum state post-circuit application is given by $|\psi(\vec{\beta}, \vec{\gamma})\rangle = \prod_{k=1}^{p} e^{-i\beta_k H_m} e^{-i\gamma_k H_c} |\psi_0\rangle$.

The primary objective of QAOA is to minimize the expectation value of the Cost Hamiltonian, expressed as $\langle\psi(\vec{\beta}, \vec{\gamma})|H_c|\psi(\vec{\beta}, \vec{\gamma})\rangle$. This minimization is achieved by optimally selecting the parameters $\vec{\beta}$ and $\vec{\gamma}$ through classical optimization techniques. This iterative process necessitates repeated preparation and measurement of the quantum state.

The culmination of QAOA is the final measurement of the quantum state, which yields a bit string. The optimization of parameters $\vec{\beta}$ and $\vec{\gamma}$ enhances the probability of measuring a string that minimizes the cost function, marking the algorithm's success in finding optimal or near-optimal solutions to the given problem.

In summary, QAOA iteratively applies quantum evolution under two Hamiltonians, the cost Hamiltonian $H_c$ and the mixer Hamiltonian $H_m$, with the aim of preparing a quantum state that minimizes the expectation value of $H_c$, and hence finds an optimal or near-optimal solution to the combinatorial optimization problem.

In the field of quantum computing, the Quantum Approximate Optimization Algorithm (QAOA) has been the subject of numerous influential research papers that have collectively deepened our understanding and furthered its development. [315, 52, 341, 103, 362, 121]

## 1.10 Variational Quantum Eigensolver (VQE)

The Variational Quantum Eigensolver (VQE) is a pivotal algorithm in quantum computing, especially within the field of quantum chemistry. It embodies a hybrid approach, merging quantum and classical computing techniques, to determine the ground state energy of molecular Hamiltonians ($\hat{H}$). VQE's core strength stems from its utilization of the variational principle. It optimizes a quantum circuit with parameters $\vec{\theta}$ to minimize the energy expectation value $\langle\psi(\vec{\theta})|\hat{H}|\psi(\vec{\theta})\rangle$. This method enables the algorithm to approximate the system's ground state. VQE is especially advantageous for near-term quantum computers, offering resilience to quantum noise and the challenges of limited qubit coherence.

The body of literature in this area highlights the importance of Variational Quantum Algorithms (VQAs), including VQE, as a forefront strategy for addressing complex quantum systems and large-scale computational problems that are beyond the reach of classical computers. These algorithms are increasingly applied in diverse contexts, such as determining the ground states of molecules, simulating the dynamics of quantum systems, and solving linear equations. What sets VQAs apart is their hybrid structure, which involves a parameterized cost function evaluated by a quantum computer and subsequently optimized by classical computational techniques. Despite their potential, VQAs face challenges in terms of trainability, accuracy, and efficiency. As a result, there is ongoing research dedicated to enhancing their performance and applicability in various quantum computing tasks.

Variational quantum circuits (VQCs) and parametrized quantum circuits (PQCs) represent a powerful class of quantum circuits characterized by their remarkable adaptability and flexibility in performing quantum computations. Unlike traditional fixed-operation circuits, VQCs and PQCs incorporate adjustable parameters that can be tuned or optimized to achieve a specific computational goal. This adaptability makes them particularly valuable for tasks like quantum simulation, optimization problems, and quantum machine learning applications.

Structure and Optimization of VQCs: A typical VQC consists of a sequence of quantum gates, some of which have adjustable parameters. These parameters are varied in a controlled manner during the computation, often through classical optimization algorithms, to find the optimal setting that yields the desired outcome. This iterative process of parameter adjustment and optimization closely resembles the training process of neural networks in classical machine learning.

VQCs for Near-Term Quantum Computing: Given the current state of near-term quantum computers, which are susceptible to errors and have limited qubit coherence

times, VQCs have emerged as a promising approach for harnessing their potential. Their inherent ability to adapt and optimize allows them to compensate for these limitations, paving the way for practical quantum computing applications.

While various VQC ansätze exist, ADAPT-VQE[302, 211, 11, 22, 12] has emerged as a particularly powerful and versatile algorithm. Several key papers contribute to its development and understanding:

## 1.11 Variational Quantum Algorithms

The field of quantum technology has been notably enriched by the advent of variational quantum algorithms. These fall within the domain of Noisy Intermediate Scale Quantum (NISQ) computing, a realm where hybrid quantum algorithms, utilizing both classical and quantum devices, hold a significant place. Within these hybrid quantum algorithms, Variational Quantum Circuits (VQCs) or Parametrized Quantum Circuits (PQCs) are commonly encountered. The variational parameters within these quantum circuits are adeptly adjusted to yield desired quantum states.

The unique methods used for constructing these quantum circuits are termed as ansatz. Among the myriad of ansatz available, the Quantum Alternating Optimization Ansatz (QAOA) stands out as one of the simplest. Typically, it forms the quantum circuit by alternating two kinds of unitary Hamiltonian generators, expressed as $e^{-i\tau H}$), where $\tau$ represents the adjustable parameters and $H$ denotes the underlying Hamiltonian, signifying the rotation in the representation.

In addition to QAOA, other ansatze like ADEPT-QAOA[302, 211, 11], have been developed for specific applications. These alternative ansatze primarily aim to minimize the usage of CNOT gates, a crucial factor in the efficiency and feasibility of quantum computations on NISQ devices.

In variational quantum algorithms, the optimizable parameters are defined within parameterized quantum circuits (PQCs) [27, 240, 286]. A PQC is a sequence of unitary operators represented by parameterized quantum gates that can be readily implemented on a quantum computer. Assuming we are working in an $n$-qubit Hilbert space, a parameterized quantum circuit can be expressed as follows:

$$U(\boldsymbol{\theta}) = \prod_{j=1}^{J} U_j(\boldsymbol{\theta}_j) W_j. \tag{1.11.1}$$

Here, $\boldsymbol{\theta} = \{\boldsymbol{\theta}_j\}_{j=1}^{J}$ are the parameters that we need to optimize, $U_j(\boldsymbol{\theta}_j) \in \mathbb{C}^{2^n \times 2^n}$ are the parameterized unitary operators, and $W_j \in \mathbb{C}^{2^n \times 2^n}$ are fixed unitary operators.

For instance, a simple example of a PQC consisting only of one-qubit Pauli rotation operators takes the form

$$U_j(\boldsymbol{\theta}_j) = \bigotimes_{m=1}^{M} e^{-i\theta_{j,k_{j,m}}\sigma_{j,k_{j,m}}},$$

where $\sigma_{j,k_{j,m}} \in \mathbb{C}^{2\times 2}$ is a single-qubit Pauli matrix that acts on $k_{j,m}$-th qubit, $\theta_{j,k_{j,m}}$ represents one of the parameters in $\boldsymbol{\theta}$, and $W_j$'s can be used to represent quantum gates that do not require parameterization, such as the controlled-NOT (CNOT) gate.

Let $d$ be the dimension of the parameters, and we write $\boldsymbol{\theta} = (\theta_1, \theta_2, \cdots, \theta_d)$. We then optimize the parameter $\boldsymbol{\theta}$ by minimizing a properly chosen cost function $f(\boldsymbol{\theta})$. As an example, the variation quantum eigensolvers (VQE) finds the smallest eigenvalue (ground-state energy) and its corresponding eigenvector (ground state) of a given Hamiltonian matrix $H$ by minimizing the energy of the state:

$$\theta^* = \text{argmin}_{\boldsymbol{\theta}} f(\boldsymbol{\theta}) = \text{argmin}_{\boldsymbol{\theta}} \langle U(\boldsymbol{\theta})\psi_0 | H | U(\boldsymbol{\theta})\psi_0 \rangle . \tag{1.11.2}$$

Here, $|\psi_0\rangle \in \mathbb{C}^{2^n}$ is a predetermined initial state that can be easily prepared on a quantum computer. For each given $\boldsymbol{\theta}$, $U(\boldsymbol{\theta})$ is implemented on a quantum computer to evolve $|\psi_0\rangle$, and the corresponding energy $f(\boldsymbol{\theta})$ and its gradient $\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta})$ can be consequently obtained with measurements. By solving the optimization problem equation 6.1.2, the minimum value gives an approximation to the smallest eigenvalue of $H$, while $U(\boldsymbol{\theta}^*)|\psi_0\rangle$ approximates the corresponding eigenvector.

Note that an alternative objective in this context is fidelity, which assesses the overlap between two wave functions. It's important to mention that while direct access to the target state is not assumed in this case, it is presumed that there is an oracle capable of accurately determining the fidelity values. The optimization problem can be formulated as follows:

$$\theta^* = \text{argmin}_{\boldsymbol{\theta}} \tilde{f}(\boldsymbol{\theta}) = \text{argmin}_{\boldsymbol{\theta}} \| \langle U(\boldsymbol{\theta})\psi_0 | \psi_t \rangle \|^2. \tag{1.11.3}$$

In this equation, we are focusing on the ground state energy of the projection operator $|\psi_0\rangle\langle\psi_0|$.

In most instances, we utilize the ground state energy, as it more closely aligns with the measurements observed in experiments.

## 1.12 Counter-Diabatic Driving in Variational Quantum Algorithms

Recent advancements in variational quantum algorithms have seen the innovative incorporation of counter-diabatic (CD) driving techniques. A notable instance of this is the quantum alternating operator ansatz (QAOA), which has been extended into a generalized form known as CD-QAOA. This approach, inspired by counter-diabatic driving procedures, is particularly tailored for quantum many-body systems. It leverages reinforcement learning (RL) to optimize the algorithm, proving effective in preparing the ground state of quantum-chaotic many-body spin chains by minimizing energy. The CD-QAOA uses terms from the adiabatic gauge potential as generators of additional control unitaries, enabling fast, high-fidelity control in non-adiabatic regimes. This method maintains the intrinsic continuous control of QAOA, such as time duration, while also treating the order of multiple unitaries in the control sequence as a discrete optimization challenge. By integrating a policy gradient algorithm with an autoregressive deep learning architecture to understand causality, the RL agent is trained to create optimal sequences of unitaries. Remarkably, this algorithm operates without direct access to the quantum state, and evidence suggests that protocols learned on smaller systems may be scalable to larger systems. This research highlights a significant stride towards more efficient and robust quantum control mechanisms, potentially revolutionizing the field of quantum computing [348].

Counter-Diabatic Driving (CDD), or shortcut to adiabaticity [122, 269], is a quantum control technique used to accelerate adiabatic processes in quantum systems, and it can indeed be described mathematically. The central idea of CDD is to suppress non-adiabatic transitions during the evolution of a quantum system. Mathematically, this is achieved by adding an auxiliary Hamiltonian $H_{CD}$ to the original Hamiltonian $H_0(t)$ of the system.

The total Hamiltonian $H(t)$ in Counter-Diabatic Driving is given by:

$$H(t) = H_0(t) + H_{CD}(t)$$

The auxiliary Hamiltonian $H_{CD}(t)$ is specifically designed to cancel the non-adiabatic transitions induced by $H_0(t)$. It is often expressed in terms of the adiabatic gauge potential $A(t)$, which depends on the instantaneous eigenstates of $H_0(t)$. The form of $H_{CD}(t)$ is typically:

$$H_{CD}(t) = i\hbar \sum_n \left( |\partial_t n\rangle \langle n| - \langle n|\partial_t n\rangle |n\rangle \langle n| \right)$$

Here, $|n\rangle$ are the instantaneous eigenstates of $H_0(t)$, and $|\partial_t n\rangle$ represents their time derivatives.

This additional term in the Hamiltonian allows the system to follow the adiabatic pathway more closely, even when the evolution is faster than the adiabatic timescale. As a result, Counter-Diabatic Driving finds applications in quantum computing and quantum information processing, where it helps in implementing fast and accurate quantum operations. Its integration into variational algorithms demonstrates significant potential in enhancing the capabilities of these algorithms[348].

## 1.13 Barren Plateaus in Quantum Algorithms

The phenomenon of barren plateaus [212, 61] presents a significant challenge in the field of quantum computing, particularly in the context of variational quantum algorithms. This term refers to the occurrence of vast regions in the optimization landscape of a quantum algorithm where the gradient is close to zero. As a result, it becomes increasingly difficult for gradient-based optimization methods to find the global minimum.

Barren plateaus are attributed to several factors, including high circuit depth, random initialization of parameters, and entanglement. They are especially problematic in large quantum systems, where the likelihood of encountering flat regions in the optimization landscape increases. This phenomenon can hinder the training process of quantum neural networks and variational quantum eigensolvers, impeding their practical applications.

Addressing barren plateaus involves strategies like careful initialization of parameters, employing problem-inspired ansatzes, and using local rather than global cost functions. Research in this area is crucial for the advancement of quantum algorithms and their scalability to solve complex problems efficiently.

In the context of quantum computing, especially when working with a system of $n$ qubits, understanding the challenges posed by barren plateaus is crucial. These plateaus are regions where the gradient of a cost function $C(\theta)$, which is dependent on the output state $|\psi(\theta)\rangle$, approaches zero as the number of qubits $n$ increases. This phenomenon is largely attributed to the concentration of measure in high-dimensional spaces. Additionally, deep circuits with high entanglement can lead to more uniform distributions across the quantum state space, which further exacerbates the issue of barren plateaus. The degree of entanglement, often measured by entanglement entropy, significantly influences the flatness of the optimization landscape.

A key aspect to consider in this scenario is the Gradient Variance. The variance of the gradient components, denoted as Var $\left(\frac{\partial C}{\partial \theta_i}\right)$, becomes a critical factor. In barren plateaus, this variance tends to be exponentially small relative to the number of qubits $n$, making it challenging to navigate the optimization landscape.

To address the challenge of barren plateaus in quantum computing, where gradients tend to vanish making optimization difficult, several strategies are employed. One effective approach is the use of shallow circuits. Reducing the depth of quantum circuits can help circumvent extreme entanglement, which is often a key factor contributing to barren plateaus. This approach is beneficial as it can simplify the circuit while maintaining functionality.

Another method involves careful parameter initialization. Starting the optimization with larger gradients can avert early stagnation in the optimization process. This allows for more effective exploration and navigation through the optimization landscape, enhancing the likelihood of finding optimal solutions.

Additionally, employing local cost functions has proven useful, especially in large quantum systems. By focusing optimization efforts on a subset of qubits rather than the entire system, the impact of the concentration of measure phenomenon is reduced. This localized approach can significantly mitigate the challenges posed by barren plateaus, making the optimization process more manageable and efficient.

These strategies collectively enhance the feasibility and effectiveness of quantum computations, especially in scenarios where the complexity of quantum circuits and the scale of quantum systems pose significant challenges.

## 1.14 Outlook

Embarking on the journey of exploring variational quantum algorithms (VQCs), we find ourselves at the forefront of significant advancements in multiple key areas. A primary focus lies beyond traditional optimization methods, where classical algorithms falter in handling the complexities of noisy quantum systems. The pursuit of novel optimization techniques, rooted in quantum mechanics, such as stochastic methods that utilize noise for exploration, and hybrid algorithms combining classical and quantum approaches, is vital.

Equally crucial is the challenge of probing the excited states in quantum systems. While VQCs have predominantly concentrated on the ground state, accessing and understanding excited states is essential for a deeper grasp of quantum phenomena. This necessitates developing specialized ansätze and robust algorithms for identifying

these states.

Another pivotal aspect is moving beyond the reliance on initial overlap between the initial and target states in VQCs. To achieve this, we need a paradigm shift towards innovative initialization strategies and evolution protocols. These new approaches should efficiently traverse the quantum state landscape without depending on pre-existing similarities.

Furthermore, expanding our focus beyond qubit-based VQCs to include alternative quantum computing platforms, such as trapped ions and continuous-variable systems, can open new avenues. Developing VQCs for these varied architectures will enhance our computational reach and facilitate access to a broader spectrum of quantum algorithms.

By tackling these crucial areas, we stand to elevate VQCs to unprecedented heights, unlocking their transformative potential in a myriad of fields, from materials science and drug discovery to machine learning and optimization. The road ahead for variational quantum algorithms is not just promising but laden with potential for remarkable discoveries and groundbreaking applications.

# Chapter 2

# Policy Gradient based Quantum Approximate Optimization Algorithm

The quantum approximate optimization algorithm (QAOA), as a hybrid quantum/classical algorithm, has received much interest recently. QAOA can also be viewed as a variational ansatz for quantum control. However, its direct application to emergent quantum technology encounters additional physical constraints: (i) the states of the quantum system are not observable; (ii) obtaining the derivatives of the objective function can be computationally expensive or even inaccessible in experiments, and (iii) the values of the objective function may be sensitive to various sources of uncertainty, as is the case for noisy intermediate-scale quantum (NISQ) devices. Taking such constraints into account, we show that policy-gradient-based reinforcement learning (RL) algorithms are well suited for optimizing the variational parameters of QAOA in a noise-robust fashion, opening up the way for developing RL techniques for continuous quantum control. This is advantageous to help mitigate and monitor the potentially unknown sources of errors in modern quantum simulators. We analyze the performance of the algorithm for quantum state transfer problems in single- and multi-qubit systems, subject to various sources of noise such as error terms in the Hamiltonian, or quantum uncertainty in the measurement process. We show that, in noisy setups, it is capable of outperforming state-of-the-art existing optimization algorithms.

## 2.1 Introduction

Noisy intermediate scale quantum (NISQ) devices are becoming increasingly accessible. However, their performance can be severely restricted due to decoherence effects. This leads to noises in all components of the quantum computer, including initial state preparation, unitary evolution, and measurement/qubit readout. Thanks to the feasibility of being implemented and tested on near term devices, hybrid quantum-classical algorithms, and in particular quantum variational algorithms (QVA), have received significant amount of attention recently. Examples of QVA include the Variational Quantum Eigensolver [244], the Quantum Approximate Optimization Algorithm (QAOA) [102], Quantum Variational Autoencoders [264], etc. The common feature of these algorithms is that the final wavefunction can be prepared by applying a unitary evolution operator, parametrized using a relatively small number of parameters, to an initial wavefunction. The parameters can then be variationally optimized to maximize a given objective function, measured on the quantum device.

In this study, we mainly focus on the Quantum Approximate Optimization Algorithm (QAOA) [102], which is a particularly simple algorithm that alternates between two different unitary time evolution operators of the form $e^{-\mathrm{i}H_0 t}$, $e^{-\mathrm{i}H_1 t}$ ($t \in \mathbb{R}$). This is also dubbed as the quantum alternating operator ansatz [129]. Both share the same acronym QAOA. We use the term QAOA in a broader sense than that in the original paper by Farhi et al. The algorithm proposed here can be potentially used for a larger class of variational quantum circuits (VQC, which includes the variational quantum eigensolver, VQE as a special case). However, it would be practically difficult to assess the efficiency and robustness of the method for general VQCs. Therefore for concreteness, in this work we specifically confine our study to QAOA.

QAOA has been studied in the context of a number of discrete [102, 200, 128] and continuous [315] optimization problems. QAOA has also been demonstrated to be universal under certain circumstances [199, 200, 224], in the sense that any element in a unitary group can be well approximated by a properly parameterized QAOA. This is highly nontrivial and is a unique quantum feature, since QAOA only has access to unitary operators generated by two specific Hamiltonians $H_0, H_1$. However, the control energy landscape of QAOA is known to be highly complex [290, 236], and optimization in it can therefore be challenging. For a one-level system, the QAOA optimization landscape in a channel decoding problem can already be quite complex [210]. For random parameterized quantum circuits (RPQCs), the average value of the gradient of the objective function has been reported to be almost zero [212]. Such vanishing gradients in large plateaus pose challenge to optimization algorithms.

If the landscape has exponentially many local minima, there is exponentially small probability of reaching the global minimum [82].

QAOA can be naturally related to quantum control, and thus also to reinforcement learning problems. This inspires studies from various angles, such as the Krotov method [304], Pontryagin's maximum principle [345] and Bayesian optimization [270], sequential minimal optimization [227], tabular reinforcement learning methods [65, 44], functional-approximation-based (deep) Q-learning methods [48, 288, 8, 358], policy gradient methods [107, 19, 66, 237, 249, 325], differential programming [272] and methods inspired by the success of AlphaZero [77]. Most studies focus on the noise-free scenarios, applicable to fault-tolerant quantum devices. In order to mitigate the errors on near-term devices, robust optimization based on sequential convex programming (SCP) has been recently studied [176, 97], which assumes that both the source and the range of magnitude of the error are known, but its exact magnitude. In such a case, the authors have found that robust optimization can significantly improve the accuracy of the variational solution.

Nonetheless, techniques such as SCP require access to information of the first as well as second order derivatives of the objective function, which can themselves be noisy and difficult to obtain on quantum devices. The objective function should also be at least continuous with respect to the error, a requirement which is not satisfied in the case of quantum uncertainty in the final measurement process (e.g. in the form of a bit flip or a phase flip). It is thus naturally desirable to only use function evaluations to perform robust optimization, while keeping the result resilient to unknown and generic types of errors.

In this paper, we demonstrate that reinforcement learning (RL) may be used to tackle all challenges above in optimizing the parameters of QAOA, and more generally QVA. Instead of directly optimizing the variational parameters themselves, we may assign a *probability distribution* to the parameter set, and perform optimization with respect to the parameters of the probability distribution, denoted $\boldsymbol{\theta}$. The modified objective function (called the expected total reward function) can then be continuous with respect to $\boldsymbol{\theta}$, even if the original objective function is not. The optimization procedure only requires a (possibly large) number of function evaluations, but no information about the first or second order derivatives. We show that a simple policy gradient method only introduces a small number of additional parameters in the optimization procedure, and can be used to optimize the parameters in QAOA. Since each step of the optimization only involves a small batch of samples, the optimization procedure can also be resilient to various sources of noise.

This paper is organized as follows. Section 2.2 provides a brief introduction of QAOA,

its connection to quantum control, and the noise models. Section 2.3 introduces the policy gradient based QAOA (PG-QAOA), in the context of noise-free and noisy optimization. After introducing the test systems in Section 2.4, we present in Section 2.5 numerical results of PG-QAOA for single-qubit and multi-qubit examples under different noise models. Section 2.6 concludes and discusses the further work.

## 2.2 Preliminaries

### QAOA and Quantum Control

Consider the Hilbert space $\mathcal{H} = \mathbb{C}^{2^N}$, with $N$ the number of qubits in the quantum system. Starting from an initial quantum state $|\psi_i\rangle \in \mathcal{H}$, in QAOA we apply two alternating unitary evolution operators [102]:

$$|\psi\rangle = U(\{\alpha_i, \beta_i\}_{i=1}^p)\,|\psi_i\rangle = e^{-\mathrm{i}H_1\beta_p} e^{-\mathrm{i}H_0\alpha_p} \cdots e^{-\mathrm{i}H_1\beta_1} e^{-\mathrm{i}H_0\alpha_1}\,|\psi_i\rangle\,. \tag{2.2.1}$$

The unitary evolution is generated by the time-independent Hamiltonian operators $H_0$ and $H_1$, each applied for a duration $\alpha_i \geq 0$ and $\beta_i \geq 0$, respectively ($i = 1, 2, \cdots, p$); we refer to $p$ as the total depth. In QAOA, we have to adjust the parameters to optimize an objective function $F(|\psi\rangle) = F(\{\alpha_i, \beta_i\}_{i=1}^p)$, e.g. minimizing the energy [141] or maximizing the fidelity of being in some target state[1]. In the latter case, for a target wavefunction denoted by $|\psi_*\rangle$, the optimization problem becomes

$$\{\alpha_i^*, \beta_i^*\}_{i=1}^p = \underset{\{\alpha_i,\beta_i\}_{i=1}^p}{\arg\max}\, F(\{\alpha_i, \beta_i\}_{i=1}^p), \tag{2.2.2}$$

$$F(\{\alpha_i, \beta_i\}_{i=1}^p) = |\,\langle\psi_*|U(\{\alpha_i, \beta_i\}_{i=1}^p)|\psi_i\rangle|^2\,. \tag{2.2.3}$$

The problem of finding the optimal parameters in QAOA can be reinterpreted as the following bilinear quantum optimal control problem

$$\mathrm{i}\partial_t\,|\psi(t)\rangle = H(t)\,|\psi(t)\rangle\,, \quad |\psi(0)\rangle = |\psi_i\rangle\,, \tag{2.2.4}$$

where $H(t) = H_0 + u(t)(H_1 - H_0), u(t) \in \{0, 1\}$. In particular, when $u(t)$ is chosen to be the following piecewise constant function

$$u(t) = \begin{cases} 0, & t \in \big[\sum_{k=1}^{i-1}(\alpha_k + \beta_k), \sum_{k=1}^{i-1}(\alpha_k + \beta_k) + \alpha_i\big), \\ 1, & t \in \big[\sum_{k=1}^{i-1}(\alpha_k + \beta_k) + \alpha_i, \sum_{k=1}^{i}(\alpha_k + \beta_k)\big), \end{cases} \quad i = 1, \ldots, p, \tag{2.2.5}$$

[1]The latter problem is often referred to as a state transfer problem. This is mainly for simplicity and serves as a proof of principle for the effectiveness of the policy gradient based method.

we recover the QAOA wavefunction (6.4.1). This is a special type of quantum control problem called the bang-bang quantum control. For a protocol of the durations $\{\alpha_i, \beta_i\}_{i=1}^p$, the total duration is defined as

$$T\left(\{\alpha_i, \beta_i\}_{i=1}^p\right) = \sum_{i=1}^{p} \left(\alpha_i + \beta_i\right). \tag{2.2.6}$$

## Noisy Objective Functions

Practical QAOA calculations can be prone to noises. For instance, the Hamiltonian may take the form $H(\delta) = \bar{H} + \delta\tilde{H}$, where $\bar{H}$ is the Hamiltonian in the absence of noise, $\tilde{H}$ is the Hamiltonian modelling the noise source, with $\delta$ the magnitude of the noise. We assume that only the range/magnitude of $\delta$ is known *a priori* and is denoted by $\Delta$, and the precise value of $\delta$ is not known. This setup will be referred to as the *Hamiltonian noise*. The explicit form of the Hamiltonian noise will be discussed later in Section 2.4. This noisy optimization problem can be solved as a max-min problem:

$$\max_{\{\alpha_i,\beta_i\}_{i=1}^p} \quad \min_{\delta\in\Delta} F(\{\alpha_i, \beta_i\}_{i=1}^p, \delta), \tag{2.2.7}$$

where

$$F(\{\alpha_i, \beta_i\}_{i=1}^p, \delta) = |\left\langle\psi_i|U(\{\alpha_i, \beta_i\}_{i=1}^p, \delta)|\psi_*\right\rangle|^2 \tag{2.2.8}$$

is the fidelity for the given noise strength and control duration.

Noise may naturally also occur due to imperfect measurement operations. For instance, the final fidelity may only be measurable up to an additive Gaussian noise, i.e.

$$F_\sigma(\{\alpha_i, \beta_i\}_{i=1}^p) = \text{clip}(F(\{\alpha_i, \beta_i\}_{i=1}^p) + \epsilon, 0, 1), \tag{2.2.9}$$

where $\epsilon \sim \mathcal{N}(0, \sigma^2)$. Here, the clip[2] function guarantees the noisy fidelity is still bounded between 0 and 1. This will be referred to as *Gaussian noise*. It mimics the case when experimentalists do lots of measurements and average the result in the end to get an estimate of the observable. By the central limit theorem, as the sample size becomes sufficiently large, the statistics of the measurement data points is well approximated by a Gaussian distribution. As a result, we use the Gaussian noise to describe the uncertainty leading to a noise in the reward signal.

[2]This is just one way to enforce the fidelity to be between 0 and 1. We admit that the clipping procedure is an artifact of the implementation; it is not necessary and does not constitute an essential feature of the PG-QAOA algorithm.

Furthermore, quantum measurements produce an intrinsic source of uncertainty due to the probabilistic nature of quantum mechanics. Assuming the target state $|\psi_*\rangle$ is an eigenstate of some measurable operator $O$ with eigenvalue $o_*$, i.e. $O|\psi_*\rangle = o_*|\psi_*\rangle$, a quantum measurement $\langle\psi| O |\psi\rangle$ produces the eigenvalue $o_*$ with probability $F(\{\alpha_i, \beta_i\}_{i=1}^p)$. Using this, we can define the following discrete cost function:

$$F_Q(\{\alpha_i, \beta_i\}_{i=1}^p) = \begin{cases} 1 & \text{with probability } F(\{\alpha_i, \beta_i\}_{i=1}^p) \\ 0 & \text{with probability } 1 - F(\{\alpha_i, \beta_i\}_{i=1}^p) \end{cases} \tag{2.2.10}$$

Assuming the same state $|\psi\rangle$ of the system is prepared anew in a series of experiments, a measurement in repeated experiments will produce a discrete set of ones and zeros, whose mean value converges to the true fidelity $F(\{\alpha_i, \beta_i\}_{i=1}^p)$ in the limit of large number of quantum measurements. This setting was considered in [44], and will be referred to as the *quantum measurement noise*. We mention in passing that, in systems with large Hilbert spaces, such as multi-qubit systems, it is in fact more appropriate to optimize the expectation value of some observable, instead of the fidelity.

## 2.3 Policy gradient based QAOA (PG-QAOA)

Being a variational ansatz, QAOA does not specify the optimization procedure to determine the variational parameters. In this paper, we demonstrate that policy gradient, which is a widely used algorithm in reinforcement learning, can be particularly robust to various sources of uncertainty in the physical system. In order to tackle the robust optimization of QAOA for general noise models, reinforcement learning algorithms provide a useful perspective.

We first reformulate the original problem as a probability-based optimization problem. The original optimization parameters are drawn from a probability distribution, described by some variational parameters $\boldsymbol{\theta}$. Optimization is then performed over the variational parameters. Such techniques are used in natural evolution strategies (NES) [332] and model-based optimization (MBO) [42]. It is also shown very recently by [359] that NES can be efficiently applied to solve combinatorial optimization problems in the quantum classical approach. If the solution of the optimization problem is unique, we should expect that the probability distribution of each parameter will converge to a Dirac-$\delta$ distribution. This probability-based approach also has the advantage of being resilient to perturbations and noise. As will be shown later, the width of the probability distribution after optimization can also be used to qualitatively monitor the magnitude of the unknown noise. A common example of a probability-based

optimization algorithm in reinforcement learning is the policy gradient algorithm, where the goal is to find the optimal policy $\pi_{\boldsymbol{\theta}}$ to perform a given task [334, 295]. An additional advantage of probability-based optimization is that it can be used to handle continuous and discrete variables in a unified fashion. Thus, the ideas we put forward below can be used to open up the way to applying RL techniques to continuous quantum control. In the context of QAOA, the durations can be treated as continuous variables without the error due to time discretization.

Let us begin by casting the QAOA control problem Eq. (2.2.6) in a reinforcement learning framework. We consider finite-horizon episodic learning task, with $p$ steps per episode, naturally defined by the discrete character of the QAOA ansatz Eq. (4.2.2).

The natural choice for the RL state space is the Hilbert space $\mathcal{H}$. However, there are a number of problems associated with this choice: (i) the wave function $|\psi\rangle$ is a mathematical concept, which cannot be directly measured in experiments (for instance, there is an arbitrary phase factor that cannot be directly measured). (ii) in quantum mechanics, every measurement would directly lead to the collapse of the wavefunction. (iii) in many-body quantum systems of $N$ particles, $\dim \mathcal{H} \sim \exp(N)$ is exponentially large which raises questions about the scalability of the algorithm to a large number of qubits. Indeed, reading out all entries of the quantum wavefunction requires full quantum tomography [309], which scales exponentially with the number of qubits $N$. This comes in stark contrast with recent applications of RL to certain optimal quantum control problems e.g. [237, 77], in which the quantum wavefunction for small Hilbert spaces is indeed accessible on a classical computer.

In our setting, since the dynamics is governed by the Schrodinger equation and initial state is also given, the quantum state at an intermediate time $|\psi(t)\rangle$ can be in principle determined from the sequence of actions taken at each time interval. Therefore, the sequence of all actions taken before a given episode step can be treated effectively as the RL state, and we work with this definition here. We mention in passing that this choice is not unique: in practice, reinforcement learning based methods often incorporate some form of embedding of the quantum state as their state. Notable examples include tabular Q-Learning [48], Q-Learning network [288, 8], LSTM based memory proximal policy optimization [19, 107].

At every step $j$ in the episode, our RL agent is required to choose two actions out of a continuous interval $[0, \infty)$ independently, representing the values of the durations $\alpha_j, \beta_j$. Hence, the action space is $\mathcal{A} = [0, \infty)$. Actions are selected using the parameterized policy $\pi_{\boldsymbol{\theta}}$. Since we use the fidelity as the objective function, the reward space is $\mathcal{R} = [0, 1]$.

In this work, we use the simplest ansatz, i.e. independent Gaussian distributions to parameterize the policy over the control durations $\{\alpha_i, \beta_i\}_{i=1}^p$ in QAOA. Since a Gaussian is uniquely determined by its mean $\mu$ and standard deviation (std) $\sigma$, we have a total of $2p$ independent variational parameters $\boldsymbol{\theta} = \{\mu_{\alpha_i}, \sigma_{\alpha_i}, \mu_{\beta_i}, \sigma_{\beta_i}\}_{i=1}^p$. The total number of parameters is $4p$ (in particular, it does not directly scale with the number of qubits $N$). The probability density of all the parameters $\pi_{\boldsymbol{\theta}}(\{\alpha_i, \beta_i\}_{i=1}^p)$ is the product of all the marginal distributions:

$$\pi_{\boldsymbol{\theta}}(\{\alpha_i, \beta_i\}_{i=1}^p) = \prod_{i=1}^{p} \pi(\alpha_i; \mu_{\alpha_i}, \sigma_{\alpha_i}) \cdot \pi(\beta_i; \mu_{\beta_i}, \sigma_{\beta_i}), \tag{2.3.1}$$

where $\pi(x; \mu, \sigma)$ is the probability density for the Gaussian distribution $\mathcal{N}(\mu, \sigma)$,

$$\pi(x; \mu, \sigma) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}. \tag{2.3.2}$$

Note that with such a choice, $x$ may become negative, which lies outside the action space $\mathcal{A}$. We can enforce the constraint using a truncated Gaussian distribution (after proper normalization) or a log-normal distribution. In practice we observe that with proper initialization, the positivity condition is observed to be automatically satisfied by the minimizer even with the simple choice in Eq. (2.3.2).

The QAOA objective function (4.2.2) for the probability-based ansatz (4.2.4) introduced above, now takes the form:

$$\{\mu^*_{\alpha_i}, \sigma^*_{\alpha_i}, \mu^*_{\beta_i}, \sigma^*_{\beta_i}\}_{i=1}^p = \underset{\{\mu_{\alpha_i}, \sigma_{\alpha_i}, \mu_{\beta_i}, \sigma_{\beta_i}\}_{i=1}^p}{\arg\max} \left( \underset{\substack{\alpha_i \sim \mathcal{N}(\mu_{\alpha_i}, \sigma_{\alpha_i}) \\ \beta_i \sim \mathcal{N}(\mu_{\beta_i}, \sigma_{\beta_i})}}{\mathbb{E}} [F(\{\alpha_i, \beta_i\}_{i=1}^p)] \right) = \arg\max_{\boldsymbol{\theta}} J(\boldsymbol{\theta}). \tag{2.3.3}$$

Here $J(\boldsymbol{\theta})$ is called the expected reward function. In this form, the objective function $J(\boldsymbol{\theta})$ can be optimized using the REINFORCE algorithm for policy gradient [334]:

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \underset{\substack{\alpha_i \sim \mathcal{N}(\mu_{\alpha_i}, \sigma_{\alpha_i}) \\ \beta_i \sim \mathcal{N}(\mu_{\beta_i}, \sigma_{\beta_i})}}{\mathbb{E}} \left[ \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(\{\alpha_i, \beta_i\}_{i=1}^p) \cdot F(\{\alpha_i, \beta_i\}_{i=1}^p) \right], \tag{2.3.4}$$

In particular, the gradient can be evaluated without information about the first order derivative of the objective function $F$. In practice, we use a Monte Carlo approximation to evaluate this gradient, as shown in Algorithm 1. In order to reduce the variance of the gradient, usually a baseline is subtracted from the fidelity [120], i.e.

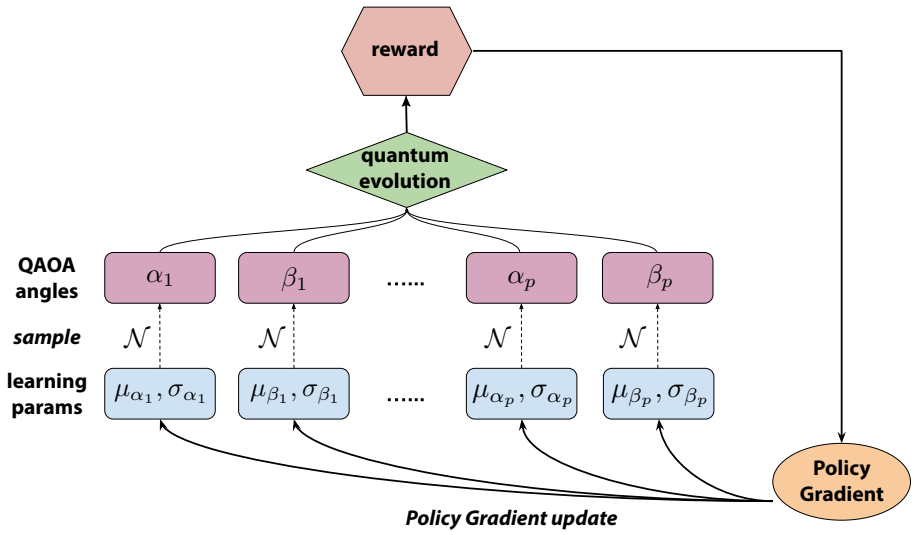



Figure 2.1: Schematic diagram for PG-QAOA. The algorithm samples a batch of QAOA time-durations (angles) from the current policy, aggregates the resulting fidelities/rewards from a quantum 'blackbox', and applies the policy gradient algorithm to update the learning parameters to improve the policy.

replacing $F(\{\alpha_i, \beta_i\}_{i=1}^p)$ with $F(\{\alpha_i, \beta_i\}_{i=1}^p) - \bar{F}$ in Eq. (2.3.4); it is easy to compute the average fidelity over the MC sample (i.e. the batch) and we use that as the baseline. The resulting algorithm will be referred to as the policy gradient based QAOA (PG-QAOA).

**Algorithm 1** Policy gradient based QAOA

**Input:** Initial guess for the mean and std $\mu^0_{\alpha_i}, \sigma^0_{\alpha_i}, \mu^0_{\beta_i}, \sigma^0_{\beta_i}, i = 1, 2, \cdots, p$; batch size $M$, learning rate $\tau_t$, total number of iterations $N_{\text{iter}}$.

1: Initialize the mean and std with the initial guess

$$(\mu_{\alpha_i}, \sigma_{\alpha_i}, \mu_{\beta_i}, \sigma_{\beta_i}) \leftarrow (\mu^0_{\alpha_i}, \sigma^0_{\alpha_i}, \mu^0_{\beta_i}, \sigma^0_{\beta_i}), i = 1, 2, \cdots, p.$$

2: **for** $t = 1, .., N_{\text{iter}}$ **do**

3: Sample batch $B$ of size $M$:

$$\alpha^j_i \sim \mathcal{N}(\mu_{\alpha_i}, \sigma_{\alpha_i}), \beta^j_i \sim \mathcal{N}(\mu_{\beta_i}, \sigma_{\beta_i}),\ i = 1, 2, \cdots, p,\ j = 1, 2, \cdots, M.$$

4: Compute the instantaneous fidelity and the averaged fidelity

$$F_j = \left| \langle \psi_* | U(\{\alpha^j_i, \beta^j_i\}_{i=1}^p) | \psi_i \rangle \right|^2, \quad \bar{F} = \frac{1}{M} \sum_{j=1}^{M} F_j.$$

5: Compute the policy gradient

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \frac{1}{M} \sum_{\{\alpha^j_i, \beta^j_i\}_{i=1}^p \in B} \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(\{\alpha^j_i, \beta^j_i\}_{i=1}^p) \cdot (F_j - \bar{F}).$$

6: Update weights $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \tau_t \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$.

7: **end for**

PG-QAOA can be naturally extended to the setting of robust optimization for the Hamiltonian noise. For the max-min problem, the policy gradient in Eq. (2.3.4) becomes

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \mathop{\mathbb{E}}_{\substack{\alpha_i \sim \mathcal{N}(\mu_{\alpha_i}, \sigma_{\alpha_i}) \\ \beta_i \sim \mathcal{N}(\mu_{\beta_i}, \sigma_{\beta_i})}} \left[ \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(\{\alpha_i, \beta_i\}_{i=1}^p) \cdot \min_{\delta} F(\{\alpha_i, \beta_i\}_{i=1}^p, \delta) \right]. \tag{2.3.5}$$

In practice, we sample independent random realizations $\delta_j$ from the noise region $\Delta$ at uniform, and use $\min_j F(\{\alpha_i, \beta_i\}_{i=1}^p, \delta_j)$ as an approximation in Eq. (2.3.5). When the fidelity itself is noisy such as the case of the Gaussian noise and the quantum noise, we simply use the measured fidelity in Eq. (2.2.9) and Eq. (2.2.10) in the policy gradient step of Eq. (2.3.5).

## 2.4 Quantum Qubit Models

We investigate the performance of PG-QAOA for a single-qubit system, and two different multi-qubit systems, defined as follows:

### Single qubit model

Consider a single-qubit system, whose QAOA dynamics is generated by the Hamiltonians

$$H_0 = -\frac{1}{2}\sigma^z + 2\sigma^x, \qquad H_1 = -\frac{1}{2}\sigma^z - 2\sigma^x, \tag{2.4.1}$$

with $\sigma^\alpha$ the Pauli matrices. The initial $|\psi_i\rangle$ and target $|\psi_*\rangle$ states are chosen to be the ground states of $H_i = -\frac{1}{2}\sigma^z + \sigma^x$ and $H_* = -\frac{1}{2}\sigma^z - \sigma^x$, respectively. This control problem was introduced and analyzed in the context of reinforcement learning in Ref. [48]: below the quantum speed limit (QSL), i.e. for total duration $T \leq T_{\rm QSL} \approx 2.41$, it is not possible to prepare the target state with unit fidelity; yet, in this regime there is a unique optimal solution which maximizes the fidelity of being in the target state, and its fidelity is less than 1. Above the QSL, $T > T_{\rm QSL}$, there exist multiple unit-fidelity solutions to this constrained optimization problem.

### Multi-qubit Models

To compare the performance of PG-QAOA against alternative algorithms, we use multi-qubit systems. For the purpose of a more comprehensive analysis, we use two different models, discussed in [48, 236].

#### Multi-qubit system I

Consider first the transverse-field Ising model, described by the Hamiltonian [48]:

$$H[h] = -\sum_{j=1}^{N-1} \sigma_{j+1}^z \sigma_j^z - \sum_{j=1}^{N} (\sigma_j^z + h\sigma_j^x). \tag{2.4.2}$$

Here $N$ is the total number of qubits. The global control field $h \in \{\pm 4\}$ can take two discrete values, corresponding to the two alternating QAOA generators $H_0 = H[-4]$ and $H_1 = H[+4]$, cf. Eq. (2.4.2). The initial state $|\psi_i\rangle$ is the ground state of $H[-2]$, and the target state $|\psi_*\rangle$ is chosen to be the ground state of $H[+2]$, so the adiabatic regime is not immediately obvious; both states exhibit paramagnetic correlations and area-law bipartite entanglement. The overlap between the initial and target states

goes down exponentially with increasing the number of qubits $N$ (with all other parameters kept fixed). This state preparation problem is motivated by applications in condensed matter theory. For $N > 2$, this qubit control problem was recently shown to exhibit similarities with optimization in glassy landscapes [82]; for $N = 2$ there exist durations $T$ for which the optimal solution is doubly-degenerate and the optimization landscape features symmetry-breaking [47].

Additionally, we can also turn on small random Hamiltonian noise to the interaction terms on the first two bonds of the spin system, denoted by $\omega_{1,2}$, which would mimic gate imperfections in the context of quantum computing:

$$H[h;\omega_1,\omega_2] = -\left(1+\omega_1\right)\sigma_1^z\sigma_2^z - \left(1+\omega_2\right)\sigma_2^z\sigma_3^z - \sum_{j=3}^{N-1}\sigma_j^z\sigma_{j+1}^z - \sum_{j=1}^{N}\left(\sigma_j^z + h\sigma_j^x\right) \quad (2.4.3)$$

The choice of noisy bonds is arbitrary. To keep the notation compact, we define the noise tuple $\delta = (\omega_1, \omega_2)$. Each $\omega_i \sim \text{uniform}(\Delta)$ with $\Delta$ the support of the uniform distribution.

**Multi-qubit system II**

Consider another benchmark example [236]. Here, we choose the two alternating Hamiltonians from QAOA as

$$H_0 = \frac{1}{2}\left(\sigma_N^z + I_N\right), \quad H_1 = \sum_{i=1}^{N-1}\left(\sigma_i^x\sigma_{i+1}^x + \sigma_i^y\sigma_{i+1}^y\right), \quad (2.4.4)$$

where $I_N$ is the identity operator. The initial state is the product state $|\psi_i\rangle = |\overline{1}\rangle = |1\rangle_1|0\rangle_2\cdots|0\rangle_N$, and the target state is the product state $|\psi_*\rangle = |\overline{N}\rangle = |0\rangle_1|0\rangle_2\cdots|1\rangle_N$. This population transfer problem amounts to a qubit transfer.

The noisy multi-qubit system II uses the gate Hamiltonians:

$$H_0 = \frac{1}{2}\left(\sigma_N^z + I_N\right), \quad H_1(\delta) = \sum_{i=1}^{N-1}\left(\sigma_i^x\sigma_{i+1}^x + \sigma_i^y\sigma_{i+1}^y\right) + \delta\sigma^z_{\lceil\frac{N}{2}\rceil-1}\sigma^x_{\lceil\frac{N}{2}\rceil}\sigma^z_{\lfloor\frac{N}{2}\rfloor+1}, \quad (2.4.5)$$

with $\delta \sim \text{uniform}(\Delta)$. Here, the three-body noise term breaks the particle number (a.k.a. magnetization) symmetry of the original noise-free system.

## 2.5 Numerical Experiments and Results

The models we introduced in Section 2.4 were also studied in [97] using SCP to mitigate the error due to Hamiltonian noise. First, we benchmark our results against

the derivative-based algorithms SCP and b-GRAPE [337] in the context of the Hamiltonian noise. We also present results for PG-QAOA in the context of the Gaussian noise and quantum measurement noise. Then we compare our results to other derivative-free optimization methods, including Nelder-Mead [108], Powell [250], covariance matrix adaptation (CMA) [133], and particle swarm optimization (PSO) [282].

All numerical experiments are performed on the Savio2 cluster at Berkeley Research Computing (BRC). Each node is equipped with Intel Xeon E5-2680 v4 CPUs with 28 cores. The PG-QAOA is implemented in the TensorFlow 1.14 [1] along with TensorlFlow Probability 0.7.0 [89]. The quantum Hamiltonian environment is implemented using QuSpin [328, 329] and QuTIP [150, 149]. The two blackbox optimization methods CMA and PSO are implemented with Nevergrad [257].

Throughout, we used the Adam optimizer [163] to train PG-QAOA with learning rate $10^{-2}$, and learning rate decay of 0.96 applied every 50 iteration steps. The training batch size $M$ is chosen either 128 or 2048 (see figure captions). The initial values for the standard deviation parameters of the policy, $\sigma_{\alpha_i}^{(0)}, \sigma_{\beta_i}^{(0)}, i = 1, 2, \cdots, p$, are either set to 0.0024 or sampled from truncation log normal distribution with mean $-3.0$ and standard deviation 0.1. In the Single-qubit testcase (cf. Section 2.5) and the Multi-qubit I testcase (cf. Section 2.4), the initial values for the mean parameters of the policy, $\mu_{\alpha_i}^{(0)}, \mu_{\beta_i}^{(0)}, i = 1, 2, \cdots, p$, are randomly sampled from a truncated normal distribution with mean 0.5 and standard deviation 0.1. In the Multi-qubit II testcase (cf. Section 2.4) , the initial values for the means are sampled from a truncated normal distribution with mean 3.0 and standard deviation 0.1. In practice, we noticed that the performance of PG-QAOA is sensitive to the initialization of the means $\mu_{\alpha_i}, \mu_{\beta_i}$. In some cases, the initialization was tuned to achieve better performance (c.f. Figure 2.6).

In the numerical experiments, we do not enforce hard constraints on the positivity of $\alpha_i$ and $\beta_i$; yet, in practice we were still able to obtain protocols with positive $\alpha_i \geq 0$ and $\beta_i \geq 0$. This is mainly because the initialization of the mean parameters in the policy is positive and sufficiently far away from zero, and because there are already optimal protocol solutions (i.e. local minima of the control landscape close to the initial values) with positive $\alpha_i$ and $\beta_i$.

## Single qubit results

Figure 2.2 (topmost row) shows snapshots of the policy during training for PG-QAOA in the noise-free case. We sample a batch of protocols from the policy learned in the

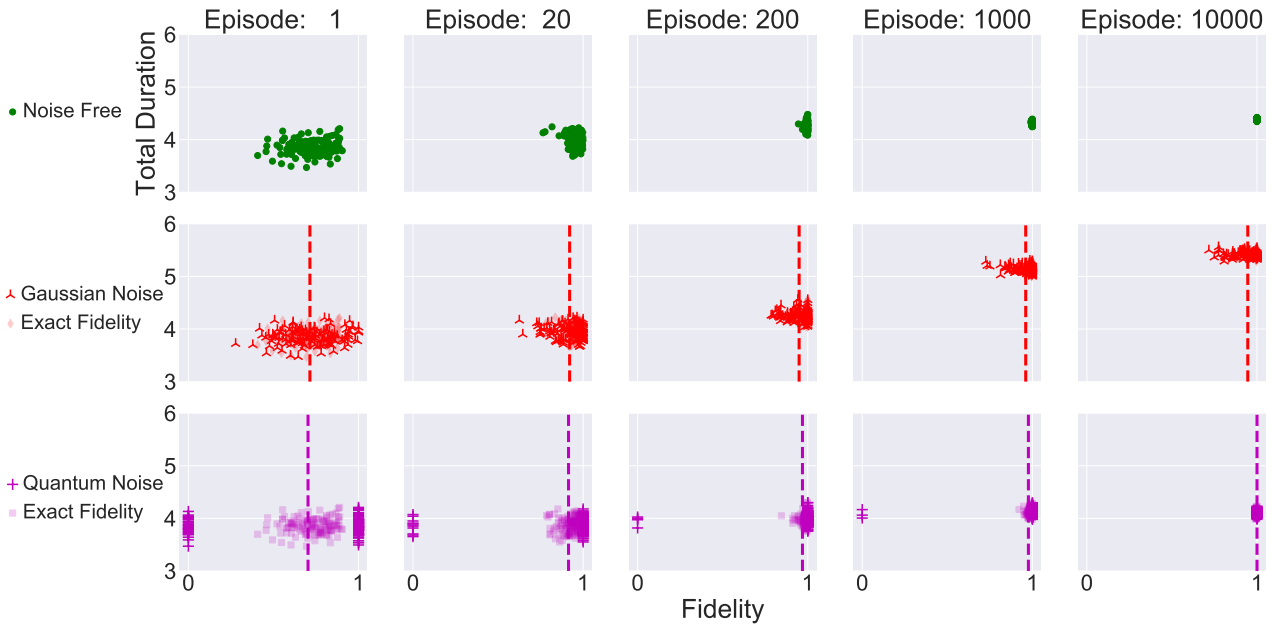



Figure 2.2: The distribution in the learning process for the single-qubit testcase. From left to right, snapshots of the training batch distribution in the (protocol duration, fidelity) space at different training episodes for PG-QAOA. Top row: noise-free fidelity problem (green circles). Middle row: Gaussian fidelity noise problem (red tri-ups), the corresponding exact fidelity values for comparison only (red diamonds, not used in training), and the mean mini-batch fidelity (dashed vertical line). Bottom row: quantum measurement noise problem (magenta crosses) with binary values $\{0, 1\}$, the corresponding exact fidelity values for comparison only (magenta squares, not used for training), and the mean mini-batch fidelity (dashed vertical line). The final learned distributions represent a set of solutions with different total protocol durations but still sharing the same optimal fidelity, demonstrating the machine learning aspect of the algorithm (see text). The standard deviation of the Gaussian noise is 0.1. The QAOA depth is $p = 4$. The PG-QAOA algorithm is trained with a single minibatch of size $M = 128$ for a total iteration number $N_{\text{iter}} = 10^4$. The initial mean values $\mu_{\alpha_i}^{(0)}, \mu_{\beta_i}^{(0)}$ are randomly sampled from a truncated normal distribution with mean 0.5 and standard deviation 0.1 (i.e. $\mathcal{N}(0.5, 0.1^2)$) and the initial standard deviation values $\sigma_{\alpha_i}^{(0)}, \sigma_{\beta_i}^{(0)}$ are sampled from a truncated log normal distribution of mean $-3.0$ and standard deviation 0.1 (i.e. Lognormal$(-3.0, 0.1^2)$).

middle of training, and show its distribution in (protocol duration, fidelity)-space. Due to the random initialization of the policy parameters $\boldsymbol{\theta}$, the algorithm starts from a broad distribution. After the number of training episodes (a.k.a. optimization iterations) increases, the mean of the training batch distribution shifts ever closer to the unit-fidelity region, as expected. At the same time, the distribution also shrinks at later training episodes, and becomes approximately a delta-function in fidelity space in the infinite-training-episode limit since the environment for the noise-free problem is deterministic (though the distribution may still have exhibit finite width due to the decay of the learning rate in the optimization procedure).

Figure 2.2 (middle and bottom rows) shows the effect of the two types of noise on the performance of PG-QAOA. We test both the *Gaussian noise*, which takes into account various classical potential measurement uncertainty sources in the lab, as well as the intrinsic *quantum measurement noise* induced by collapsing the wavefunction during measurements. In the case of quantum measurement noise (magenta), we use only binary fidelity values for the reward PG-QAOA, cf. Eq. (2.2.10); the exact fidelity values for the batch (which are not binary) are shown for comparison purposes only. We emphasize that we do not repeat the quantum measurement on the *same* protocol several times, but only take a single quantum measurement for each protocol from the sampled batch in every iteration. The mean batch fidelity is shown as a vertical dashed line. In the case of Gaussian noise (red), the noisy fidelity values used for training are not binary; PG-QAOA is thus well-suited to handle both classical and quantum noise effects. Because we clip the Gaussian-noisy fidelities to fit in the interval $[0, 1]$, the mean fidelity of the policy (vertical dashed red line) remains slightly away from unity even after a large number of training episodes, introducing a small gap, also visible in the training curves for the multi-qubit examples (Figure 2.4, left).

Note that the policy optimized using PG-QAOA converges at later training episodes for both noisy settings (measurement and Hamiltonian noise). An interesting feature is the remaining finite width along the protocol duration axis: these unit-fidelity protocols are indistinguishable from the point of view of the objective function and are thus equally optimal. Hence, above the QSL, PG-QAOA is capable of learning multiple solutions simultaneously, unlike conventional optimal control algorithms, showcasing one of the advantages of using reinforcement learning. We can indeed verify that these distribution points correspond to distinct protocols, by visualizing the batch trajectories on the Bloch sphere (the projective space of the single-qubit Hilbert space), cf. **??**. We mention in passing that, depending on the initialization of the policy parameters, PG-QAOA finds a different (but equivalent w.r.t. the reward) local basin of attraction in the control landscape, as can be seen from the difference

in the mean total protocol duration at later training episodes for the noise-free and the two noisy cases.

## Multi-qubit results

Figure 2.3 shows the training curves of PG-QAOA for an increasing number of qubits $N$ and QAOA depths $p$. In accord with the fact that the multi-spin fidelity decreases exponentially with increasing $N$, the PG-QAOA algorithm takes longer to converge.

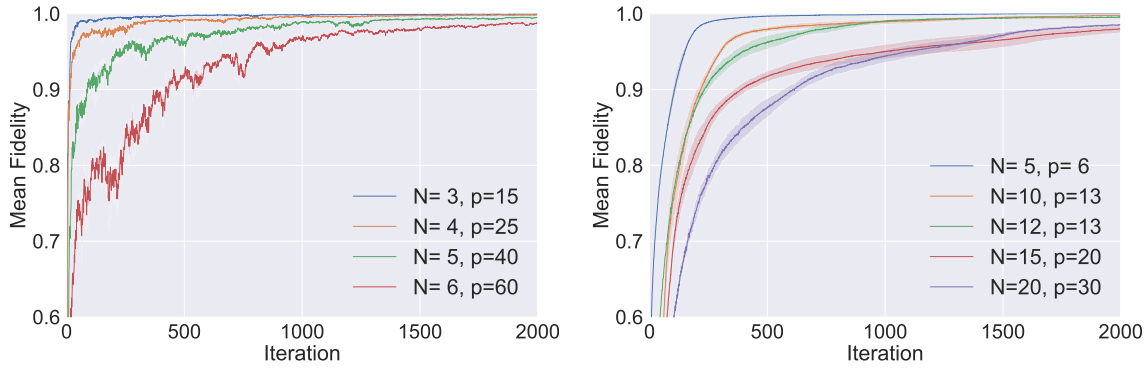



Figure 2.3: Multi-qubit systems, noise-free case. Learning curves (reward vs. episode number) for the Multi-qubit I testcase (a) and the Multi-qubit II testcase (b), for a different number of qubits $N$ and QAOA depth $p$ for three different random seeds. The PG-QAOA algorithm is trained with batch size $M$ of 128 for 2000 iterations. The means initialization is sampled from truncated $\mathcal{N}(0.5, 0.1^2)$ [left] and $\mathcal{N}(1.5, 0.1^2)$ [right]. The stds initialization is from truncated Lognormal$(-3.0, 0.1^2)$.

Adding Gaussian and quantum measurement noise, in Fig. 2.4 we show the training curves for PG-QAOA for $N=3$ qubits. For each noisy case, we present the actual mean fidelities (red for the Gaussian noise and magenta for the quantum measurement noise); the exact fidelities (green) are shown only for comparison and are not used in training. Note that learning from quantum measurements is more prone to noise in the initial stage of the optimization, yet the algorithm converges within a smaller number of episodes compared to the case of the Gaussian noise. For Gaussian noise, similar to the single-qubit case, we observe a small gap between the exact fidelity and the noisy fidelity due to clipping the noisy fidelities to fit within the interval $[0, 1]$. Empirically, we observe the gap size to be almost always about half the Gaussian noise level. This indicates that the probability distribution is moving towards the correct direction (with fidelity close to unity) even though the observed fidelity is away from it due to the noise.

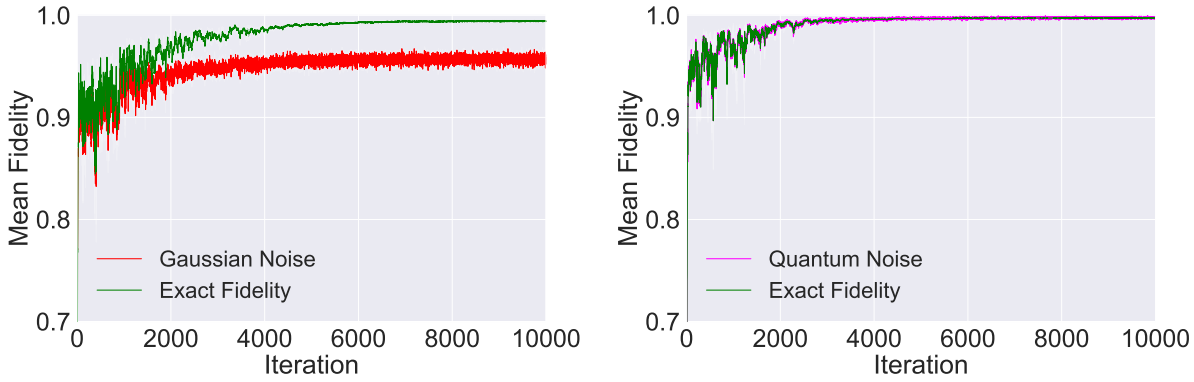



Figure 2.4: Multi-qubit testcase I, training curves: the reward (mean batch fidelity, red for Gaussian noise and magenta for quantum measure noise) used in PG-QAOA against the number of training episodes (i.e. iterations). For comparison purposes only, we also show the exact noise-free mean fidelity (green). Left: Gaussian noise. Right: quantum measurement noise. The standard deviation of the Gaussian noise is 0.1. The number of qubits is $N = 3$. The batch sizes for Gaussian noise and quantum measurement noise are 128 and 2048, respectively. The initial mean values are sampled from truncated $\mathcal{N}(0.5, 0.1^2)$ and the initial standard deviation values – from truncated Lognormal$(-3.0, 0.1^2)$ for both noisy cases.

We now benchmark PG-QAOA against a number of different optimal control algorithms. In order to compare PG-QAOA with state-of-the-art optimization methods using gradient and Hessian information such as b-GRAPE and SCP, we evaluate their performance using both the batch average and the worst-case fidelity as reference. For protocol durations $\{\alpha_i, \beta_i\}_{i=1}^p$, the average and worst-case fidelity within a given support for the uniform distribution $\Delta$, are defined as

$$F_{\text{avg}}(\{\alpha_i, \beta_i\}_{i=1}^p) = \frac{1}{|\Delta|} \int_\Delta F(\{\alpha_i, \beta_i\}_{i=1}^p, \delta) \, \mathrm{d}\delta \tag{2.5.1}$$

$$F_{\text{worst-case}}(\{\alpha_i, \beta_i\}_{i=1}^p) = \min_{\delta \in \Delta} F(\{\alpha_i, \beta_i\}_{i=1}^p, \delta). \tag{2.5.2}$$

A comparison for testcases multi-qubit I and II are shown in Figure 2.5 and Figure 2.6, respectively. In terms of both the average and worst case, PG-QAOA performs comparably to the SCP; although PG-QAOA is derivative-free and uses a first-order derivative optimizer, it can occasionally even reach better solutions than SCP w.r.t. the average fidelity. PG-QAOA clearly outperforms b-GRAPE [337] in the numerical experiments involving a small number of qubits. We also observe a performance

drop for PG-QAOA when the number of qubits is increased. Properly scaling up the performance of PG-QAOA with increasing $N$ remains a topic of further investigation.

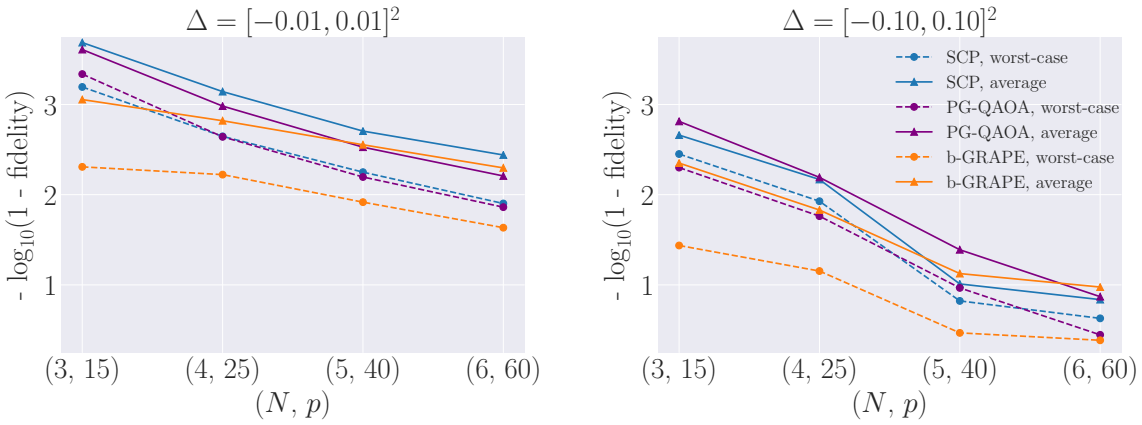



Figure 2.5: Multi-qubit testcase I, algorithms comparison for Hamiltonian gate noise. Fidelity achieved by PG-QAOA (purple), SCP (blue) and b-GRAPE (orange) for a few different numbers of qubits $N$ and total QAOA depth values $p$. The two panels correspond to different values of the support $\Delta$ of the uniform distribution used for the Hamiltonian gate noise. We show both the average fidelity (solid lines), and the worst protocol (dashed lines), cf. Eq. (2.5.1) and Eq. (2.5.2), respectively. The PG-QAOA algorithm is trained with the mini-batch size $M = 128$, except $N = 6$, where $M = 1024$. The initial values for the means are sampled from a truncated $\mathcal{N}(0.5, 0.1^2)$ and the initial values for the standard deviations were kept constant at 0.0024.

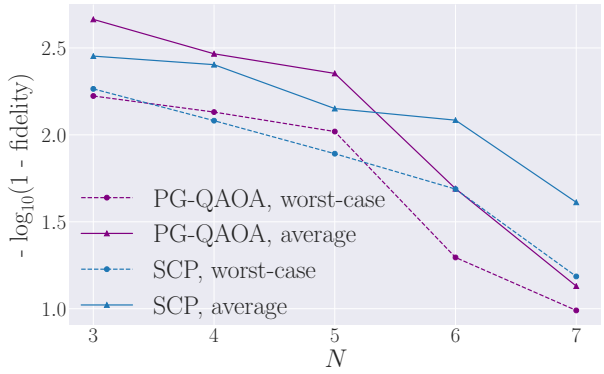



Figure 2.6: Multi-qubit testcase II, algorithms comparison for Hamiltonian gate noise. The comparison between PG-QAOA (purple) and SCP (blue) in terms of robust QAOA for multi-qubit case II with different number of qubits $N$. The QAOA depth is $p = N + 1$ and the support $\Delta$ of the uniform distribution used for the Hamiltonian gate noise is $[-0.15, 0.15]$. We show both the average fidelity (solid lines), and the worst protocol (dashed lines), cf. Eq. (2.5.1) and Eq. (2.5.2), respectively. The PG-QAOA algorithm is trained with minibatch sizes $M = 128$ for $10^4$ iterations. The initial values of the standard deviations are kept constant at 0.0024; the initial values for the means were drawn from $\mathcal{N}(1.0, 0.2^2)$ for $N = 3$, $\mathcal{N}(1.5, 0.2^2)$ for $N = 4$, and $\mathcal{N}(3.0, 0.2^2)$ for $N > 4$.

Last, in Figure 2.7 we show the comparison among other widely used blackbox optimization methods, such as Nelder-Mead, Powell, covariance matrix adaptation (CMA) and particle swarm optimization (PSO). In contrast to PG-QAOA which learns in distribution (i.e. in practice using MC-sampled batches), the other algorithms accept a single scalar cost function value to optimize. Therefore, we use the mean fidelity over a (potentially noisy) training batch; this constitutes a fair comparison, since the mean batch fidelity is precisely the definition of the reward in policy gradient. The different algorithms have a comparable performance in the noise-free case (Figure 2.7, leftmost column). In the presence of measurement noise in the reward function, we observe a decrease in performance in all algorithms. At the same time, PG-QAOA still outperforms other algorithms, which is clearly visible when the number of qubits $N$ increases [3]. PG-QAOA appears less sensitive to the size of the Gaussian noise; moreover, PG-QAOA appears particularly suitable for handling the quantum measurement noise.

[3]Note that, for $N = 6, 8, 10$, we keep $p = 60$ fixed, so the maximum obtainable fidelity is expected to decrease.

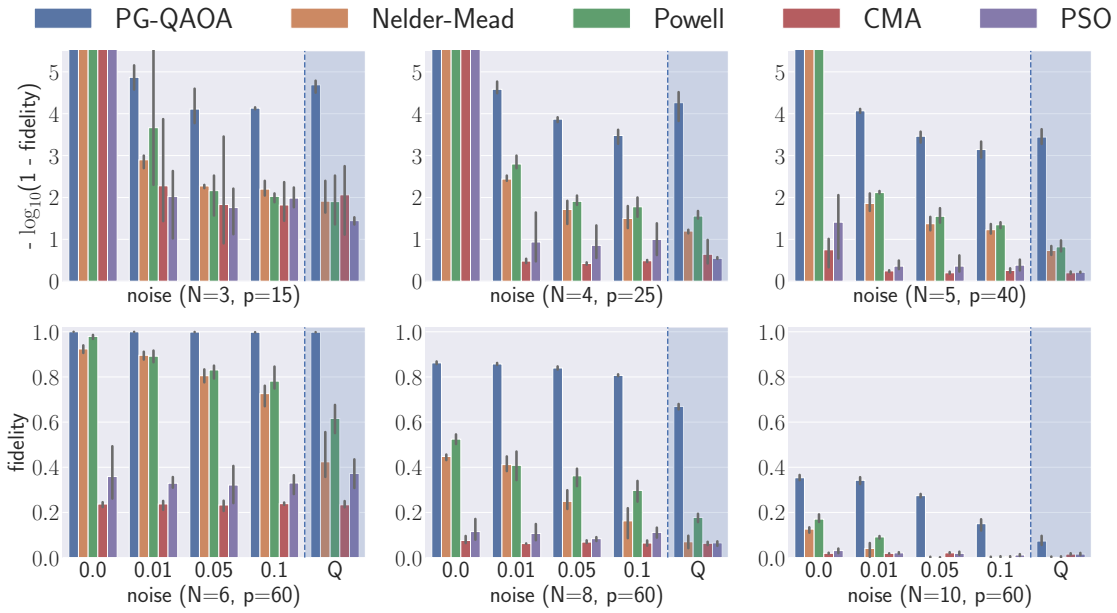



Figure 2.7: Multi-qubit testcase I. Comparison between different optimization algorithms for $N = 3, 4, 5$ qubits (the first row) and $N = 6, 8, 10$ qubits (the second row), and different fidelity noise level (cf. $x$-axis for the standard deviation of the Gaussian noise; the label "Q" (shaded area) stands for quantum measurement noise): PG-QAOA (blue), Nelder-Mead (orange), Powell (green), CMA (red), and PSO (purple). The comparison is in log-scale (upper row), and the normal scale (lower row). PG-QAOA outperforms the rest in the presence of noise. The batch sizes are $M = 2048$ for all the methods, except for $N = 10$, where $M = 256$, and the total number of iterations is $10^4$. For all PG-QAOA experiments, the initial values for the means are sampled from a truncated $\mathcal{N}(0.5, 0.1^2)$ and the standard deviations initialization – from truncated Lognormal$(-3.0, 0.1^2)$.

## 2.6 Conclusion and Outlook

Due to intrinsic limitation of near term quantum devices, error mitigation techniques can be essential for the performance of quantum variational algorithms such as QAOA. Many classical optimization algorithms (derivative-free or those requiring derivative information) may not perform well in the presence of noise. We demonstrate that probability-based optimization methods from reinforcement learning can be well suited for such tasks. This work considers the simplest setup, where we parameterize each optimization variable using only two variables describing an i.i.d. Gaussian distribution. The probability distribution is then optimized using the policy gradient

method, which allows to handle continuous control problems. We demonstrate that PG-QAOA does not require derivatives to be computed explicitly, and can perform well even if the objective function is not smooth with respect to the error. The performance of PG-QAOA may even be sometimes comparable to that of much more sophisticated algorithms, such as sequential convex programming (SCP), which require information of first and second order derivatives of the objective function. PG-QAOA also compares favorably to a number of commonly used blackbox optimization methods, particularly in experiments with noise and other sources of uncertainty.

Viewed from the perspective of reinforcement learning, the Gaussian probability distribution used in this work is one of the simplest possible choices. More involved distributions, such as multi-modal Gaussian distributions, normalizing flow-based models [166, 93], autoregressive models [112],and long short-term memory (LSTM) models may be considered. Based on our preliminary results, these methods can introduce a significantly larger number of parameters, but the benefit is not yet obvious. We can also employ more advanced RL algorithms, such as the natural policy gradient method (NPG) [155], the trust region policy optimization (TRPO) [279] and the proximal policy optimization method (PPO) [278]. Finally, this work only considers implementations on a classical computer. Implementing and testing PG-QAOA on near term quantum computing devices such as those provided by IBM Q will be our future work.

# Chapter 3

# Counterdiabatic Driving inspired Reinforcement Learning based variational quantum algorithms

The quantum alternating operator ansatz (QAOA) is a prominent example of variational quantum algorithms. We propose a generalized QAOA called CD-QAOA, which is inspired by the counterdiabatic (CD) driving procedure, designed for quantum many-body systems and optimized using a reinforcement learning (RL) approach. The resulting hybrid control algorithm proves versatile in preparing the ground state of quantum-chaotic many-body spin chains by minimizing the energy. We show that using terms occurring in the adiabatic gauge potential as generators of additional control unitaries, it is possible to achieve fast high-fidelity many-body control away from the adiabatic regime. While each unitary retains the conventional QAOA-intrinsic continuous control degree of freedom such as the time duration, we consider the order of the multiple available unitaries appearing in the control sequence as an additional discrete optimization problem. Endowing the policy gradient algorithm with an autoregressive deep learning architecture to capture causality, we train the RL agent to construct optimal sequences of unitaries. The algorithm has no access to the quantum state, and we find that the protocol learned on small systems may generalize to larger systems. By scanning a range of protocol durations, we present numerical evidence for a finite quantum speed limit in the nonintegrable mixed-field spin-1/2 Ising and Lipkin-Meshkov-Glick models, and for the suitability to prepare ground states of the spin-1 Heisenberg chain in the long-range and topologically ordered parameter regimes. This work paves the way to incorporate recent success from deep learning for the purpose of quantum many-body control.

## 3.1 Introduction

The ability to prepare a quantum many-body system in its ground state is an important milestone in the quest for understanding and identifying novel collective quantum phenomena. The degree to which ground states can be confidently prepared in present-day quantum simulators, delineates the limits of our capabilities to investigate the properties of new materials or molecules, and to propose innovative technological applications based on quantum effects, such as high-temperature superconductors and superfluids, magnetic field sensors, topological quantum computers, or synthetic molecules.

Quantum simulators, such as ultracold and Rydberg atoms [193, 35], trapped ions [130, 34, 223, 88], nitrogen vacancy centers [96, 275, 58], and superconducting qubits [88, 342], all require the development of state preparation schemes via *real-time* dynamical processes. Despite their high level of controllability, finding short protocols to prepare strongly-correlated ground states under platform-specific constraints, is a challenging problem in AMO-based quantum simulation platforms, due to the exponentially large Hilbert space dimensions of quantum many-body systems. On this background, speed-efficient protocols also become progressively more important for near-term quantum computing devices [17], where simulation errors grow with the protocol duration due to imperfections in the implementation of the basic gate operations.

Developing versatile methods for ground state preparation will enable quantum simulators to investigate hitherto unexplored quantum phases of matter, and determine the behavior of order parameters, correlation lengths and critical exponents. Theoretically, although an exact mathematical expression for the ground state might be known in some models, it remains still largely unclear how to prepare it in a unitary dynamical process. In generic models, the lack of closed-form analytical solutions motivates the use of numerical algorithms. Prominent examples for quantum state preparation include established quantum control algorithms, such as GRAPE [161] and CRAB [53], and variational quantum eigensolvers (VQE) [244], such as the quantum approximate optimization ansatz (QAOA) [102].

In this study, we present a novel hybrid reinforcement learning (RL)/optimal control algorithm based on an autoregressive deep learning architecture. We improve the current state-of-the-art for digital quantum control techniques by enhancing the capabilities to find optimal protocols that prepare the ground state of quantum many-body systems. The emerging versatile algorithm combines discrete and continuous control parameters to achieve maximum flexibility in its applicability to a number of different models.

To cope with the complexity of preparing ordered states in quantum many-body systems, we introduce a novel ansatz inspired by variational gauge potentials and counter-diabatic (CD) driving [85, 29, 173, 46]. This allows us to excite the system away from equilibrium in a controllable manner to find short high-fidelity protocols away from the adiabatic regime. We demonstrate that combining features of CD driving with the digital simulation character of conventional QAOA yields superior performance over a wide range of protocol durations and physical models. Compared to the standard counter-diabatic driving algorithms, CD-QAOA represents a more flexible ansatz which allows us to take into account (i) experimental constraints, such as drift terms that cannot be switched off, and (ii) control degrees of freedom not present in CD driving; (iii) CD-QAOA is not tied to a drive protocol which obeys specific boundary conditions (such as vanishing protocol speed). Unlike continuous CD driving, CD-QAOA offers a simple and easy-to-apply variational ansatz without reference to the exact ground state of the system, paving the way for versatile digital quantum control.

In particular, our RL agent constructs unitary protocols that transfer the population into the ground state of three nonintegrable spin models (spin-1/2 and spin-1 mixed-field Ising chains, and the anisotropic spin-1 Heisenberg chain) which feature long-range and topological order, and the integrable Lipkin-Meshkov-Glick (LMG) model which allows us to present simulations for a large number of particles. We show numerical evidence for the existence of a finite quantum speed limit in the nonintegrable mixed-field spin-1/2 Ising model: an almost perfect system-size scaling indicates that this behavior persists in the thermodynamic limit. Our RL agent has no access to unmeasurable quantum states which grow exponentially with the number of degrees of freedom in the system: this allows the protocols we find to generalize across a number of system sizes [for the spin-1/2 mixed-field Ising model], opening up the door to apply ideas of transfer learning to quantum many-body control. Finally, we demonstrate that the CD-QAOA ansatz has direct practical implications in digital quantum control: it leads to much shorter circuit depths while simultaneously improves the fidelity of the prepared state, which can be utilized to reduce detrimental errors in modern quantum computers.

## 3.2 Generalized Continuous-Discrete Quantum Approximate Optimization Ansatz

To prepare many-body quantum states, we seek a unitary process $U$ which brings the system from a given initial state $|\psi_i\rangle$ to the ground state $|\psi_{\mathrm{GS}}\rangle$ of the Hamiltonian

$H$ (which we call the target state $|\psi_*\rangle$). Typically, Hamiltonians can be decomposed as a sum of two non-commuting parts $H = H_1 + H_2$, e.g. the kinetic and interaction energy. We want to construct

$$U(\{\alpha_j\}_{j=1}^q, \tau) = \prod_{j=1}^{q} U(\alpha_j, \tau_j) \tag{3.2.1}$$

from a sequence $\tau$ of $q$ consecutive unitaries (or their generators) $\tau_j$ chosen from a set $\mathcal{A}$, with $\tau_j \neq \tau_{j+1}$. Each $U(\alpha_j, \tau_j)$ is parametrized by a continuous degree of freedom $\alpha_j$ (e.g. time or rotation angle), i.e. $U(\alpha_j, \tau_j) = \exp(-i\alpha_j\tau_j)$. We formulate state preparation as an optimization problem which consists of determining (i) the sequence $\tau$, and (ii) the values of the variational parameters $\alpha_j$, such that $U|\psi_i\rangle \approx |\psi_{\mathrm{GS}}\rangle$.

Our goal is to prepare the ground state of a Hamiltonian $H$, without having access to the ground state itself. Therefore, we use energy as a cost function

$$E(\{\alpha_j\}_{j=1}^q, \tau) = \langle\psi_i|U^\dagger(\{\alpha_j\}_{j=1}^q, \tau) H U(\{\alpha_j\}_{j=1}^q, \tau)|\psi_i\rangle, \tag{3.2.2}$$

or energy-density $E/N$ which has a well-behaved limit when increasing the number of particles $N$ [1]. We denote the ground state energy by $E_{\mathrm{GS}} = \langle\psi_{\mathrm{GS}}|H|\psi_{\mathrm{GS}}\rangle$.

Note that conventional QAOA is recovered as a special case where one only considers two unitaries $U_j = U(\alpha_j, H_j) = \exp(-i\alpha_j H_j)$, $j = 1, 2$, and $\tau$ is one of the two alternating sequences. Whenever nested commutators of $H_j$ span the entire Lie algebra which generates transport on the complex projective space associated with the Hilbert space $\mathcal{H}$ of the system, applying QAOA is already enough to prepare any state, provided that the underlying circuit depth $q$ is sufficiently large, and the optimal $\alpha_j$ can be found [153]. While true in theory, this is often impractical, since (i) it requires access to in principle unbounded durations, (ii) it increases the number of optimization parameters $\alpha_j$, and – with it – the probability to get stuck in a local minimum of the control landscape, and (iii) the condition that nested commutators of $H_j$ span the entire Lie algebra is generally not satisfied for the $H_j$'s of interest in quantum many-body physics due to, e.g., symmetry constraints.

The generalized QAOA ansatz [Eq. equation 5.2.1] allows us to utilize a larger set of unitaries $\mathcal{A}$ to construct the optimal sequence and to reduce the circuit depth $q$. Inspired by counter-diabatic (CD) driving, we find that a particularly suitable choice in the context of quantum many-body state manipulation, is given by the operators in the adiabatic gauge potential series [Sec. 3.3]. Therefore, we call the

[1] We focus on pure states, although the cost function can trivially be generalized to mixed states.

resulting algorithm CD-QAOA. A different ansatz using more than two unitaries was considered in Ref. [365].

Compared to conventional QAOA, CD-QAOA introduces a discrete high-level optimization to find the optimal protocol sequence $\tau$. The combined optimization landscape can be particularly difficult to navigate, due to the existence of so-called barren plateaus where exponentially many directions have vanishing gradients [212, 60, 118, 146]. Additionally, the total number of all allowed protocol sequences, $|\mathcal{A}|(|\mathcal{A}|-1)^{q-1}$ [2], scales exponentially with the number of unitaries $q$, and presents a challenging discrete combinatorial optimization problem per se; indeed, state preparation, formulated as optimization, can feature a glassy landscape [82, 47] [Sec. 3.11]. However, overcoming these potential difficulties is associated with a potential gain: CD-QAOA allows retaining the flexibility offered by continuous optimization, while increasing the number of independent discrete control degrees of freedom to $|\mathcal{A}|$; this enables us to reach larger parts of the Hilbert space in shorter durations, and with a smaller circuit depth, as compared to conventional QAOA.

Thus, we formulate ground state preparation as a two-level optimization scheme [3]. (1) Low-level optimization: given a fixed sequence $\tau$, we find the optimal values of $\alpha_j$ using a continuous optimization solver, e.g. SLSQP [4] [Sec. 3.9]. To cope with the associated rugged optimization landscape [Sec. 3.11], we run multiple realizations of random initial conditions and post-select the values which yield minimum energy. This continuous optimization problem is also present in conventional QAOA. (2) High-level optimization: in addition to the low-level optimization, we also perform a discrete optimization for the sequence $\tau$ itself, to determine the optimal order in which unitaries from the set $\mathcal{A}$ should occur. To tackle this combinatorial problem, we formulate the high-level optimization as a reinforcement learning (RL) problem. We learn the optimal protocol using Proximal Policy Optimization, a variant of policy gradient. The policy is parameterized by a deep autoregressive network, which allows choosing the control unitaries $U(\alpha_j, \tau_j)$ sequentially. In practice, we sample a batch of sequences from the policy, evaluate the energy of each sequence in the low-level optimization, and apply policy gradient to update the parameters of the policy. This two-level optimization procedure is repeated in a number of training episodes until convergence [Sec. 3.8].

[2] Considering $\tau_j$ as choice of unitaries, we impose the extra constraint that, even though unitaries can be repeated in the sequence $\tau$, the same unitary cannot appear consecutively (or else one can combine the two corresponding choices $\tau_j$ into a single variable).

[3] A similar procedure appeared recently in Ref. [194], although they considered a different problem setup with greedy or beam search algorithm.

[4] In principle, one can use any optimizer which allows constraining the sum $\sum_j \alpha_j = T$.

## 3.3 Variational State Preparation inspired by Counter-Diabatic Driving

A natural question arises as to how to choose the set $\mathcal{A}$ of unitaries for the generalized discrete-continuous QAOA ansatz. One possibility is to consider a set of universal elementary quantum gates, e.g., in the context of a quantum computer [184, 90], and in this case $\alpha_j$ are angles of rotation. We leave this exciting possibility for a future study, and focus here on many-body ground state preparation instead.

The complexity of many-body systems motivates the use of a physics-informed approach to defining the control unitaries in $\mathcal{A}$. Suppose we initialize the system in the ground state of the parent Hamiltonian $H(\lambda = 0)$; we target the ground state of $H(\lambda{=}1)$, seeking the functional form of a time-dependent protocol $\lambda(t)$. If the instantaneous ground state of $H(\lambda)$ remains gapped during the evolution, the adiabatic theorem guarantees the existence of a solution $\lambda(t)$, $t \in [0, T]$, provided $T$ is large compared to the smallest inverse gap along the adiabatic trajectory. However, when the gap is known to close (e.g. across a phase transition), or when the state population transfer has to be done fast, adiabatic state preparation fails.

Compared to the adiabatic paradigm, gauge potentials provide additional control directions in Hilbert space which enable paths that non-adiabatically lead to the target state in a short time. In many-body systems, it is not known in general how to determine the exact gauge potential required for CD driving. However, it is possible to define variational approximations [280, 135] using an operator-valued series expansion [Sec. 3.12] similar to a Schrieffer-Wolff transformation [340], or Shortcuts to Adiabaticity methods [140, 90]. Nonetheless, recent numerical simulations suggest that the exact gauge potential in generic many-body systems is a non-local operator [280, 241] which renders the series expansion asymptotic.

For these reasons, here we consider the constituent terms to every order of the variational gauge potential series, $H_j$, independently, and use them to generate the set of unitaries $\mathcal{A}{=}\{\mathrm{e}^{-i\alpha_j H_j}\}$ for CD-QAOA [5]. We emphasize that our CD-QAOA ansatz is not designed to approximate the gauge potential itself, as opposed to Ref. [341], yet it yields similar benefits w.r.t. preparing the target state. In Sec. 3.5 we compare directly our approach with the variational gauge potential ansatz from Ref. [280].

Since CD-QAOA is a generalization of QAOA aimed to be useful in practice, we need to ensure the accessibility of the control terms $H_j$. Because they appear in

[5] Below, we sometimes abuse notation and set $\mathcal{A}{=}\{H_j\}$, denoting the set of unitaries by their generators.

| short-hand notation | spin operator $H_j$ |
|---|---|
| $X$ | $\sum_i S_i^x$ |
| $Z$ | $\sum_i S_i^z$ |
| $Z\|Z$ | $\sum_i S_i^z S_{i+1}^z$ |
| $Z\|Z+Z$ | $\sum_i J S_i^z S_{i+1}^z + h_z S_i^z$ |
| $Y$ | $\sum_i S_i^y$ |
| $XY$ | $\sum_i S_i^x S_i^y + S_i^y S_i^x$ |
| $YZ$ | $\sum_i S_i^y S_i^z + S_i^z S_i^y$ |
| $X\|Y$ | $\sum_i S_i^x S_{i+1}^y + S_i^y S_{i+1}^x$ |
| $Y\|Z$ | $\sum_i S_i^y S_{i+1}^z + S_i^z S_{i+1}^y$ |
| $X\|Y-XY$ | $\sum_i [S_{i+1}^x - a S_i^x] S_i^y + [S_{i+1}^y - a S_i^y] S_i^x$ |
| $Y\|Z-YZ$ | $\sum_i [S_{i+1}^z - b S_i^z] S_i^y + [S_{i+1}^y - b S_i^y] S_i^z$ |
| $X\hat{Y}$ | $\frac{1}{N} \sum_{i,j} S_i^x S_j^y + S_i^y S_j^x$ |
| $Z\hat{Y}$ | $\frac{1}{N} \sum_{i,j} \left(S_i^z + \frac{I}{2}\right) S_j^y + S_i^y \left(S_j^z + \frac{I}{2}\right)$ |

Table 3.1: Short-hand notation for the generators $H_j$ used to construct the set of unitaries $\mathcal{A} = \{\mathrm{e}^{-i\alpha_j H_j}\}_{j=1}^{|\mathcal{A}|}$ in CD-QAOA. The | indicates operators acting on neighboring sites. Terms from the variational gauge potential series are shown in the lower group [cf. Sec. 3.12 for the derivation].

the first few orders of the gauge potential series, $H_j$ are (sums of) *local* many-body operators [cf. Sec. 3.12]. Thus, in principle, there is no physical obstruction to emulate them in the lab, although this depends on the details of the experimental platform (especially for the interaction terms). Additionally, in the context of many-body systems where energy is extensive, in order to guarantee that we do not tap into a source of infinite energy, we constrain the norm of the generators $\alpha_j H_j$: we view $\alpha_j \geq 0$ as time durations, and fix $\sum_{j=1}^{q} \alpha_j = T$, with $T$ the total protocol duration. This keeps $\alpha_j$ on the same order of magnitude as the coupling constants in the parent Hamiltonian whose ground state we want to prepare.

## 3.4 Many-Body Ground State Preparation

We consider four non-integrable many-body systems of increasing complexity: the spin-1/2 and spin-1 mixed-field Ising models, the spin-1 Heisenberg model, and the integrable Lipkin-Meshkov-Glick (LMG) model where a large number of degrees of freedom is accessible in a classical simulation. The goal of the RL agent is to prepare their ordered ground states, starting from a product state. To generate training

data, we compute numerically the exact time evolution of the system. We apply CD-QAOA using a set of unitaries built from the terms in the series expansion for the variational gauge potential. To determine the allowed terms in the gauge potential series, cf. Table 3.1 (lower group), we consider the minimal set of symmetries shared by the Hamiltonian and the initial and target states [Sec. 3.12].

## Mixed-Field Spin-1/2 Ising Chain

Consider first the antiferromagnetic mixed-field spin-1/2 Ising chain of $N$ lattice sites

$$\begin{aligned} H &= H_1+H_2, \\ H_1 &= \sum_{j=1}^{N} JS^z_{j+1}S^z_j+h_zS^z_j, \quad H_2=\sum_{j=1}^{N} h_xS^x_j, \end{aligned} \tag{3.4.1}$$

where $[S^\alpha_i, S^\beta_j] = \delta_{ij}\varepsilon^{\alpha\beta\gamma}S^\gamma_j$ are the spin-1/2 operators. We use periodic boundary conditions and work in the zero momentum sector of positive parity. In the following, $J=1$ sets the energy unit, and $h_z/J=0.809$ and $h_x/J=0.9045$. We initialize the system in the $z$-polarized product state $|\psi_i\rangle=|\uparrow\cdots\uparrow\rangle$, and we want to prepare the ground state of $H$ in a short time $T$, i.e., away from the adiabatic regime. We verified that similar results can be obtained starting from $|\downarrow\cdots\downarrow\rangle$.

To acquire an intuitive understanding of the advantages brought by the gauge potential ansatz, consider first the non-interacting system at $J = 0$, for which the control problem reduces to a single spin. Both the initial and target states lie in the $xz$-plane of the Bloch sphere, and hence the shortest unit-fidelity protocol generates a rotation about the $y$-axis. In conventional QAOA, one would construct a $y$-rotation out of the $X$ and $Z$ terms [cf. Table 3.1] present in the Hamiltonian. For a single spin, this construction is always possible due to the Euler angle representation of SU(2), but for the interacting spin chain this is no longer the case. The role of the gauge potential $Y$ is to 'unlock' precisely this geodesic in parameter space, and make it accessible as a dynamical process. This allows preparing the target state faster, compared to the original $X, Z$ control setup. In the language of variational optimization, an accessible $Y$ term includes the shortest-distance protocol into the variational manifold, and the RL agent easily finds the exact solution [Sec. 3.13].

For the interacting system, $J>0$, applying conventional QAOA using the two gates $U_j=\mathrm{e}^{-i\alpha_jH_j}$ with $H_1=Z|Z+Z$ and $H_2=X$ is straightforward, but it does not yield a high-fidelity protocol [Fig. 3.1 (blue squares)]. It was recently reported that much

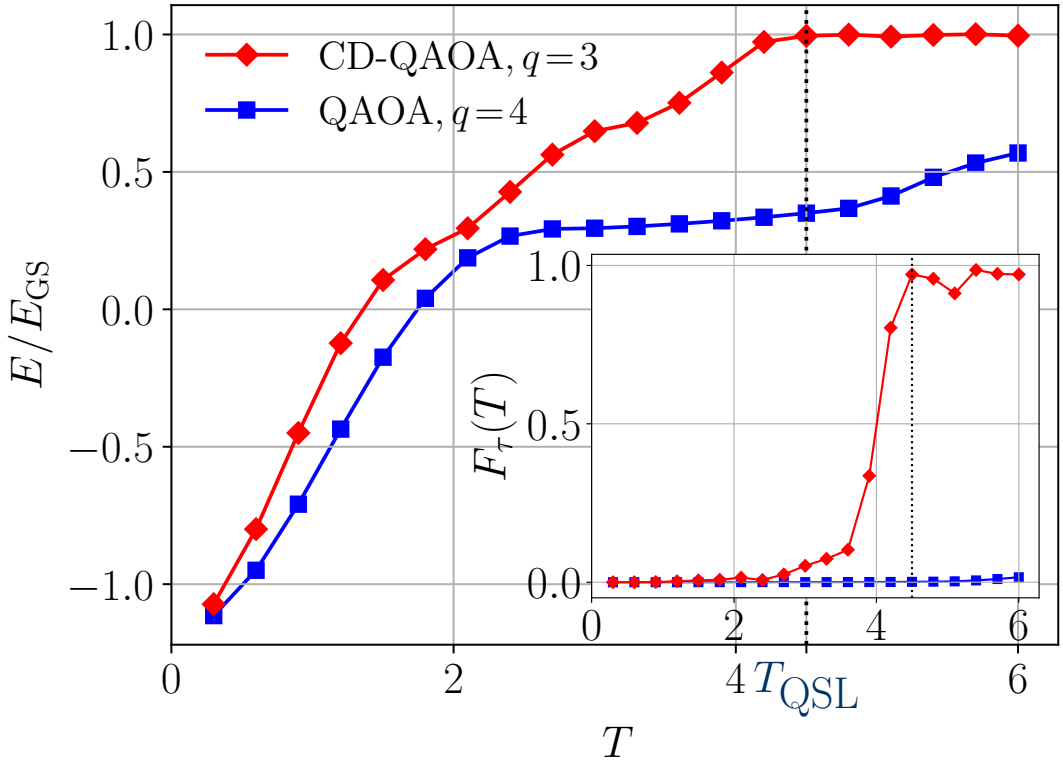



Figure 3.1: Spin-1/2 Ising model: energy minimization and the corresponding many-body fidelity [inset] against protocol duration $T$ obtained using conventional QAOA (blue squares) and CD-QAOA (red diamonds) with circuit depths $p = q/2 = 2$ and $q = 3$, respectively. The dotted vertical line marks the quantum speed limit $T_{\mathrm{QSL}}$. CD-QAOA outperforms conventional QAOA. The initial and target states are $|\psi_i\rangle = |\uparrow \cdots \uparrow\rangle$ and $|\psi_*\rangle = |\psi_{\mathrm{GS}}(H)\rangle$ for $h_z/J = 0.809$ and $h_x/J = 0.9045$. The alternating unitaries for conventional QAOA are generated by $\mathcal{A}_{\mathrm{QAOA}} = \{Z|Z+Z, X\}$[cf. Eq. equation 4.2.3]; for CD-QAOA, we extend this set using adiabatic gauge potential terms to $\mathcal{A}_{\mathrm{CD-QAOA}} = \{Z|Z+Z, X; Y, X|Y, Y|Z\}$. The cardinality of the CD-QAOA sequence space is $|\mathcal{A}|(|\mathcal{A}|-1)^{q-1} = 80$. The number of spins is $N = 18$ with a Hilbert space size of $\dim(\mathcal{H}) = 7685$.

better energies can be obtained, using a three-step QAOA which consists of the three terms in the Hamiltonian equation 4.2.3, $Z|Z$, $X$, and $Z$, applied in a fixed order [209]; invoking again an Euler angle argument provides an explanation: the $X$ and $Z$ terms effectively generate the $Y$ gauge potential term.

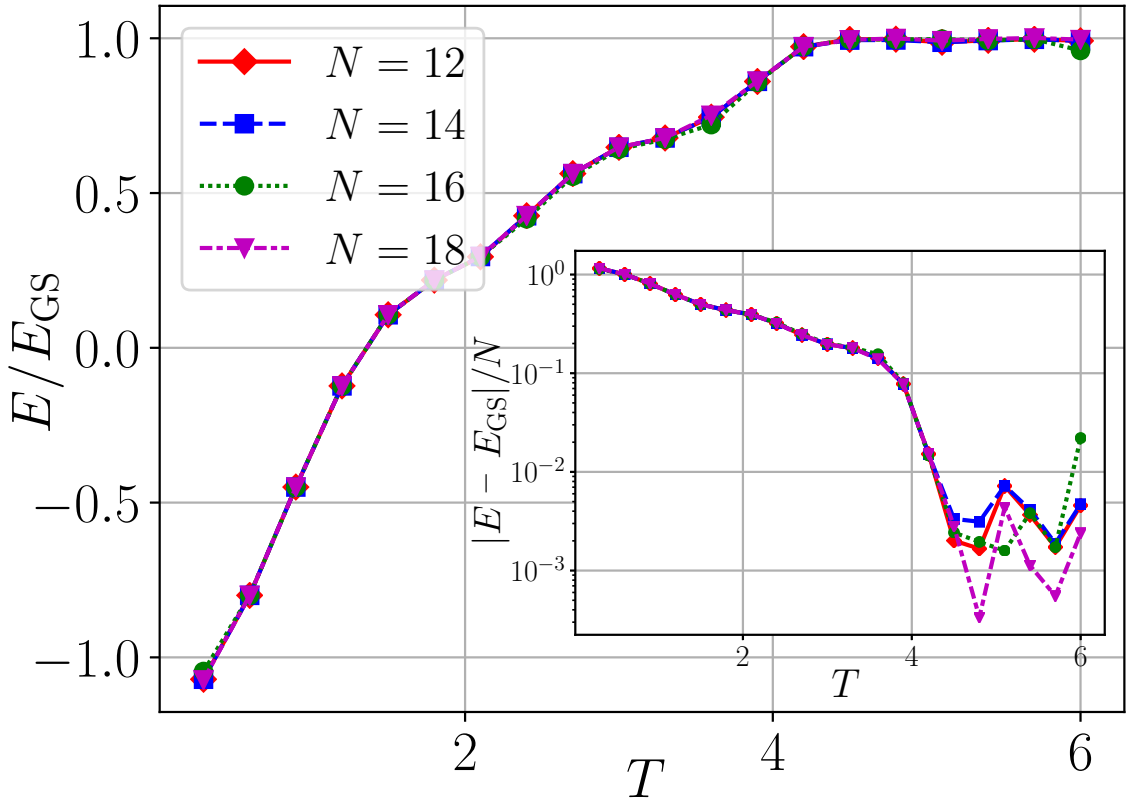



Figure 3.2: Spin-1/2 Ising model: energy minimization and the corresponding mean absolute error [inset, log scale] against protocol duration $T$ for different system sizes using CD-QAOA with circuit depths $q=3$. system-size scaling of the variational energy density suggests the results hold for larger systems. For the number of spins of $N=12,14,16,18$, the Hilbert space sizes are $\dim(\mathcal{H})=224,687,2250,7685$ respectively. The model parameters are the same as in Fig. 3.1.

In stark contrast to conventional QAOA, adding just the zero-order term $H_3=Y$ from the gauge potential series [Sec. 3.12], we find that CD-QAOA already gives a significantly improved protocol; this is achieved by the high-level discrete optimization which selects the order of the operators in the sequence. However, we can do better: since $|\psi_i\rangle$ is a product state while $|\psi_*\rangle$ is not, and because $H_3$ is a sum of single-particle terms, in order to create the target many-body correlations using a fast dynamical process, we also include the two-body first-order gauge potential terms $H_4=X|Y$ and $H_5=Y|Z$: this results in a nonadiabatic evolution that prepares the interacting ground state to an excellent precision [Fig. 3.1 (red diamonds)].

In Ref. [141], it was shown that, in the integrable limit $h_z = 0$, one can prepare the ground state of the system at the critical point using a circuit of depth $q = 2N$ with conventional QAOA. Albeit for the specific initial and target states chosen, we find that it only takes CD-QAOA a depth of $q = 3$ to reach the target ground state, independent of the system size $N$ [6]. This result, though model-dependent, may come as a surprise at first sight, given that the mixed field Ising chain is a quantum chaotic system without a closed-form solution which makes it susceptible to heating away from the adiabatic limit.

Our data also reveals a finite many-body QSL at $T_{\mathrm{QSL}} \approx 4.5$. Importantly, this QSL appears insensitive to the system size to a very good approximation [Fig. 3.2], and we expect it to persist in the thermodynamic limit. The absence of a finite QSL in conventional QAOA in the mixed-field Ising chain suggests that the observation of a QSL using CD-QAOA depends on the specific set of unitaries related to the variational gauge potential, showcasing the utility of our ansatz for many-body control. Remarkably, we find an almost perfect system-size collapse of the target state energy density curves as a function of the total protocol duration $T$. In Sec. 3.6, we explore this feature and demonstrate the ability of the RL agent to learn on small system sizes and subsequently generalize its knowledge to control bigger systems with exponentially larger Hilbert spaces.

CD-QAOA performs successfully on the nonintegrable spin-1/2 mixed-field Ising chain, for a circuit depth as short as $q = 3$. This shows an advantage of our ansatz, when compared to conventional QAOA. However, the small size of the sequence space, $|\mathcal{A}|(|\mathcal{A}| - 1)^{q-1} = 80$ at $|\mathcal{A}| = 5$, poses a natural question regarding the necessity of using sophisticated search algorithms, such as RL, to find the control sequence. We now show that this is a peculiarity of the physical system, as we turn our attention to a larger sequence space.

## Heisenberg Spin-1 Chain

The eight-dimensional spin-1 group SU(3) provides a significantly larger space of gauge potential terms to build the optimal protocol from. We consider a total of $|\mathcal{A}| = 9$ unitaries: five are generated by the imaginary-valued terms in the gauge potential series: $Y, XY, YZ, X|Y, Y|Z$ [cf. Table. 3.1], plus the two real-valued QAOA operators $H_1$ and $H_2$, which build the Hamiltonian $H = H_1 + H_2$ whose ground state we target [Eq. equation 3.4.2], and the two real-valued Hamiltonian terms $X|X$ and $Z$. At $q = 18$, this amounts to $|\mathcal{A}|(|\mathcal{A}| - 1)^{q-1} \approx 10^{16}$ possible sequences. The

[6] The role of the RL algorithm is to decide which three out of the five unitaries $U_j$ to apply and in which order.

exponential scaling of the sequence space size with $q$ renders applying exhaustive search algorithms infeasible, and justifies the use of sophisticated algorithms, such as RL.

The (anisotropic) spin-1 Heisenberg model reads as:

$$\begin{aligned} H &= H_1 + H_2, \\ H_1 &= J\sum_{j=1}^{N}(S^x_{j+1}S^x_j + S^y_{j+1}S^y_j), \quad H_2 = \Delta\sum_{j=1}^{N} S^z_{j+1}S^z_j, \end{aligned} \tag{3.4.2}$$

with the spin exchange coupling $J=1$ set as energy unit, and $\Delta$ – the anisotropy parameter; we use periodic boundary conditions and work in the ground state sector of zero momentum and positive parity, defined by the projector $\mathcal{P}$. In the thermodynamic limit, this model features a rich ground state phase diagram including ferromagnetic (FM, $\Delta/J \ll -1$), XY ($-1 \lesssim \Delta/J \lesssim 0$), topological/Haldane ($0 \lesssim \Delta/J \lesssim 1$), and antiferromagnetic (AFM, $\Delta/J \gg 1$) order [7], with phase transitions belonging to different universality classes [68, 247, 185]. While the FM, XY, and AFM states are characterized by a local order parameter, the gapped Haldane state has topological order not captured by Landau-Ginzburg theory. We consider the AFM initial state $|\psi_i\rangle = \mathcal{P}\,|\uparrow\downarrow\uparrow\downarrow\cdots\rangle$, and target the ground states of Eq. equation 3.4.2 deep in the FM, XY, and Haldane phases, where system-size effects are the smallest. Because CD-QAOA is not restricted to adiabatic evolution, the conventional paradigm of a closing spectral gap when transferring the population between two states displaying different order, does not apply in our non-equilibrium setup, even in the thermodynamic limit.

Figure 3.3 shows a comparison between conventional QAOA with alternating sequence between the Hamiltonians $H_1$ and $H_2$, and CD-QAOA. We find that CD-QAOA shows superior performance for all three ordered ground states: while the gain over conventional QAOA for the Haldane state is already a faster protocol, we clearly see how the gauge potential terms can prove essential for reaching the ground state in the FM and XY phases within the available durations. Note that the FM target state is doubly degenerate, and minimizing the energy, it ends up in an arbitrary superposition within the ground state manifold. Interestingly, we do not identify any distinction from preparing states with long-range and topological order, presumably due to the small system sizes that we reach in our classical simulation.

The CD-QAOA protocol sequences found by the RL agent have peculiar structures [Sec. 3.13]: some of them resemble closely the alternating sequence of conventional QAOA, with the notable difference of applying additional unitaries to rotate the

[7] We define 'order' in the context of phase transitions in condensed matter physics.

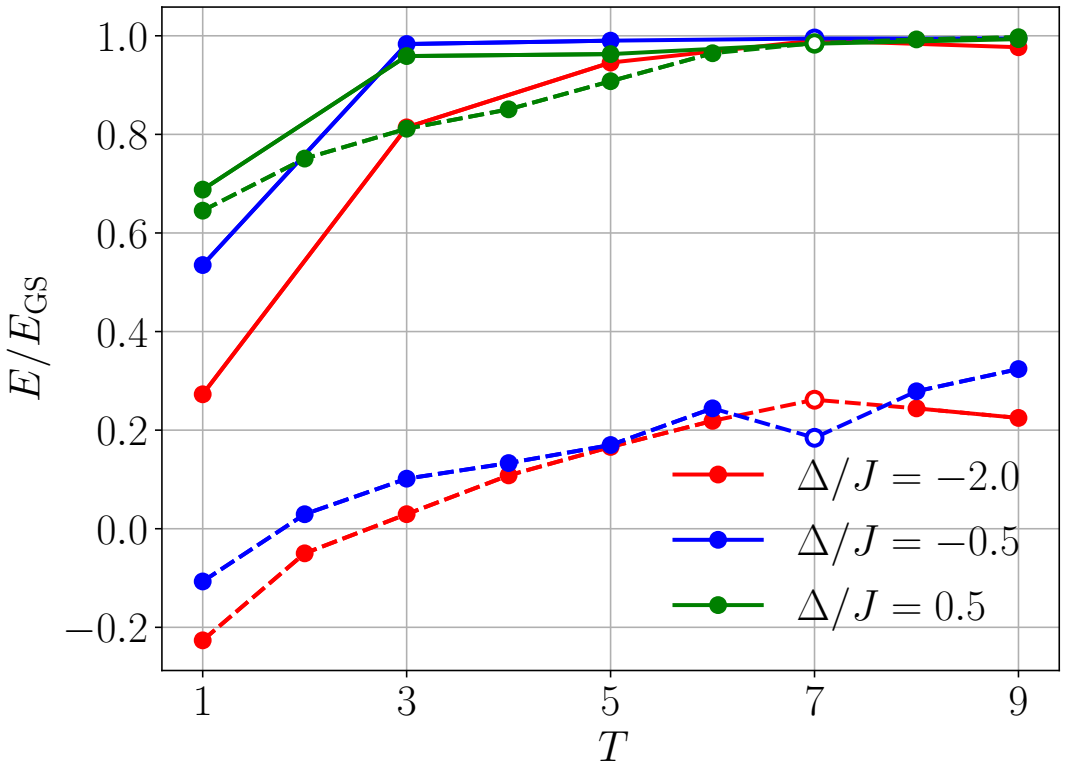



Figure 3.3: Heisenberg spin-1 chain: energy minimization against protocol duration $T$ using conventional QAOA (dashed lines) and CD-QAOA (solid lines) for three different states. We start from the AFM state $|\psi_i\rangle = \mathcal{P}\Uparrow\downarrow\uparrow\downarrow\cdots\rangle$ and target three different parameter regimes, corresponding to the FM ($\Delta/J = -2.0$) state, XY ($\Delta/J = -0.5$), and Haldane ($\Delta/J = 0.5$) states, respectively. CD-QAOA outperforms conventional QAOA ($p = q/2$), more notably in the FM and XY targets where it allows us to reach close to the target state using a short protocol duration. The empty symbols mark the duration at which we show the evolution of the system in Fig. 3.20. The alternating unitaries for conventional QAOA are generated by $\mathcal{A}_{\mathrm{QAOA}} = \{H_1, H_2\}$ [cf. Eq. equation 3.4.2]; for CD-QAOA, we extend this set using adiabatic gauge potential terms to $\mathcal{A}_{\mathrm{CD-QAOA}} = \{H_1, H_2, Z, X|X; Y, XY, YZ, X|Y - XY, Y|Z - YZ\}$. The circuit depths are $q = 28$ ($\Delta/J = -2.0$), $q = 18$ ($\Delta/J = -0.5$) and $q = 18$ ($\Delta/J = 0.5$). The cardinality of the CD-QAOA sequence space is $|\mathcal{A}|(|\mathcal{A}|-1)^{q-1} \approx 10^{16}$ at $q = 18$. The system size is $N = 8$, where $\dim(\mathcal{H}) = 498$.

state to a suitable basis, either at the beginning or at the end of the sequence. While

this is formally equivalent to starting from or targeting a rotated state, the rotations use two-body operators; hence, the resulting basis does not coincide with any of the distinguished $S^x$, $S^y$ and $S^z$ directions. Variationally determining such effective bases demonstrates yet another advantage offered by the CD-QAOA ansatz. Another kind of encountered sequence contains two different sets of alternating unitaries, similar to two independent QAOA ansatzes concatenated one after the other. Finally, for those values of $T$, where CD-QAOA and QAOA have the same performance, we have also observed that CD-QAOA finds precisely the QAOA sequence. In this case, conventional QAOA already generates the shortest path, and the extra gauge potential terms to second-order do not give any advantage; a better performance might be expected when the three- and four-body higher-order terms from the gauge potential series are included.

Similar to other optimal control algorithms, RL agents typically find local minima of the optimization landscape; thus, there is no guarantee that the CD-QAOA protocols provide global optimal solutions; however, these sequences can serve as an inspiration to build future variational ansatzes tailored for many-body systems.

## Lipkin-Meshkov-Glick Model

The non-integrable character of the previously discussed models precludes us from applying CD-QAOA with a large number of degrees of freedom, since reliably simulating their dynamics on a classical computer is prohibitively expensive. In order to study the behavior of CD-QAOA in a large enough system which also features a quantum phase transition, we now turn our attention to an exactly solvable many-body system.

The Lipkin-Meshkov-Glick (LMG) Hamiltonian [196] describes spin-1/2 particles on a fully-connected graph of $N$ sites:

$$\begin{aligned} H &= H_1 + hH_2, \\ H_1 &= -\frac{J}{N}\sum_{i,j=1}^{N} S_i^x S_j^x, \quad H_2 = \sum_{j=1}^{N}\left(S_j^z + \frac{1}{2}\right), \end{aligned} \tag{3.4.3}$$

where $J$ is the uniform interaction strength and $h$ the external magnetic field. In the thermodynamic limit, $N \to \infty$, the system undergoes a quantum phase transition at $h_c/J = 1$ between a ferromagnetic (FM) ground state in the $x$-direction for $h/J \ll 1$, and a paramagnetic ground state for $h/J \gg 1$. The spectral gap $\Delta_{\mathrm{LMG}}$ between the ground state and excited states closes as $\Delta_{\mathrm{LMG}}(h_c) \sim N^{-1/3}$ at the critical point [38]. Realizing the LMG model is within the scope of present-day experiments

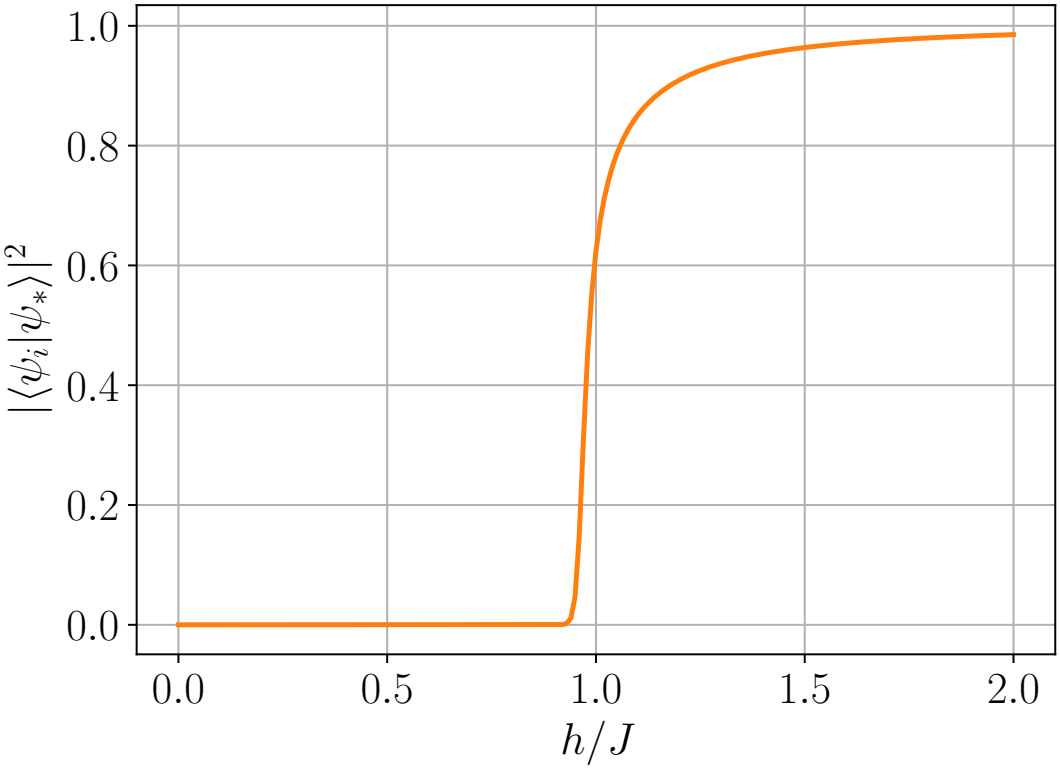



Figure 3.4: LMG model: the overlap between the initial state $|\psi_i\rangle$ and the target state $|\psi_*\rangle$ is vanishingly small in the ferromagnetic phase $h/J \ll 1$, which motivates the parameter choice for the target state. In the vicinity of the critical point, the overlap increases and approaches unity in the limit $h/J \to \infty$. Note that, in the FM phase, the ground state is doubly degenerate, in which case the overlap is computed w.r.t. the ground state manifold: $|\langle\psi_i|\psi_*^{(1)}\rangle|^2 + |\langle\psi_i|\psi_*^{(2)}\rangle|^2$. In the paramagnetic phase, the ground state is unique. We used $N = 501$ spins.

with ultracold atoms [291, 81]; therefore, developing fast ground state preparation techniques can prove useful in practice.

Defining the total spin operators as $S^\alpha = \sum_{j=1}^N S_j^\alpha$, the Hamiltonian takes the form $H = -J/N\,(S^x)^2 + h\,(S^z + N/2)$. Hence, the total spin is conserved, $[H, S\cdot S] = 0$, and the ground state symmetry sector contains a total of $N+1$ states, i.e. $\dim(\mathcal{H}) = N+1$, which allows us to simulate large system sizes.

Our goal is, starting from the $z$-polarized paramagnetic initial state, $|\psi_i\rangle = |\downarrow\downarrow\cdots\rangle$, to target an arbitrary superposition in the doubly-degenerate FM ground state manifold, at fixed values of the external field $h/J$ which controls the magnitude of the transversal fluctuations on top of the ferromagnetic order. Figure 3.4 shows that the overlap of the initial and target states is vanishingly small in the FM phase, and approaches quickly unity across the critical point into the paramagnetic phase. Therefore, we choose to prepare ground states in the FM phase where the problem naturally appears more difficult.

Figure 3.5 shows a comparison between CD-QAOA and QAOA on the LMG model at $h/J = 0.5$ for $N = 501$ spins [more $h/J$ values are shown in Sec. 3.13]. First, note the superior performance of CD-QAOA, as compared to conventional QAOA in a range or short durations $T$ in the nonadiabatic driving regime. We applied CD-QAOA with two different sets of generators: $\mathcal{A} = \{H_1, H_2; Y\}$, and $\mathcal{A}' = \{H_1, H_2; Y, \hat{XY}, \hat{ZY}\}$ [cf. Table 3.1] and found that, for the LMG model, the higher-order two-body terms $\hat{XY}, \hat{ZY}$ do not offer any advantage deep in the FM phase. This observation can be understood as follows: to turn the $z$-polarized initial state into the $x$-ferromagnet, it is sufficient to perform a rotation about the $y$-axis, which coincides precisely with the single-body term in the gauge potential series expansion [cf. Sec. 3.12]. Indeed, for all protocol durations smaller than the quantum speed limit, $T < T_{\mathrm{QSL}}$, the RL agent finds that the optimal protocol consists of a single $Y$-rotation, while for $T \geq T_{\mathrm{QSL}}$ the optimal protocol is degenerate, and typically involves the various terms from $\mathcal{A}$. This finding allows us to extract the QSL as a function of the external field $h$, cf. Fig. 3.6.

Close to the critical point $h_c$, we observe strong sensitivity in the best found protocols to system-size effects, and a single $Y$-rotation is no longer optimal below the QSL. Interestingly, at the critical point (and in the paramagnetic phase), the optimal protocol is given by QAOA: in this regime, despite the larger set of terms $\mathcal{A}$ we use in CD-QAOA, the RL agent correctly identifies the sequence of alternating $H_1$ and $H_2$ terms as optimal, which shows the versatility of CD-QAOA: the algorithm can always select a smaller effective subspace of actions when this is advantageous in the parameter regime of interest.

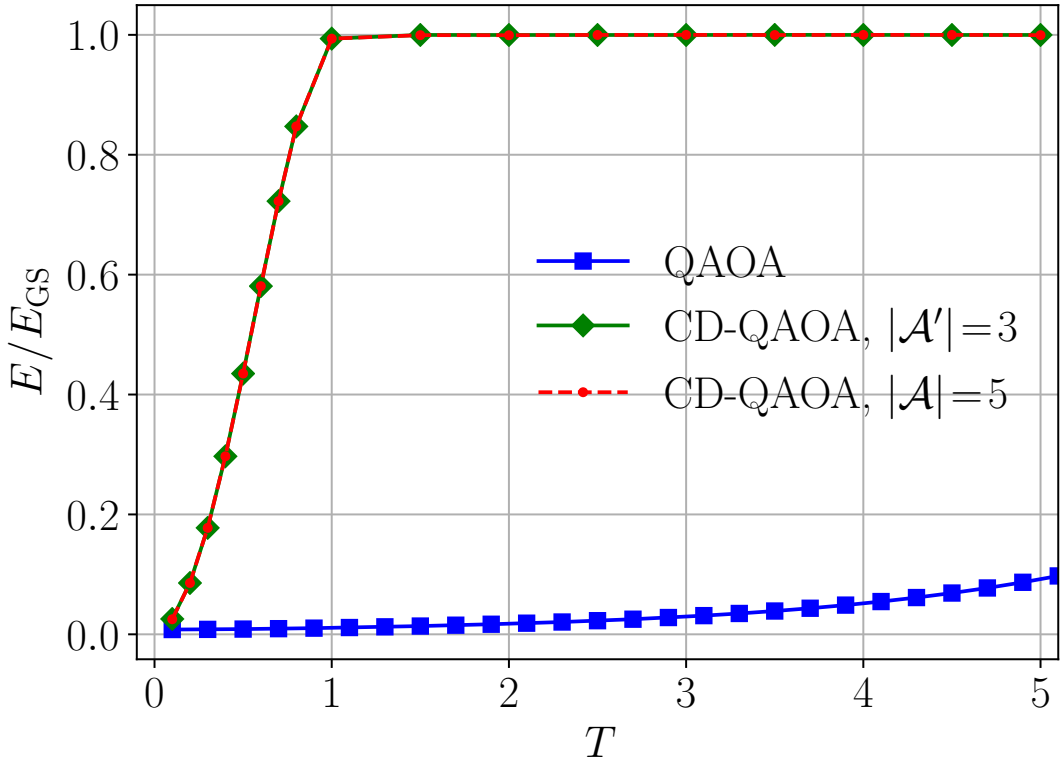



Figure 3.5: LMG model: energy minimization against protocol duration $T$ using conventional QAOA (blue square) and CD-QAOA (red dashed line, green solid line). We start from the $z$-polarized state $|\psi_i\rangle = |\downarrow\downarrow\cdots\rangle$ and target ground state of LMG Hamiltonian equation 3.4.3. CD-QAOA significantly outperforms conventional QAOA for short durations. The alternating unitaries for conventional QAOA are generated by $\mathcal{A}_{\mathrm{QAOA}} = \{H_1, H_2\}$ [cf. Eq. equation 3.4.3]; for CD-QAOA, we extend this set using adiabatic gauge potential terms to $\mathcal{A}_{\mathrm{CD-QAOA}} = \{H_1, H_2; Y, \hat{XY}, \hat{YZ}\}$ and $\mathcal{A}'_{\mathrm{CD-QAOA}} = \{H_1, H_2; Y\}$. The external field is $h/J = 0.5$; the circuit depth is $q = 8$, and the system size is $N = 501$, where effective Hilbert dimension $\dim(\mathcal{H}) = 502$.

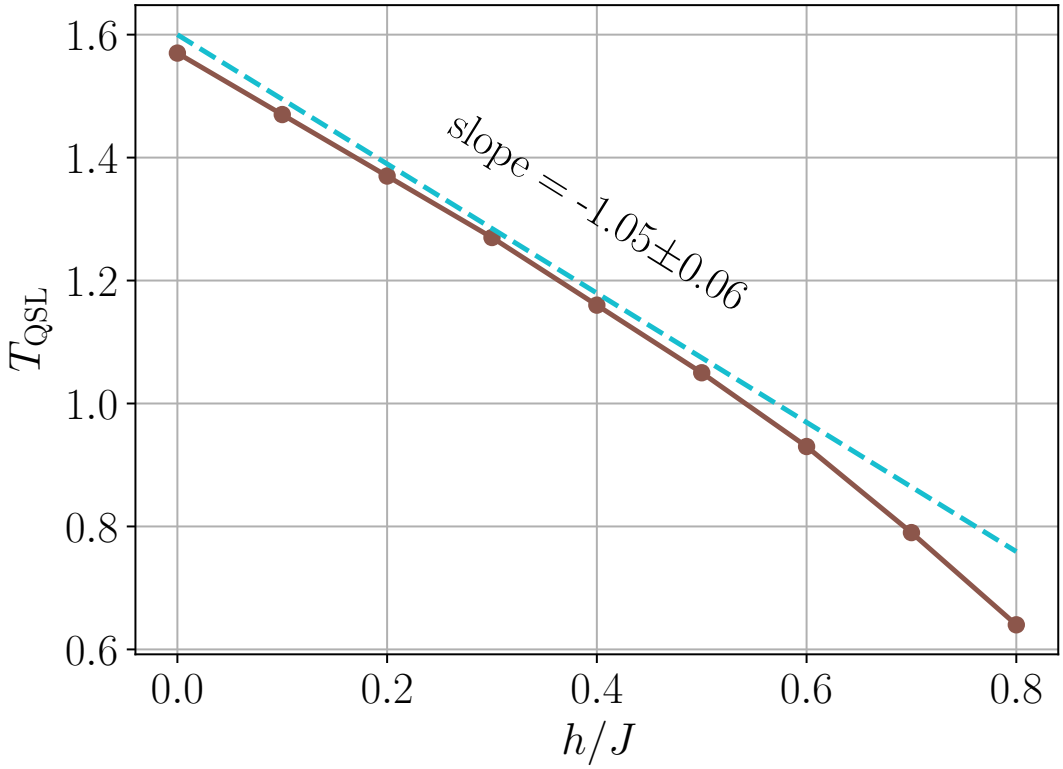



Figure 3.6: LMG model: Quantum speed limit, $T_{\mathrm{QSL}}$, as a function of the transverse field $h$, for a target state in the ferromagnetic phase. At $h/J = 0$, we have $T_{\mathrm{QSL}} = \pi/2$, which is the angle required to turn the $z$-polarized initial state into the $x$-ferromagnet. For finite $h/J$ quantum fluctuations in the target ferromagnetic ground state decrease the angle required to transfer the population from the initial state, which results in a smaller value of $T_{\mathrm{QSL}}$. The dashed cyan line is a least squares fit for small values of $h/J$, suggesting the behavior $T_{\mathrm{QSL}}(h) = -h/J + \pi/2 + \mathcal{O}(h^2)$. We used $N = 501$ spins.

## 3.5 Comparison with Counter-Diabatic Driving

To compare and contrast the CD-QAOA ansatz with CD and adiabatic driving [280], consider the driven spin-1 Ising model [8]:

$$\begin{aligned} H(\lambda) &= \lambda(t)H_1 + H_2, \qquad (3.5.1) \\ H_1 &= \sum_{j=1}^{N} J S_{j+1}^z S_j^z + h_x S_j^x, \qquad H_2 = \sum_{j=1}^{N} h_z S_j^z, \end{aligned}$$

[8] We deliberately use a different form in Eq. equation 4.9.1 as compared to Eq. equation 4.2.3; the former may appear more natural in quantum many-body physics, where the transverse-field Ising model $H_1$ can be mapped to free fermions.

where $\lambda(t) = \sin^2\left(\frac{\pi t}{2T}\right)$, $t \in [0, T]$, is a smooth protocol satisfying the boundary conditions for CD driving: $\lambda(0)=0$, $\lambda(T)=1$, $\dot{\lambda}(0)=0=\dot{\lambda}(T)$. The initial state is the ground state at $t=0$, i.e. $|\psi_i\rangle=|\downarrow\cdots\downarrow\rangle$, while the target state is the ground state of the Ising model at $t = T$ for $h_z/J = 0.809$ and $h_x/J = 0.9045$. Unlike the setup in Sec. 3.4, adiabatic state preparation following the protocol $\lambda(t)$, suggests using the QAOA generators $\mathcal{A}_{\text{QAOA}} = \{H_1, H_2\}$.

Figure 3.7 shows a comparison between different methods using the best found energy density (main figure), and the corresponding many-body fidelity (inset). Let us focus on CD-QAOA and QAOA first. As expected, CD-QAOA (red) performs better for short durations $T$, since it contains conventional QAOA (red) as an ansatz, i.e. $\mathcal{A}_{\text{QAOA}} \subsetneq \mathcal{A}_{\text{CD-QAOA}}$. We emphasize that such a performance is not guaranteed in practice, since it is conceivable that the RL agent gets stuck in a local minimum associated with lower energy than the QAOA solution [Sec. 3.11], e.g., if the deep autoregressive network architecture is not expressive enough, or if the learning rate schedules are not well-tuned to the problem. Unlike the spin-1/2 Ising model, here we cannot clearly identify a finite QSL, as the CD-QAOA energy keeps improving with increasing circuit depth $q$ [Sec. 3.8].

To construct the counter-diabatic Hamiltonian $H_{\text{CD}} \approx H(\lambda)+\dot{\lambda}\mathcal{X}(\{\beta_j\})$ for Eq. equation 4.9.1, we make a variational ansatz [280] for the gauge potential $\mathcal{X}$, and solve for the optimal parameters $\beta_j$ numerically [Sec. 3.12]. We note the following differences between this approach and CD-QAOA: (i) the variational gauge potential depends on time $t$ continuously, which requires further discretization when performing a gate-based implementation. (ii) the number of variational parameters in the standard variational gauge potential method is $N_T|\mathcal{A}|$ with $N_T$ the number of steps used to discretize the time interval $[0, T]$; instead, in CD-QAOA, we have $q$ variational parameters. (iii) the variational gauge potential method does not constrain the magnitude of the variational coefficients $\beta_j$, and hence the time-averaged norm of $H_{\text{CD}}$ over the protocol can grow indefinitely; especially for short durations $T$ this typically gives a higher fidelity. By contrast, in CD-QAOA the time-averaged norm of the unitary generators $\alpha_j H_j$ summed along the sequence, is kept bounded via the constraint $\sum_j \alpha_j = T$. Nonetheless, in practice, we find that these norms are on the same order of magnitude for all methods considered [Sec. 3.13].

As anticipated, Fig. 3.7 shows that CD driving performs better than adiabatic driving, and the two agree in the limit of large $T$. Moreover, we see explicitly that the CD and QAOA solutions are far from the adiabatic regime. Not surprisingly, CD driving outperforms conventional QAOA for small $T$, as it can increase the values of the variational parameters (and with it the norm) indefinitely. However, CD-QAOA

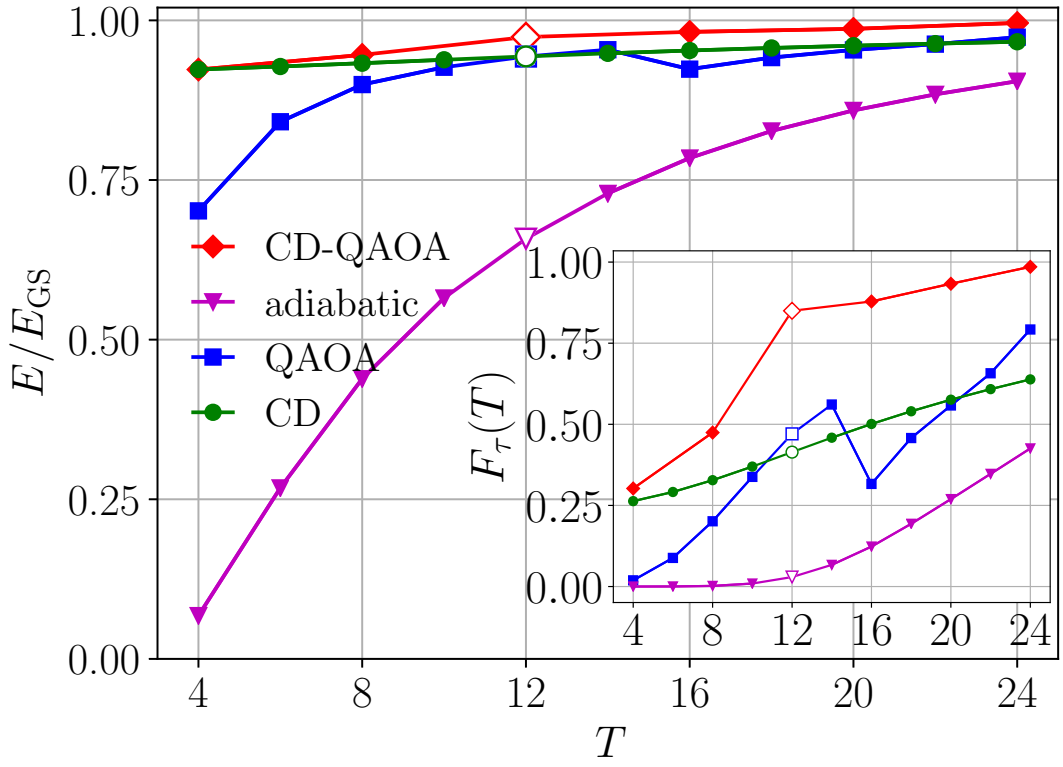



Figure 3.7: Spin-1 Ising model: energy minimization and the corresponding many-body fidelity [inset] against different protocol duration $T$ for four different optimization methods: CD-QAOA (red line), conventional QAOA (blue line), variational gauge potential (green) and adiabatic evolution (magenta). The empty symbols mark the duration for which the evolution of physical quantities is shown in Fig. 3.24. The initial and target states are $|\psi_i\rangle = |\downarrow \cdots \downarrow\rangle$ and $|\psi_*\rangle = |\psi_{\mathrm{GS}}(H)\rangle$ for $h_z/J = 0.809$ and $h_x/J = 0.9045$. The alternating unitaries for conventional QAOA are generated by $\mathcal{A}_{\mathrm{QAOA}} = \{H_1, H_2\}$ [cf. Eq. equation 4.9.1]; for CD-QAOA, we extend this set using adiabatic gauge potential terms to $\mathcal{A}_{\mathrm{CD-QAOA}} = \{H_1, H_2; Y, XY, YZ, X|Y, Y|Z\}$. The variational gauge potential in CD driving uses all five imaginary-valued gauge potentials $\{Y, XY, YZ, X|Y, Y|Z\}$. The CD- and adiabatic driving simulations are both based on the smooth protocol function $\lambda(t) = \sin^2\left(\frac{\pi t}{2T}\right)$, with a time-discretization step $\Delta t = 0.2$. The value of $q = 20$ and the size of sequence space is $|\mathcal{A}|(|\mathcal{A}|-1)^{q-1} \approx 10^{15}$. The system size is $N = 8$, where $\dim(\mathcal{H}) = 498$.

| $T$ | $E/E_{\mathrm{GS}}$ $[N=10]$ | | | |
|---|---|---|---|---|
| | CD-QAOA | CD | QAOA | adiabatic |
| 4 | **0.943837** | 0.923199 | 0.79534 | 0.067807 |
| 8 | **0.961383** | 0.933067 | 0.93386 | 0.438856 |
| 12 | **0.990415** | 0.942857 | 0.96275 | 0.658182 |

Table 3.2: Spin-1 Ising model: comparison of the best obtained energy ratio $E/E_{\mathrm{GS}}$ after optimization, for four different optimization methods: CD-QAOA, variational CD driving, conventional QAOA, and adiabatic evolution, at $T = 4, 8, 12$ for $N = 10$ qutrits, where $\dim(\mathcal{H}) = 3219$. The remaining setup and parameters are the same as in Fig. 3.7.

consistently outperforms CD driving in the entire $T$-range; the contrast is especially pronounced in the many-body fidelity [Fig. 3.7, inset]. CD-QAOA makes use of the variational power of QAOA, combining it with physics-motivated input from CD driving.

Table 3.2 shows a comparison with the best obtained energies for $N = 10$ spin-1 particles (qutrits): the superior performance of CD-QAOA remains despite the exponentially growing Hilbert space size. Reaching significantly larger system sizes is infeasible with the present-day computational power: we note that this a feature of the quantum system rather than a drawback of CD-QAOA, cf. discussion on LMG model in Sec. 3.4.

We emphasize that CD-QAOA features some important advantages as compared to CD driving: (1) Due to the nested commutators in the definition of time-ordered exponentials, the QAOA dynamics can effectively implement total unitaries $U(\{\alpha_j\}_{j=1}^{q}, \tau)$ generated by effective non-local operators; therefore, CD-QAOA can, in principle, realize a nonlocal effective Hamiltonian as an approximation to the true CD Hamiltonian, thereby overcoming convergence issues related to operator-valued series expansions. (2) CD-QAOA lifts the boundary constraint present in adiabatic and CD driving where the initial and target Hamiltonians are eigenstates of $H(0)$ and $H(1)$, respectively; an interesting open question is whether a local effective Hamiltonian exists, which captures the evolution of the system in this case. Examining the evolution of the entanglement entropy and other local observables induced by the optimal protocol, suggests that this is indeed the case [Sec. 3.13]. (3) One can add any control unitary to the set $\mathcal{A}$, not just terms related to gauge potentials: CD-QAOA has high flexibility to accommodate experimental constraints. (4) determining the variational gauge

potential in CD-driving requires using the exact ground state in order to minimize the action, which can be a significant drawback when the ground state is not known or cannot be computed.

## 3.6 Transfer Learning and Generalization of the RL Algorithm to different System Sizes

The scale collapse in the energy density of the spin-1/2 Ising model presents a testbed for the transfer learning capabilities of RL. In transfer learning, the RL agent learns to control one physical system, and is then used to manipulate another. In our case, the two systems are given by the same Ising model at two different system sizes. Note that transfer learning would have not been possible, had we defined the learning problem using the full quantum states, because the latter are vectors in Hilbert space whose size grows exponentially with $N$.

To apply transfer learning, consider first a fixed protocol duration $T$. For every fixed system size $N$, we first train a different RL agent. Next, we build the set of protocols across all system sizes, found by these agents, and determine the number of unique protocols [cf. legend in Fig. 3.8]. Finally, we apply all unique protocols to all system sizes available, and store the energy densities they result in. This leaves us with a set of energy density values for every fixed $T$. The error bars in Fig. 3.8 show the best and the worst protocols over this set. Observe that, below the QSL, there are only a few points $T$ where the best control protocol is the same across all system sizes. Transfer learning works well, as can be seen by the small error bars. In this regime, the RL agent generalizes its knowledge and learns universal features of the protocol, required to control the Ising model. In contrast, for $T > T_{\mathrm{QSL}}$, there are many more protocols giving approximately similar ground state energies. While the corresponding energies are similar in value, the agent does not generalize. Nevertheless, we checked that, in this regime, training on smaller system sizes still provides a useful pre-training procedure for learning on larger systems.

## 3.7 Discussion/Outlook

We analyzed many-body ground state preparation using unitary evolution in the spin-1/2 Ising model, the spin-1 anisotropic Heisenberg and Ising models, and the fully connected LMG spin-1/2 model. We introduced the CD-QAOA ansatz: an RL agent optimizes the order of unitaries in the protocol sequence, generated from terms in the adiabatic gauge potential series, and obtains short high-fidelity protocols away from

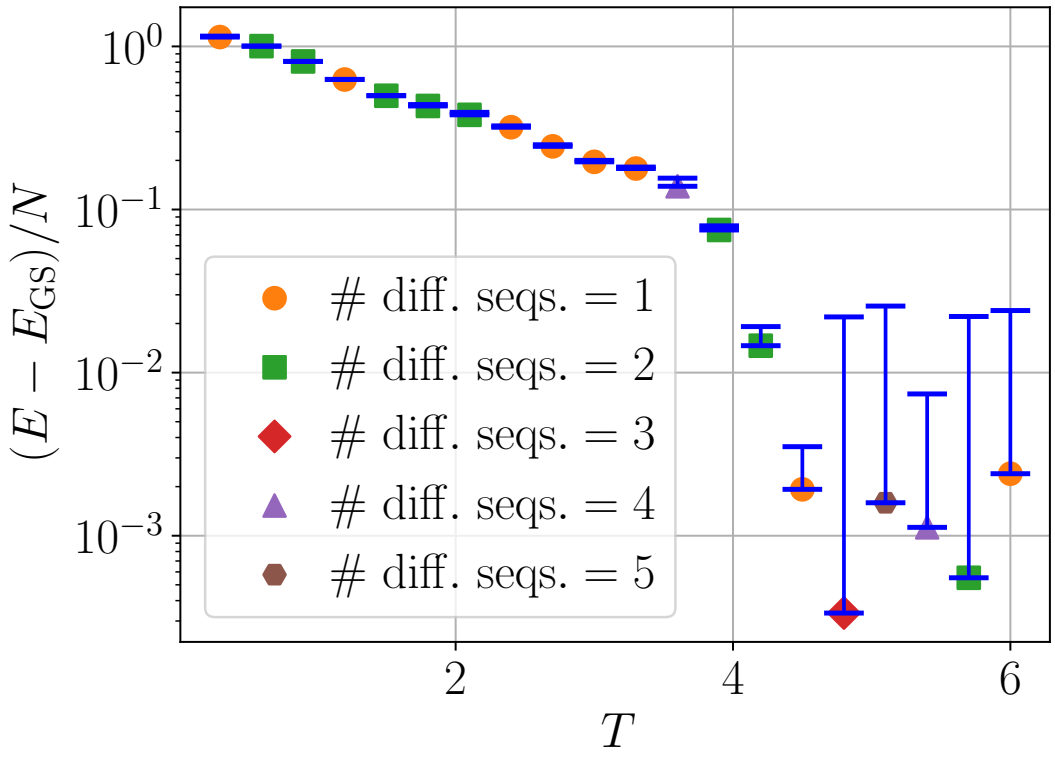



Figure 3.8: Spin-1/2 Ising model: Protocol generalization across various system sizes. The marker types show the number of different protocols found by the RL agent at a fixed $T$ across all system sizes $N = 6, 10, 12, 14, 16$ and 18. Each protocol is applied to every system size $N$ at a fixed $T$ which results in a set of cost function values; the error bars designate the range between the largest and smallest cost function value. The parameters are the same as in Fig. 3.1.

the adiabatic regime. The resulting algorithm combines the strength of continuous and discrete optimization into a unified and versatile control framework. We find that our CD-QAOA ansatz outperforms consistently both conventional QAOA, and variational CD driving across different models and protocol durations. An interesting open question is whether one can use CD-QAOA to find a nonlocal approximation to the variational gauge potential itself, which is beyond the scope of asymptotic series expansions. Another straightforward application of CD-QAOA would be imaginary time evolution [24].

For the nonintegrable spin-1/2 Ising chain, we reveal the existence of a finite quantum speed limit. Moreover, we find a remarkable system-size collapse of the energy

curves suggesting that the sequences found by the agent hold in the thermodynamic limit; this is corroborated by numerical experiments on transfer learning which demonstrate that one can train the agent on one system size while it generalizes to larger systems. In the Heisenberg spin-1 system, CD-QAOA allows preparing long-range and topologically ordered ground states, even when the initial state does not belong to the phase of the target state. The optimal protocols found by the RL agent contain nontrivial basis rotations, intertwined with alternating QAOA-like subsequences, suggesting new ansätze for more efficient variants of CD-QAOA. Numerical studies of nonequilibrium quantum many-body systems, in turn, suffer from limitations related to the exponentially large dimension of the underlying Hilbert space: future work can investigate dynamics beyond exact evolution.

Compared to conventional QAOA, using terms from the variational gauge potential series has higher expressivity, which results in much shorter, yet better performing, circuits. This method can be used, e.g., to reduce the cumulative error in quantum computing devices. However, gauge potential terms are not always easy to realize in experiments since they implement imaginary-valued terms which break time-reversal symmetry; that said, it is often possible to generate such terms using auxiliary real-valued operators via a generalization of the Euler angles, or by means of change-of-frame transformations [280]. Moreover, as we have demonstrated, CD-QAOA admits non-gauge potential terms as building blocks for control sequences, e.g., universal gate sets. Other experimental constraints, such as the presence of drift terms, which cannot be switched off, can also be conveniently incorporated by redefining the set of unitaries $\mathcal{A}$.

Finally, let us remark that RL provides only one possible set of algorithms to explore the exponentially large space of protocol sequences; in practice, one can apply other discrete optimization techniques, e.g. genetic algorithms and search algorithms like Monte-Carlo Tree Search (MCTS).

## 3.8 High-level optimization: Policy Gradient using Deep Autoregressive Networks

Recently, progress made in machine learning (ML) [98, 215, 55, 56] has raised the question as to how we can harness such modern advances to improve techniques to manipulate quantum systems. Examples of ML applications include model-based optimization [293], differentiable programming [273] and Bayesian inference [271] quantum control, cavity control [105], designing quantum end-to-end learning schemes [336] and measurement-based adaptation protocols [4], as well as applications to quantum

error-correction [107, 228].

Reinforcement learning (RL) algorithms [295, 265], such as policy gradient [237, 19, 249], Q-learning [48, 50] and AlphaZero [78], have recently attracted the attention of physicists, and in particular how they can be combined with physically motivated VQEs for improved performance. In RL, policy gradient has been proposed as an alternative optimizer for QAOA showcasing the robustness of RL-based optimization to both classical and quantum sources of noise [346]; a related study applied Proximal Policy Optimization (PPO) to prepare the ground state of the transverse-field Ising model [325]. The QAOA ansatz with policy gradient has been applied to efficiently find optimal variational parameters for unseen combinatorial problem instances on a quantum computer [160]; Q-learning was used to formulate QAOA into an RL framework to solve difficult combinatorial problems [110], and in the context of digital quantum simulation [37].

In the following, we introduce the details of the Reinforcement Learning algorithm used for the high-level optimization in this work.

## Reinforcement Learning Basics

Reinforcement learning (RL) comprises a class of machine learning algorithms where an agent learns how to solve a given task through interactions with its environment using a trial-and-error approach [295]. It is based on a Markov Decision Process (MDP) defined by the tuple $(\mathcal{S}, \mathcal{A}, p, R)$ where $\mathcal{S}$ and $\mathcal{A}$ represent the state and action spaces, $p : \mathcal{S} \times \mathcal{S} \times \mathcal{A} \to [0, 1]$ defines the transition dynamics, and $R : S \times A \to \mathbb{R}$ is the reward function that describe the environment. Let $\pi(a_j|s_j) : \mathcal{A} \times \mathcal{S} \to [0, 1]$ denote a stochastic policy that defines the probability distribution of choosing an action $a_j \in \mathcal{A}$ given the state $s_j \in \mathcal{S}$. Rolling out the policy $\pi(a_j|s_j)$ in the environment can also be viewed as sampling a trajectory $\tau \sim \mathbb{P}^\pi(\cdot)$ from the MDP, where $\mathbb{P}^\pi(\tau) = p_0(s_1)\pi(a_1|s_1)p(s_2|s_1, a_1) \cdots \pi(a_q|s_q)p(s_{q+1}|s_q, a_q)$ is the probability for the trajectory $\tau$ to occur, $q$ sets the episode or trajectory length, and $p_0$ is the distribution of the initial state; an example for a trajectory is $\tau = (s_1, a_1, ...., a_q, s_{q+1})$. The goal in RL is to find a policy that maximizes the expected return:

$$J(\boldsymbol{\theta}) = \mathbb{E}_{\tau\sim\mathbb{P}^\pi}\left[\sum_{j=1}^{q} R(s_j, a_j)\right]. \tag{3.8.1}$$

To maximize the expected return $J(\boldsymbol{\theta})$, we use policy gradient – an RL algorithm, which is (i) on-policy, i.e. trajectories have to be sampled from the current policy $\pi_{\boldsymbol{\theta}}$: $\pi = \pi_{\boldsymbol{\theta}}$, and (ii) model-free, i.e. the agent does not need to have a model for

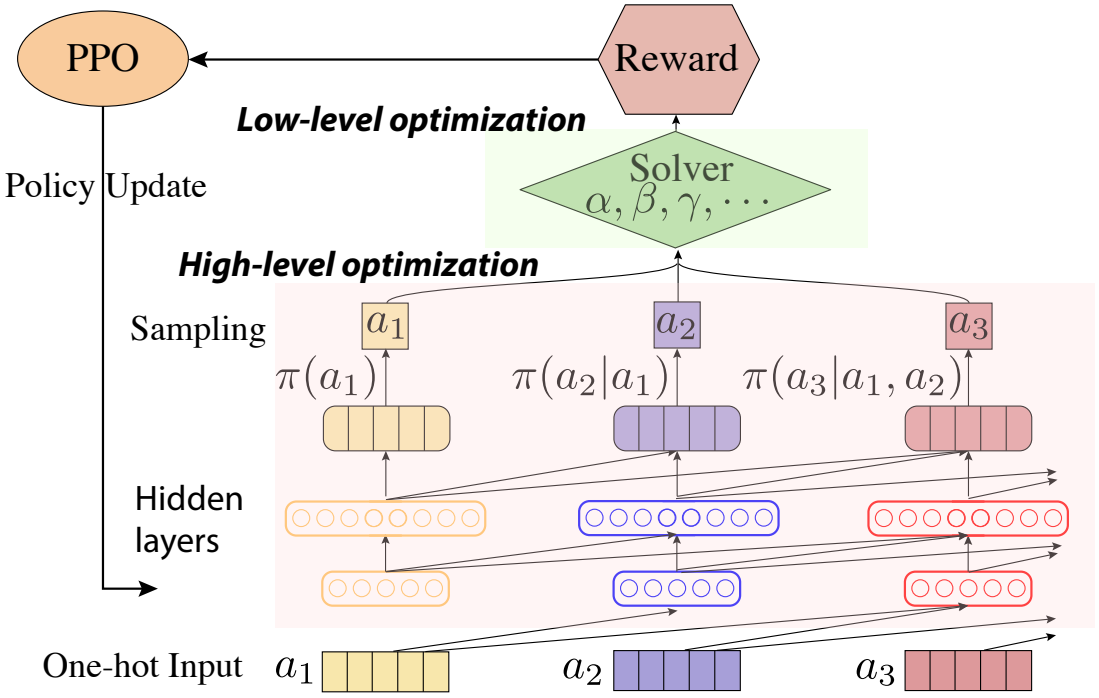



Figure 3.9: Schematics of CD-QAOA with an autoregressive policy network. The ancestral sampling procedure used for training is displayed in Fig. 3.10. The details of the network structure and its training hyperparameters are shown in Table 4.2.

the environment dynamics: $p(s'|s, a)$ is assumed unknown for the purpose of finding the optimal policy. Highly expressive function approximators, such as deep neural networks, help parametrize the policy using variational parameters $\boldsymbol{\theta}$. Policy gradient gradually improves the expected return in a number of iterations (or training episodes), by increasing the probability for actions that lead to higher rewards, and decreasing the probability for actions that lead to lower rewards, until it reaches a (nearly) optimal policy.

We mention in passing that we use interchangeably the terms return and cost function (the latter being the negative of the former): the goal of the RL agent is thus to maximize the expected return, or to minimize the cost function.

Figure 3.10: The exact sampling algorithm for CD-QAOA with an autoregressive policy network, where faded nodes and connections represent unused nodes and connections. The action at each time step is generated sequentially, by computing its respective conditional categorical distribution, and sampling according to that. Notice that only a single column is processed at each time step, and in order to sample a complete sequence of actions in an episode one needs to make a forward pass through the network architecture $q$ times.

## Policy Gradient Reinforcement Learning for Quantum Many-Body Systems

**Actions**: To apply the reinforcement learning formalism to quantum control, we identify taking actions at each time step within a learning episode, with selecting unitaries one at a time within the circuit depth $q$. Choosing the same unitary at two consecutive time steps is prohibited because the same actions can be merged resulting in a lower effective circuit depth $q-1$. At the initial time step $j=1$, the quantum wavefunction is given by the initial state $|\psi_i\rangle$; for each intermediate protocol step $j$, the action $a_j = H_j$ is chosen according to the policy $\pi_{\boldsymbol{\theta}}$. Note that the RL agent only selects the generator $H_j$ out of the set of available actions $\mathcal{A}$ (or alternatively – which unitary to apply). In other words, unlike Ref. [346], the RL part of CD-QAOA is *not* concerned with finding the corresponding optimal duration $\alpha_j$; one can think of this low-level continuous optimization as being part of the environment [cf. Sec. 3.9][9]. At the end of the episode, the quantum state is evolved by applying the entire generated circuit $U(\{\alpha_j\}_{j=1}^{q}, \tau)$ to the initial quantum state $|\psi_i\rangle$.

**States**: Since the initial state $|\psi_i\rangle$ is fixed and thus the quantum state at any time step $j$ is uniquely determined by the previous actions taken, here we represent the RL state by concatenating all the previous actions up to step $j$ [44]. One reason for this is that, in many-body quantum systems, the number of components in the quantum state scales exponentially with the system size $N$, which quickly leads to a computational bottleneck for the simulation on classical computers. A second advantage of this choice is that the first layer of the underlying deep neural network architecture, which parametrizes the policy, will not depend on the system size $N$ either, which allows the algorithm to handle a large number of degrees of freedom. Using the quantum state would not be viable on quantum computers either, because quantum states are unphysical mathematical constructs that cannot be measured. Therefore, we can simplify the form of trajectories to consist of actions only, e.g. $\tau = (a_1, a_2, \ldots, a_q)$.

[9] It is also possible to define an RL framework for hybrid continuous-discrete control where optimization is entirely based on RL, cf. Ref. [351].

Rewards: The reward $R_j = R(s_j, a_j)$ is chosen as the negative energy density at the end of the episode:

$$R_j = \begin{cases} 0, & \text{if } j < q \\ -E(\{\alpha_j\}_{j=1}^{q}, \tau)/N, & \text{if } j = q. \end{cases}$$

We use energy density, since it is an intensive quantity which has a well-defined limit when increasing the number of particles $N$. In all figures, we show the relative energy $E/E_{\mathrm{GS}}$ for clarity (the ground state energy $E_{\mathrm{GS}}$ is typically negative in our models), but the RL agent is always trained with the (negative) energy density $-E/N$. Rewards can also be other observables or nonobservable quantities, such as the overlap squared between two quantum states (fidelity), or the entanglement entropy.

Notice that the reward is sparse: only at the end of the episode is the negative energy density given as a reward; there is no instantaneous reward during the sequence [and thus we can use interchangeably the terms reward and total return]. This is motivated by the quantum nature of the control problem, where a projective measurement results in a wavefunction collapse.

## Policy Parametrization using an Autoregressive Neural Network

An essential part of the policy gradient algorithm is the definition of the policy $\pi_{\boldsymbol{\theta}}$. It is common to parametrize the policy with a highly expressive function approximator, such as a neural network. In our setup, we use a deep autoregressive network, which has recently been used in physics applications of learning to generate samples from free energies in statistical mechanics models [338], and variational approximators for quantum many-body states [281]. This architecture is selected to incorporate causality by factorizing the total probability into a product of conditional probabilities:

$$\pi_{\boldsymbol{\theta}}(a_1, a_2, \cdots, a_q) = \pi_{\boldsymbol{\theta}}(a_1) \prod_{j=2}^{q} \pi_{\boldsymbol{\theta}}(a_j | a_1, \cdots, a_{j-1}), \tag{3.8.2}$$

where marginal distribution $\pi_{\boldsymbol{\theta}}(a_1)$ and conditional distribution $\pi_{\boldsymbol{\theta}}(a_j | a_1, \cdots, a_{j-1})$ are discrete categorical distributions over $\mathcal{A}$. This kind of parametrization explicitly tells how the actions taken in the earlier steps of an episode affect the actions selected later on during the same episode. Such a causal requirement would not be necessary, had we used the full quantum state, which would make the dynamics of the environment Markovian. Each of the conditional probabilities in Eq. equation 3.8.2 can be modeled explicitly using the autoregressive neural network architecture, which

naturally allows the policy to depend on all the previous actions only. The structure of the policy network is shown in Fig. 3.9, the sampling of the autoregresssive policy is depicted in Fig. 3.10 and the hyperparameters of the algorithm (including the number of parameters) are given in Table 4.2.

## Training Procedure: Proximal Policy Optimization (PPO)

In each iteration of the policy gradient algorithm, a batch of sampled trajectories $\{\tau^k\} = \{(a_1^k, \cdots, a_q^k)\}_{k=1}^M$ are rolled out (i.e. sampled) from the current policy, where $M$ is the batch/sample size. Then, the return $R(\tau^k)$ corresponding to trajectory $\tau^k$ is computed as

$$R(\tau^k) = \sum_{j=1}^{q} R_j^k = -E(\{\alpha_j^k\}_{j=1}^q, \tau^k)/N.$$

To compute the energies, we use the low-level optimization to determine the best-estimated values of $\alpha_j$, given a sequence $\tau$, see Sec. 3.9. To minimize the chance of getting stuck in a suboptimal local minimum, each sequence is evaluated multiple times, starting from a different initial realization for the $\alpha_j$-optimizer, and the best result is selected [Sec. 3.11].

For every iteration, we can define three quantities for a fixed batch of samples: (i) mean reward (over the current batch), (ii) max reward (over the current batch), and (iii) history best (best-encountered reward over all the previous iterations). These quantities measure the performance of the learned policy, and are shown in Fig. 4.4. Figure 3.12 shows the scaling of these quantities for the spin-1 Ising chain, as a function of the episode length $q$. The performance of CD-QAOA increases because the action space for a larger value of $q$ always contains as a subset the action space for a smaller $q$.

In order to improve the policy represented by the autoregressive network, the RL algorithm interacts with the quantum environment by querying the reward for samples from the current policy. Each trajectory is assigned a reward, once the simulation of the quantum dynamics is complete [note that, as of present date, the simulation may be more expensive if evaluated on a quantum computer]. Thus, it is advantageous to reduce the sample size needed to learn the policy, i.e., to improve the sample efficiency.

The vanilla policy gradient method is known for its poor data efficiency. Thus, we adopt Proximal Policy Optimization (PPO) [278], a more robust and sample-efficient policy gradient type algorithm. To be more specific, we use the following clipped

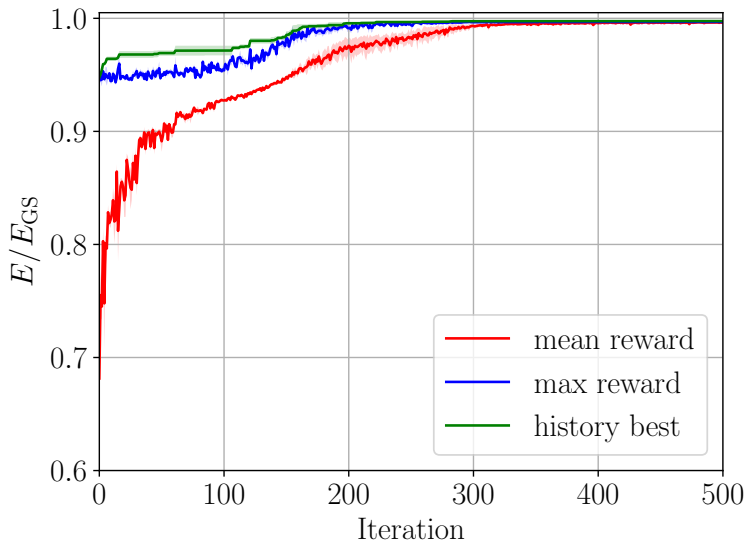



Figure 3.11: Spin-1 Ising model: training curves for CD-QAOA with energy minimization as a cost function. The mean negative energy density (red) is computed for a sample generated using the policy at the current iteration; max (blue) is the maximum within the sample; the history best (green) is the best-encountered policy during the entire training process (i.e., considering all iterations). Each curve shows the average out of three simulations corresponding to three different seed values for the high level RL optimization; the fluctuations around the seed-averages are shown as a narrow shaded area. The total duration is $T = 28$ and the number of spin-1 particles is $N = 8$. The initial and target states are $|\psi_i\rangle = |\downarrow \cdots \downarrow\rangle$ and $|\psi_*\rangle = |\psi_{\mathrm{GS}}(H)\rangle$ for $h_z/J = 0.809$ and $h_x/J = 0.9045$. The CD-QAOA action space is $\mathcal{A}_{\text{CD-QAOA}} = \{Z|Z+X, Z; Y, XY, YZ, X|Y, Y|Z\}$, and we use $q = 20$.

objective function:

$$\mathcal{G}(\boldsymbol{\theta}) = \mathbb{E}_{\tau\sim\pi_{\boldsymbol{\theta}_t}}\Big[\quad \min\big\{\rho_{\boldsymbol{\theta}}(\tau)A_{\boldsymbol{\theta}_t}(\tau), \qquad \text{clip}\left(\rho_{\boldsymbol{\theta}}(\tau), 1-\epsilon, 1+\epsilon\right) A_{\boldsymbol{\theta}_t}(\tau)\big\}\Big]. \tag{3.8.3}$$

Here, $\tau = (a_1, a_2, \cdots, a_q)$ is the action sequence sampled from the previous policy $\pi_{\boldsymbol{\theta}_t}$ [cf. Algorithm 4]. Typically, the policy from the last iteration is chosen to be the old policy; $\rho_{\boldsymbol{\theta}}(\cdot) = \frac{\pi_{\boldsymbol{\theta}}(\cdot)}{\pi_{\boldsymbol{\theta}_t}(\cdot)}$ is the importance sampling weight between the new

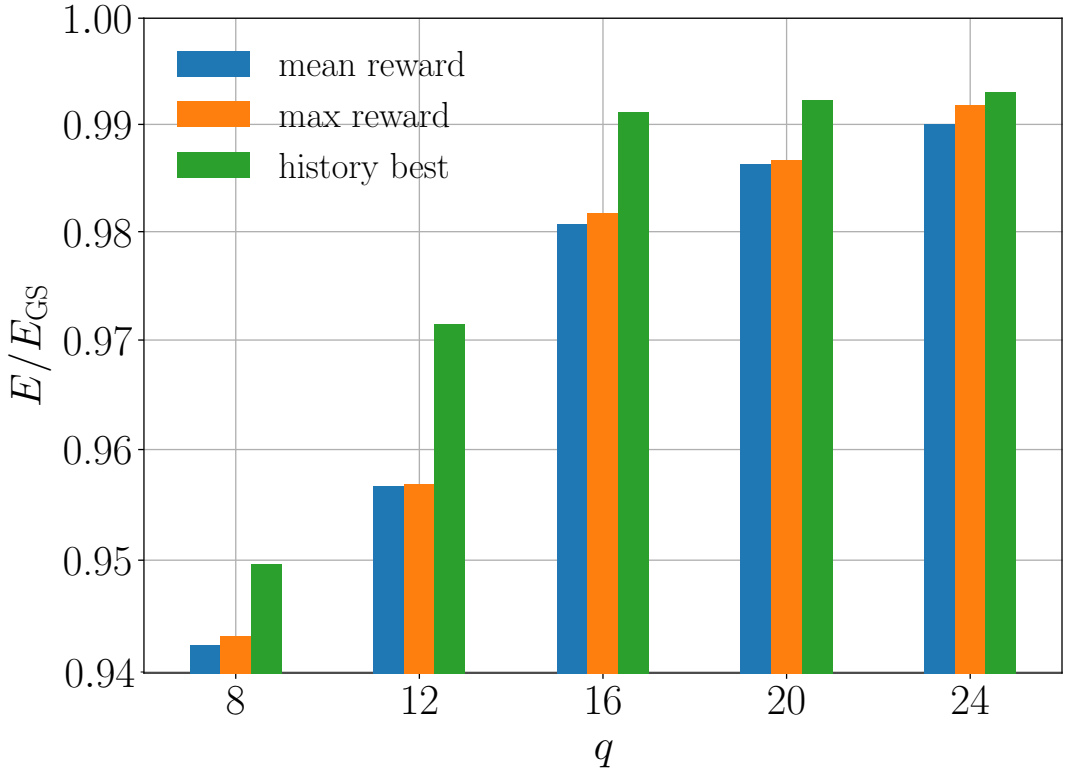



Figure 3.12: Spin-1 Ising model: energy minimization against different circuit depths $q$ using CD-QAOA. The mean negative energy density (blue) is computed for a sample generated using the final, learned policy; max (orange) is the maximum within the sample; the history best (green) is the best encountered policy during the entire training process (i.e., considering all iterations). The total duration $T=20$ and the values of $q$ ranges from 8 to 24. The other model parameters are the same as in Fig. 4.4.

policy $\pi_{\boldsymbol{\theta}}$ and the old policy $\pi_{\boldsymbol{\theta}_t}$; $A_{\boldsymbol{\theta}_t}(\tau) = R(\tau) - b$ is the advantage function, where $b$ is called a baseline: the advantage measures the reward gain of choosing a specific action, w.r.t. the baseline. For example, a simple baseline can be the average reward, e.g., $b = \mathbb{E}_{\tau\sim\pi_{\boldsymbol{\theta}_t}}[R(\tau)]$, and then the advantage measures how much better (or worse) an action is w.r.t. the average; in the numerical experiments, we use an exponential moving average [cf. Sec. 3.8 for the details].

Further, the clip function,

$$\text{clip}(r, x, y) = \max\big(\min(r, x), y\big),$$

clips the value of $r$ within the interval $[x, y]$, which is used to restrict the likelihood ratio in the range $[1-\epsilon, 1+\epsilon]$; this prevents the policy update from deviating too much from the old policy after one gradient update. The clipped objective function is designed to improve the policy as well as to keep it within some vicinity of the last iteration, whence the name Proximal Policy Optimization.

We update the network parameters $\boldsymbol{\theta}$ by ascending along the gradient of the RL objective $\mathcal{G}(\boldsymbol{\theta})$. To provide intuition about the PPO objective, consider the following limiting case. If we only have the first term in the objective, i.e. $\mathcal{G}_1(\boldsymbol{\theta}) = \mathbb{E}_{\tau\sim\pi_{\boldsymbol{\theta}_t}}[\rho_{\boldsymbol{\theta}}(\tau)A_{\boldsymbol{\theta}_t}(\tau)]$, we obtain the following gradient:

$$\begin{aligned}\nabla_{\boldsymbol{\theta}}\mathcal{G}_1(\boldsymbol{\theta}) &= \mathbb{E}_{\tau\sim\pi_{\boldsymbol{\theta}_t}}[\nabla_{\boldsymbol{\theta}}\rho_{\boldsymbol{\theta}}(\tau)A_{\boldsymbol{\theta}_t}(\tau)] \\ &= \mathbb{E}_{\tau\sim\pi_{\boldsymbol{\theta}_t}}\left[\frac{\nabla_{\boldsymbol{\theta}}\pi_{\boldsymbol{\theta}}(\tau)}{\pi_{\boldsymbol{\theta}_t(\tau)}}A_{\boldsymbol{\theta}_t}(\tau)\right].\end{aligned}$$

Since we are taking the gradient with respect to $\boldsymbol{\theta}$, it will pass through $\pi_{\boldsymbol{\theta}_t}$ and $A_{\boldsymbol{\theta}_t}(\tau)$. Furthermore, whenever the parameters $\boldsymbol{\theta} \approx \boldsymbol{\theta}_t$, the gradient above is identical to the policy gradient:

$$\begin{aligned}\nabla_{\boldsymbol{\theta}}\mathcal{G}_1(\boldsymbol{\theta}) &\approx \mathbb{E}_{\tau\sim\pi_{\boldsymbol{\theta}}}\left[\frac{\nabla_{\boldsymbol{\theta}}\pi_{\boldsymbol{\theta}}(\tau)}{\pi_{\boldsymbol{\theta}(\tau)}}A_{\boldsymbol{\theta}}(\tau)\right] \\ &= \mathbb{E}_{\tau\sim\pi_{\boldsymbol{\theta}}}[\nabla_{\boldsymbol{\theta}}\log\pi_{\boldsymbol{\theta}}(\tau)A_{\boldsymbol{\theta}}(\tau)].\end{aligned}$$

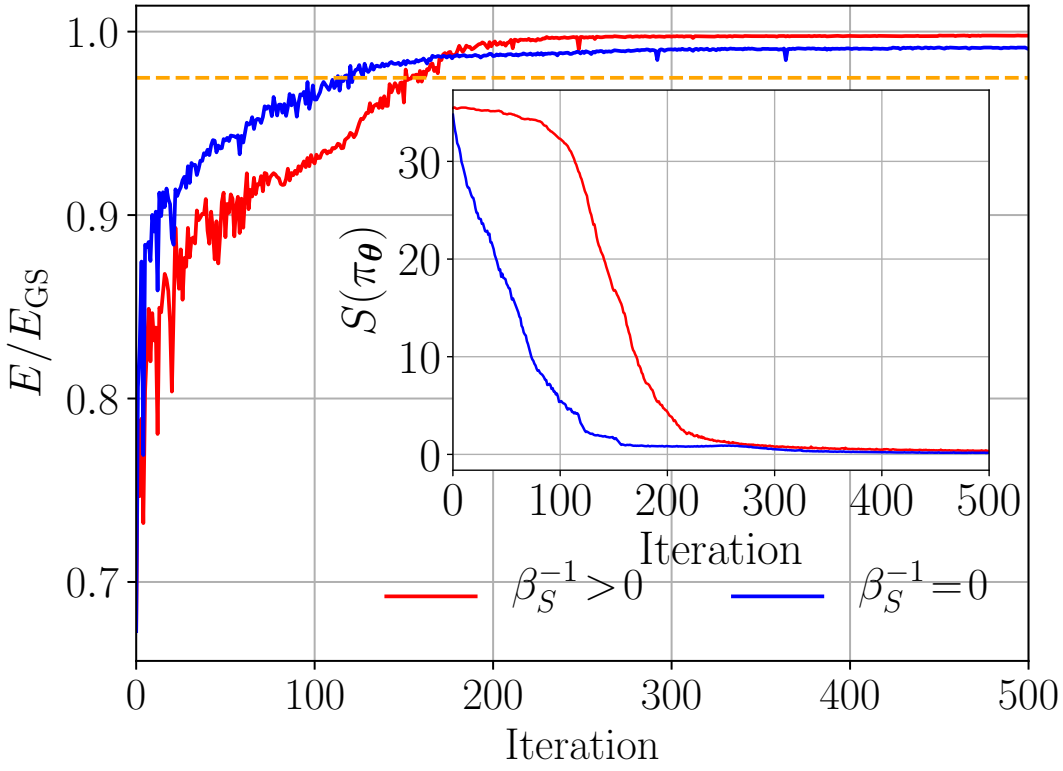



Figure 3.13: Spin-1 Ising model: Comparison of the mean reward with ($\beta_{S,\{0\}}^{-1} = 0.1$) and without ($\beta_S^{-1} = 0$) the entropy bonus during training. For comparison, the dashed horizontal line marks the performance of QAOA. The inset shows the evolution of the policy information entropy during training. Adding entropy gives more room for the RL agent to explore the space of policies instead of directly exploiting the knowledge it obtains. As becomes clear from the figure, the RL algorithm with the entropy bonus achieves a better final performance at the end of training, at the cost of suffering an intermediate lower reward at the beginning of training. The simulation parameters are the same as in Fig. 4.4.

However, PPO performs multiple gradient updates on the sampled data, rendering policy learning more sample efficient [278].

**Incentivizing Exploration using Entropy**

Maintaining a balance between exploration and exploitation is another major challenge for the reinforcement learning algorithm. Too much exploration prevents the agent from adopting the best strategy it knows so far; on the contrary, too much exploitation limits the agent from attempting new actions and achieving a potentially higher reward. Therefore, it is more appropriate for the agent to explore substantially in the initial iterations of the training procedure, and to gradually switch over to exploitation towards the end of the training procedure.

In order to incentivize the agent to explore the action space at the beginning of training, we include an entropy 'bonus' [366, 127] to the PPO objective from Eq. equation 3.8.3. To do this, consider the maximal-entropy objective, where the agent aims to maximize the sum of the total reward and the policy entropy $S$ [cf. Eq. equation 3.8.5]:

$$\begin{aligned}\mathcal{J}(\boldsymbol{\theta}) &= \mathcal{G}(\boldsymbol{\theta}) + \beta_S^{-1}\mathcal{S}(\pi_{\boldsymbol{\theta}}) \\ &= \mathbb{E}_{\tau=(a_1,\cdots,a_q)\sim\pi_{\boldsymbol{\theta}_t}}\Bigg[\min\{\rho_{\boldsymbol{\theta}}(\tau)A_{\boldsymbol{\theta}_t}(\tau),\ \mathrm{clip}\left(\rho_{\boldsymbol{\theta}}(\tau), 1-\epsilon, 1+\epsilon\right)A_{\boldsymbol{\theta}_t}(\tau)\} \\ &\qquad + \beta_S^{-1}\sum_{j=1}^{q}\mathcal{S}\big(\pi_{\boldsymbol{\theta}}(\,\cdot\,|a_1,\cdots,a_{j-1})\big)\Bigg],\end{aligned} \tag{3.8.4}$$

where $\mathcal{S}\big(\pi_{\boldsymbol{\theta}}(\,\cdot\,|a_1,\cdots,a_{j-1})\big) \equiv \mathcal{S}\big(\pi_{\boldsymbol{\theta}}(\,\cdot\,)\big)$, for $j=1$. The trade-off between exploration and exploitation is controlled by the coefficient $\beta_S^{-1}$, which carries a meaning analogous to temperature in statistical mechanics: for $\beta_S^{-1}\to 0$ (or $\beta_S\to\infty$), any exploration is limited to the intrinsic probabilistic nature of the policy; if training is successful, it is expected that, for deterministic environments, the policy eventually converges to a delta distribution (over the action space) at the later training iterations; this may deteriorate exploration and learning. However, in the opposite limit, $\beta_S^{-1}\to\infty$ (or $\beta_S\to 0$), every action is selected with equal probability, and the values of the policy $\pi$ become irrelevant. Therefore, in practice, we use a decay schedule for the inverse temperature $\beta_S^{-1}$ to gradually reduce exploration [see Sec. 3.8].

Since the marginal distribution $\pi_{\boldsymbol{\theta}}(\,\cdot\,)$ and conditional distribution $\pi_{\boldsymbol{\theta}}(\,\cdot\,|a_1,\cdots,a_{j-1})$ are discrete categorical distributions over $\mathcal{A}$, we can compute a closed form expression for the entropy of the categorical distribution policy. For trajectory $\tau^i=(a_1^i,\cdots,a_q^i)$,

the $j$-th term in the entropy bonus simplifies to

$$\begin{aligned}&\mathcal{S}\big(\pi_{\boldsymbol{\theta}}(\cdot|a_1^i,\cdots,a_{j-1}^i)\big) \\ &=-\sum_{a\in\mathcal{A}}\pi_{\boldsymbol{\theta}}(a|a_1^i,\cdots,a_{j-1}^i)\log\pi_{\boldsymbol{\theta}}(a|a_1^i,\cdots,a_{j-1}^i).\end{aligned} \tag{3.8.5}$$

We emphasize that the entropy considered here is the Shannon or information entropy associated with the policy as a probability distribution, and should be contrasted with the thermodynamic entropy, associated with the logarithm of the density of protocol configurations (a.k.a. density of states) in the optimization landscape. The Shannon entropy helps exploration in the space of policies, and thus the annealing of the corresponding Lagrange multiplier, $\beta_S^{-1}$, is not related to thermal annealing in the optimization/energy landscape in a straightforward manner. Moreover, notice that the policy optimization is part of the classical postprocessing of the quantum data, i.e., it does not compromise the nature of the quantum data which is fed to the algorithm in form of rewards.

Figure 3.13 shows a comparison of PPO with and without entropy, as controlled by the value of the temperature $\beta_S^{-1}$. Introducing the policy information entropy keeps the policy a bit broader in the initial stages of training which enhances exploration; towards the end of training the information entropy is not needed: therefore, we gradually "anneal" $\beta_S^{-1}$, cf. Sec. 3.8.

## Technical Details

We train the CD-QAOA algorithm for 500 epochs/iterations with a mini-batch size of $M = 128$. Throughout the training, we sample trajectories according to the marginal and conditional policy distributions given by the autoregressive network.

We use Adam to perform gradient descent on the objective in Eq. equation 4.3.6 with the default parameters $\beta_1 = 0.9$ and $\beta_2 = 0.999$, which define the exponential decay rate for the first and second moment estimates, respectively. The learning rate is initialized as $\alpha_{\{\mathrm{lr},0\}} = 0.01$ and decays by a factor of 0.96 every 50 steps in a staircase fashion. To be more precise, the learning rate at the $k$-th iteration with the exponential decay reads as $\alpha_{\mathrm{lr},\{k\}} = 0.01 \cdot 0.96^{\lfloor * \rfloor k/50}$. The subscript $\{k\}$ denotes the iteration/episode number.

We also introduce an exponential decay schedule for the pre-factor [a.k.a. temperature] $\beta_S^{-1}$ of the entropy bonus from Eq. equation 4.3.6. The temperature initializes at $\beta_{S,\{0\}}^{-1} = 0.1$ and decays by a factor of 0.9 every 10 steps. At the $k$-th iteration, the temperature is $\beta_{S,\{k\}}^{-1} = 0.1 \cdot 0.9^{k/10}$. Eventually, the temperature is annealed to zero.

We estimate the advantage function by $A_{\boldsymbol{\theta}_{\text{old}}}(\tau) = R(\tau) - b$, where $b$ is the baseline used to reduce the variance of the estimation. Our baseline $b$ uses an exponential moving average (EMA) of the previous rewards. EMA stabilizes the training and also leverages the past reward information to form a lagged baseline. In practice, we find that the RL algorithm can achieve better rewards compared with using the average of current samples as the baseline. To be more specific, the exponential moving baseline update is $b_{\{k\}} = \eta b_{\{k-1\}} + (1-\eta)\bar{R}_{\{k\}}$, where $b_{\{0\}} = 0$ and $\eta = 0.95$. Here, $\bar{R}_{\{k\}}$ is the sample average of the reward at the $k$-th iteration, i.e. $\bar{R}_{\{k\}} = \frac{1}{M}\sum_{i=1}^{M} R^i_{\{k\}}(\tau^i)$.

In terms of policy optimization, we perform multiple steps of ADAM on the objective [Eq. equation 4.3.6]. The gradient update steps are 4 per minibatch. The clipped parameter in the objective is set to $\epsilon = 0.1$.

The hyperparameters of the algorithm are listed in Table 4.2.

**Algorithm 2** CD-QAOA with autoregressive network based policy

**Input:** batch size $M$, learning rate $\eta_t$, total number of iterations $T_{\text{iter}}$, exponential moving average coefficient $m$, entropy coefficient $\beta_S^{-1}$, PPO gradient steps $K$.

Generate and select the gauge potential sets $\mathcal{A}$ using Algo. 3.

Initialize the autoregressive network and initialize the moving average $\hat{R}=0$.

**for** $t = 1, .., T_{\text{iter}}$ **do**

  Autoregrssively sample a batch of discrete actions of size $M$, denoted as $B$:

$$\tau^k=(a_1^k, a_2^k, \cdots, a_q^k) \sim \pi_{\boldsymbol{\theta}}\left(a_1, a_2, \cdots, a_q\right), \ k=1,2,\cdots,M.$$

  Apply the SLSQP solver to the lower-level continuous optimization (cf. Sec 3.9):

$$\min_{\{\alpha_j^k\}_{j=1}^q} \left\{ N^{-1}E(\{\alpha_j^k\}_{j=1}^q, \tau^k) \middle| \sum_{j=1}^q \alpha_j^k = T;\ 0 \le \alpha_j^k \le T \right\}.$$

  Use the negative energy density as the return and compute the moving average:

$$R_k = -N^{-1}E(\{\alpha_j^k\}_{j=1}^q, \tau^k), \quad \hat{R} = m\cdot\hat{R} + (1-m)\cdot\frac{1}{M}\sum_{k=1}^M R_k.$$

  Compute the advantage estimates $A_k = R_k - \hat{R}$.

  Initialize the parameter $\boldsymbol{\theta}_{t+1}^{[1]}=\boldsymbol{\theta}_t$.

  **for** $\kappa=1, .., K$ **do**

    Evaluate the likelihood of samples using the parameters from last iteration and current iteration, i.e. $\pi_{\boldsymbol{\theta}_t}(\tau^k)$, $\pi_{\boldsymbol{\theta}_{t+1}^{[\kappa]}}(\tau^k)$, and compute the importance weight $\rho_k^{[\kappa]}=\pi_{\boldsymbol{\theta}_{t+1}^{[\kappa]}}(\tau^k)/\pi_{\boldsymbol{\theta}_t}(\tau^k)$.

    Use the advantage estimate and importance weight to compute $\mathcal{G}_k, \mathcal{S}_k$, following Eq. equation 3.8.3 and Eq. equation 3.8.5.

    Compute the CD-QAOA objective Eq. equation 4.3.6 and backpropagate to get the gradients:

$$\nabla_{\boldsymbol{\theta}}\mathcal{J}(\boldsymbol{\theta}_{t+1}^{[\kappa]}) = \frac{1}{M}\sum_{\{a_j^{\{k\}}\}_{j=1}^q \in B} \nabla_{\boldsymbol{\theta}}\left[\mathcal{G}_k^{[\kappa]} + \beta_S^{-1}\mathcal{S}_k^{[\kappa]}\right].$$

    Update weights $\boldsymbol{\theta}_{t+1}^{[\kappa+1]} \leftarrow \boldsymbol{\theta}_{t+1}^{[\kappa]} + \eta_t \nabla_{\boldsymbol{\theta}}\mathcal{J}(\boldsymbol{\theta}_{t+1}^{[\kappa]})$.

  **end for**

  Update the parameter $\boldsymbol{\theta}_{t+1} \leftarrow \boldsymbol{\theta}_{t+1}^{[K+1]}$

**end for**

| **Parameter** | **Value** |
|---|---|
| optimizer | Adam [163] |
| learning rate ($\eta_{\{0\}}$) | $1 \cdot 10^{-2}$ |
| learning rate decay steps | 50 |
| learning rate decay factor | 0.96 |
| learning rate decay style | Staircase |
| RL temperature ($\beta_{S,\{0\}}^{-1}$) | $1 \cdot 10^{-1}$ |
| RL temperature decay steps | 10 |
| RL temperature decay factor | 0.9 |
| RL temperature decay style | Smooth |
| baseline exponential moving decay factor ($m$) | 0.95 |
| gradient steps (PPO) | 4 |
| clip parameter $\epsilon$ | 0.1 |
| number of hidden layers | 2 |
| number of hidden units per layer ($d_{\text{hidden}}$) | 112 |
| nonlinearity | ReLU |
| number of samples per minibatch ($M$) | 128 |

Table 3.3: Hyperparameter values for training the autoregressive deep learning model. In the case of $|\mathcal{A}_{\text{CD-QAOA}}|=9, q=18$ [cf. Eq. equation 3.3] the total number of parameters is 24431; for $|\mathcal{A}_{\text{CD-QAOA}}|=7, q=20$ [cf. Eq. equation 3.7] the total number of parameters is 21985.

## 3.9 Low-level optimization: finding optimal protocol time steps $\alpha_j$

In order to determine the values of the time steps $\alpha_j$, we proceed as follows. For any given sequence of actions (or protocol sequence) $\tau=(a_1,\cdots,a_q)$ of total duration $T$, we solve the following low-level optimization problem:

$$\min_{\{\alpha_j\}_{j=1}^q} \left\{ N^{-1}E(\{\alpha_j\}_{j=1}^q,\tau) \middle| \sum_{j=1}^{q} \alpha_j = T;\ 0 \leq \alpha_j \leq T \right\} \tag{3.9.1}$$

where $q$ is the sequence length (circuit depth), $N$ is the system size, and $E(\cdot)$ is the energy of the final quantum state [cf. Eq. equation 3.2.2] after evolving the initial quantum state $|\psi_i\rangle$ according to the fixed protocol $\tau$.

Note that the $\alpha_j$-optimization is both bounded and constrained. It fits naturally into

the framework of the Sequential Least Squares Programming (SLSQP). SLSQP solves the nonlinear problem in Eq. equation 3.9.1 iteratively, using the Han-Powell quasi-Newton method with a Broyden–Fletcher–Goldfarb–Shanno (BFGS)[238] update of the B-matrix (an approximation to the Hessian matrix), and an $L1$-test function within the step size.

During each iteration of policy update, a batch of trajectories $\{\tau^i\} = \{(a_1^i, \cdots, a_q^i)\}_{i=1}^M$ is sampled. Each trajectory sequence $\tau^i$ is assigned a reward, by solving the optimization problem in Eq. equation 3.9.1. Since performing the low-level optimization in Eq. equation 3.9.1 is independent of the high-level optimization discussed in Sec. 3.8, we run the former concurrently to boost the efficiency of the algorithm. We distribute every sequence $\tau^i = (a_1^i, \cdots, a_q^i)$ to a different worker process and aggregate the results back to the master process in the end. In practice, we use the batch size $M = 128$, and we distribute the simulation on 4 nodes with 32 cores each, so that each core solves only one optimization at a time.

Recently, it was demonstrated that it is possible to perform the continuous optimization on par with the discrete one, which eliminates the need to use a solver and results in a fully RL optimization approach [351].

## 3.10 Scaling with the number of particles $N$, the protocol duration $T$, and the circuit depth $q$

Next, we discuss the computational scaling of CD-QAOA. While there are a number of (hyper-)parameters in the algorithm, here we focus on the system size $N$, the protocol duration $T$, and the circuit depth $q$ – which are physically the most relevant ones. We also consider the continuous and discrete optimization steps separately (the continuous step being also an essential part of conventional QAOA).

When it comes to the continuous optimization performed by a solver [cf. Sec. 3.9], the main computational cost comes from the quantum evolution itself. The basic operation inside the solver is a multiplication of the matrix exponential $\exp(-i\alpha_j H_j)$ by the state $|\psi_i\rangle$. The Hamiltonian $H_j$ is stored as a sparse matrix, and the action of the matrix exponential onto the quantum state, $\exp(-i\alpha_j H_j)\,|\psi_i\rangle$, can be evaluated without computing the matrix exponential itself with the help of a sparse matrix-vector product; this operation scales exponentially with the system size $N$, i.e. $O(\exp(cN))$ for some constant $c$. If we denote the sequence length (a.k.a. circuit depth) by $q$, then the total cost for evaluating a single value of the continuous angle $\alpha$ scales as $O(q\exp(cN))$. We stress that this cost is also incurred by conventional QAOA.

| $N$ | $T$ | $q$ | $t_{\text{solver}}$ (sec/iter) | $t_{\text{RL}}$ (sec/iter) |
|---|---|---|---|---|
| 10 | 20 | 20 | $57.254 \pm 13.829$ | $0.042 \pm 0.005$ |
| 8 | 20 | 20 | $17.24 \pm 2.554$ | $0.055 \pm 0.024$ |
| 6 | 20 | 20 | $10.559 \pm 3.963$ | $0.028 \pm 0.004$ |
| 4 | 20 | 20 | $6.021 \pm 5.149$ | $0.027 \pm 0.002$ |
| 10 | 28 | 20 | $68.55 \pm 19.044$ | $0.055 \pm 0.019$ |
| 10 | 24 | 20 | $61.425 \pm 15.171$ | $0.038 \pm 0.009$ |
| 10 | 20 | 20 | $57.254 \pm 13.829$ | $0.042 \pm 0.005$ |
| 10 | 16 | 20 | $49.043 \pm 12.447$ | $0.041 \pm 0.007$ |
| 10 | 12 | 20 | $39.33 \pm 13.976$ | $0.038 \pm 0.006$ |
| 10 | 8 | 20 | $24.689 \pm 14.348$ | $0.033 \pm 0.008$ |
| 10 | 4 | 20 | $7.023 \pm 2.651$ | $0.025 \pm 0.001$ |
| 8 | 20 | 24 | $20.723 \pm 3.903$ | $0.065 \pm 0.024$ |
| 8 | 20 | 20 | $17.24 \pm 2.554$ | $0.055 \pm 0.024$ |
| 8 | 20 | 16 | $12.626 \pm 3.129$ | $0.024 \pm 0.004$ |
| 8 | 20 | 12 | $8.641 \pm 2.654$ | $0.02 \pm 0.003$ |
| 8 | 20 | 8 | $5.511 \pm 2.18$ | $0.016 \pm 0.002$ |
| 8 | 20 | 4 | $2.092 \pm 1.312$ | $0.011 \pm 0.002$ |

Table 3.4: Wall clock running time of the two-level CD-QAOA optimization steps for the with different system sizes $N$, protocol durations $T$, and circuit depths $q$. The right-hand side of the table shows the time used for the lower-level solver (column $t_{\text{solver}}$) and the time spent for the high-level RL algorithm (column $t_{\text{RL}}$) at every successful iteration. The total cost can then be obtained by multiplying the time for $t_{\text{solver}}$ by the appropriate number of repetitions (e.g., continuous solver realizations, policy sample batch size, PPO training episodes, etc.), taking into account any parallelization if present. Every number represents an average over 40 independent runs with the corresponding standard deviation shown; the significant deviation in $t_{\text{solver}}$ is caused by the random initial solver state used which causes the algorithm to take a different number of steps to converge within the given tolerance, cf. Sec. 3.11. This test is carried out on a single processor Intel Core i7-8700K CPU 6-core 3.70GHz.

For the discrete optimization performed using reinforcement learning [Sec. 3.8], notice first that the machine learning model is agnostic to the physical quantum model, because we do not use information about the quantum model to train the policy, cf. Sec. 3.8. Because the policy input is, by construction, independent of the quantum state, the input layer of the neural network architecture is shielded from the exponential growth of the physical Hilbert space with $N$. Hence, the deep neural network is *independent* of the Hilbert space dimension. Further, we use an autoregressive network model which scales linearly with the sequence length $q$, and also linearly with the size of the available action set $|\mathcal{A}|$. Thus, the total computational cost for the reinforcement learning optimization scales as $O(q|\mathcal{A}|)$. The scaling of the neural network with the variational networks parameters (weights and biases) is trivially given by the matrix-vector multiplication, as is the case for typical ML deep networks, and is also independent of the physics of the controlled system.

A comparison of the wall clock time for the discrete and continuous optimization steps is provided in Table 3.4. We distinguish between the continuous solver optimization and the discrete RL optimization, and show the average times for *one* successful step of each in the two columns on the right-hand-side. The total cost can then be obtained by multiplying the time for $t_{\text{solver}}$ by the appropriate number of repetitions (e.g., continuous solver initial conditions, policy sample batch size, PPO training episodes, etc.), and by multiplying the time for $t_{\text{RL}}$ by the number of PPO iterations, thereby taking into account any parallelization if used; for instance, the most expensive simulation we performed ran for about 109 hours on four nodes (Intel Xeon Skylake 6130 32-core 2.1 GHz) to produce the $N = 10$, $T = 12$, $q = 20$ data point shown in Table 3.2.

We emphasize that the time $t_{\text{solver}}$ required for the continuous optimization is an essential part of conventional QAOA, and is the current limiting factor for reaching large system sizes, as is the case in merely all simulations of quantum dynamics on classical computing devices. In sharp contrast, the cost for training the deep autoregressive network is $N$-independent, and $t_{\text{RL}}$ per iteration is negligible; however, the choice of RL algorithm can strongly impact the number of iterations. CD-QAOA is, thus, suitably designed for potential applications on quantum simulators and quantum computers which will enable accessing large system sizes bypassing the exponential bottleneck intrinsic to simulations of quantum dynamics on classical devices.

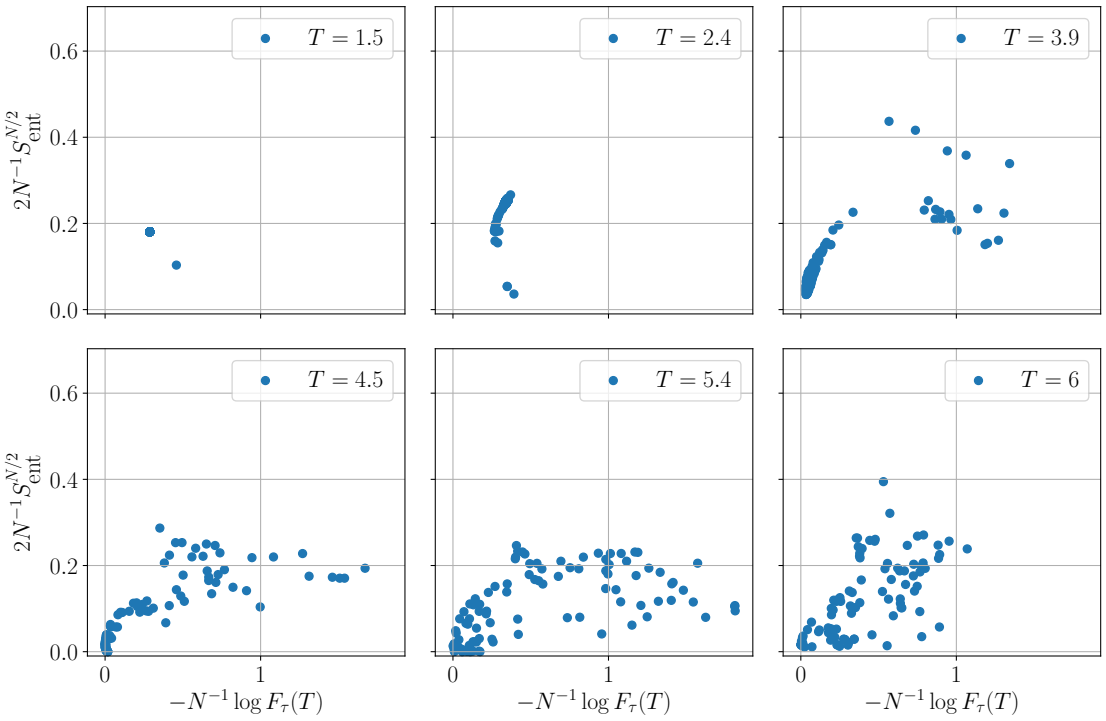



Figure 3.14: Spin-1/2 Ising model: Visualization of the continuous optimization landscape for the durations $\alpha_j$ in the fidelity-entanglement entropy plane, for the best sequence found by the RL agent [see Sec. 3.11]. Each point corresponds to a local minimum, obtained using the SLSQP optimizer, starting from a uniformly drawn random initial condition. The system size is $N = 16$, and the rest of the parameters are the same as in Fig. 3.1.

## 3.11 Many-Body Control Landscape

Let us briefly address the question about how hard the many-body ground state preparation problems are, that we introduced in the main text. To this end, recall that CD-QAOA has a two-level optimization structure: (i) discrete optimization to construct the optimal sequence of unitaries [Sec. 3.8], and (ii) continuous optimization to find the best angles, given the sequence, to minimize the cost function [Sec. 3.9]. Here, we focus exclusively on the continuous optimization landscape, and postpone the discrete landscape to a future study.

The RL agent learns in batches/samples of $M = 128$ sequences, which sample the current policy at each iteration step and provide the data set for the policy gradient

algorithm. To evaluate each sequence in the batch, we use SLSQP to optimize for the durations $\alpha_j$ in a constrained and bounded fashion: $\sum_j \alpha_j = T$ and $0 \leq \alpha_j \leq T$ [cf. Sec. 3.9]. This provides us with the full unitary $U(\{\alpha_j\}_{j=1}^q, \tau)$; applying it to the initial state we obtain the reward value for this sequence. This procedure repeats iteratively as the RL agent progressively discovers improved policies.

Once the RL agent has learned an optimal sequence, i.e. after the optimization procedure is complete, we focus on the best sequence from the sample, and examine how difficult it is to find the corresponding durations $\alpha_j$ using SLSQP. To this end, we draw $q$ values at random from a uniform distribution over the interval $[0, T/q]$, and use them as initial conditions for the $\alpha_j$, to initialize the SLSQP optimizer with. We use the same $q$ as the circuit depth so that the initial durations $\alpha_j^{(0)}$ are, on average, equal. We then repeat this procedure $P$ times, and generate a sample $\mathcal{M} = \left\{\{\alpha_j^n\}_{j=1}^q\right\}_{n=1}^P$ of the local minima in the optimization landscape for $\alpha_j$'s. The larger $P$, the better our result for the true reward assigned to $\tau$ is.

Notice that, in the beginning of the training, the RL agent is still in the exploration stage and the reward estimation does not need to be too accurate; this reward estimation needs to be more accurate as the agent switches over exploitation during the end of the training. In order to make the algorithm computationally more efficient, we introduce a linear schedule for the number of realizations of the $\alpha_j$-optimizer, starting from 3 with an increment of 1 every 30 iteration steps, i.e. $P_{\{k\}}^{\text{tot}} = 3 + \lfloor k/30 \rfloor$, where subscript $k$ indicates the iteration number for the RL policy optimization. In order to further save time in the reward estimation, we also introduce some randomness here by sampling $P_{\{k\}}$ from a uniform distribution over $1, 2, \cdots, P_{\{k\}}^{\text{tot}}$.

Even though they all correspond to the same sequence, every local minimum in $\mathcal{M}$ represents a potentially different protocol, since the durations $\alpha_j$ will cause the initial quantum state to evolve into a different final state. We can evaluate for every protocol in $\mathcal{M}$ the negative log-fidelity, $-\log F_\tau(T)$, and entanglement entropy of the half chain, $S_{\text{ent}}^{N/2}$. Since the target state for the Ising model is an ordered ground state, it has area-law entanglement. Figure 3.14 shows a cut through the landscape in the fidelity-entanglement entropy plane for a few different durations $T$ for the spin-1/2 Ising model. The better solutions are located in the lower left corner. The proliferation of local minima across the quantum speed limit has recently been studied in the context of RL [82] and QAOA [209]. This behavior indicates the importance of running many different SLSQP realizations, or else we may mis-evaluate the reward of a given sequence and the policy gradient will perform poorly.

Figure 3.14 also provides a plausible explanation for the destruction of the scaling collapse for $T \gtrsim T_{\text{QSL}}$ [Fig. 3.2]. Although the precision of the SLSQP optimizer

is set at $10^{-6}$, the energy curves for large durations no longer fall on top of each other with a larger relative error. Hence, the occurrence of many local minima of roughly the same reward, which correspond to different protocols, effectively removes any universal features from the obtained solution; therefore, different system size simulations end up in different local minima.

## 3.12 Variational Gauge Potentials

Consider the generic Hamiltonian

$$H(\lambda) = H_0 + \lambda H_1, \tag{3.12.1}$$

with a general smooth function $\lambda = \lambda(t)$. We define a state preparation problem where the system is prepared in the ground state of $H_0$ at time $t = 0$, and we want to transfer the state population in the ground state of $H$ by time $t = T$.

Unlike adiabatic protocols, counter-diabatic driving relaxes the condition of being in the instantaneous ground state of $H(\lambda)$ during the evolution. The idea is to reach the target state in a shorter duration $T$ (compared to the adiabatic time) at the expense of creating controlled excitations [w.r.t. the instantaneous $H(\lambda)$] during the evolution, which are removed before reaching the final time $T$. To achieve this, one can define a counter-diabatic Hamiltonian $H_{\rm CD}$. In general, the original $H(\lambda)$ differs from $H_{\rm CD}$, whose ground state the system follows adiabatically:

$$H_{\rm CD}(\lambda) = H(\lambda) + \dot{\lambda} A_\lambda, \tag{3.12.2}$$

where $A_\lambda$ is the gauge potential; $A_\lambda$ is defined implicitly as the solution to the equation [173]

$$[\partial_\lambda H + i[A_\lambda, H], H] = 0. \tag{3.12.3}$$

The boundary conditions $H_{\rm CD}(\lambda(0)) = H(\lambda(0))$ and $H_{\rm CD}(\lambda(T)) = H(\lambda(T))$ impose the additional constraint $\dot{\lambda}(0) = 0 = \dot{\lambda}(T)$ which suppresses excitations at the beginning and at the end of the protocol.

Using Eq. equation 3.12.3, one can convince oneself that the gauge potential $A_\lambda$ of a real-valued Hamiltonian $H$ is always imaginary-valued [173].

For generic many-body systems, it has recently been argued that the gauge potential $A_\lambda$ is a nonlocal operator [280]. Nevertheless, one can proceed by constructing a variational approximation $\mathcal{X} \approx A_\lambda$, which minimizes the action

$$\mathcal{S}(\mathcal{X}) = \langle G^2(\mathcal{X}) \rangle - \langle G(\mathcal{X}) \rangle^2, \quad G(\mathcal{X}) = \partial_\lambda H + i[\mathcal{X}, H]. \tag{3.12.4}$$

For ground state preparation, $\langle\cdot\rangle = \langle\psi_{\rm GS}(\lambda)|\cdot|\psi_{\rm GS}(\lambda)\rangle$ is the instantaneous ground state expectation value w.r.t. $H(\lambda)$. More generally, one can use $\langle\cdot\rangle = {\rm Tr}(\rho_{\rm th}\times(\cdot))$, where $\rho_{\rm th}\propto\exp(-\beta H)$ is a thermal density matrix at temperature $\beta^{-1}$: for $\beta\to\infty$, we recover the ground state expectation value; for $\beta\to 0$ all eigenstates are weighted equally.

We mention in passing that alternative schemes to approximation the adiabatic gauge potential have also been considered [140].

## Spin Hamiltonians

### Real-valued Spin-1/2 Hamiltonians

Let $H$ now be a real-valued spin-1/2 Hamiltonian with translation and reflection invariance. Such a system is given, e.g., by the mixed-field Ising model, discussed in the main text. We now construct an ansatz for the variational gauge potential $\mathcal{X}$ which obeys these symmetries, and is imaginary valued.

We can organize the terms contained in $\mathcal{X}$ according to their multi-body interaction type, as follows. The only single-body imaginary valued term we can write is $\sum_j \beta_j S^y_j$. Translation and reflection symmetries, whenever present in $H$, further impose that the coupling constant $\beta_j=\beta$ be site-independent, i.e. spatially uniform. Hence, this is the zeroth-order term in our variational gauge potential construction, cf. Eq. equation 3.12.5.

Next, we focus on the two-body terms. Because the exact $A_\lambda$ is imaginary valued for real-valued Hamiltonians, we may only consider interaction terms where $S^y$ appears precisely once: $S^xS^y$ and $S^yS^z$. For spin-1/2 systems, the two operators have to act on different sites, or else one can further simplify their product to single-body operators using the algebra for Pauli matrices. Once again, translation invariance dictates that the coupling constants are uniform in space, while reflection invariance requires us to take a symmetric combination. Imposing further that the interaction be short-range (we want to construct the most local variational ansatz), we arrive at

$$\begin{aligned}\mathcal{X}(\{\beta_l^{(k)}\}) = \sum_j \quad & \beta_0^{(0)}(\lambda) S^y_j + \beta_1^{(0)}(\lambda)\left(S^x_{j+1}S^y_j + S^y_{j+1}S^x_j\right) + \\ & + \beta_1^{(1)}(\lambda)\left(S^z_{j+1}S^y_j + S^y_{j+1}S^z_j\right). \end{aligned} \tag{3.12.5}$$

The coefficients $\beta_l^{(k)}$ are the variational parameters that we need to determine to find the approximate CD protocol. To find their optimal values, we minimize the action $\mathcal{S}(\mathcal{X})$ [173]. Note that, since we do not have a closed-form expression for the

instantaneous ground state of $H(\lambda)$, we do the minimization numerically at every fixed time $t$ along the protocol $\lambda(t)$ [cf. Sec. 3.12].

We can, in principle, add the next order terms to the series; however, they will either be less local, or consist of three- and higher-body interactions, which is hard to implement in experiments.

### Real-valued Spin-1 Hamiltonians

The situation is more interesting for spin-1 systems: the eight-dimensional Lie algebra $\mathfrak{su}(3)$, which generates SU(3), contains three distinct imaginary-valued directions, which form a closed subalgebra $\mathfrak{su}(2) \subsetneq \mathfrak{su}(3)$, and hence there is more room to generate imaginary-valued combinations. To find all imaginary-valued terms consistent with a set of symmetries, we use QuSpin's functionality to implement an algorithm [Sec. 3.12] that lists them for generic bases [328, 329].

***Translation and Reflection Symmetric spin-1 Hamiltonians***, such as the spin-1 Ising and Heisenberg models, have a similar expansion to their spin-1/2 counterparts, but allow for more terms. Restricting the expansion to two-body terms, we have

$$\begin{aligned}\mathcal{X}(\{\beta_l^{(k)}\}) = \sum_j \Big[ & \beta_0^{(0)}(\lambda) S_j^y + \beta_1^{(0)}(\lambda) \left(S_j^x S_j^y + S_j^y S_j^x\right) + \beta_2^{(0)}(\lambda) \left(S_j^z S_j^y + S_j^y S_j^z\right) \\ & + \beta_0^{(1)}(\lambda) \left([S_{j+1}^x - a S_j^x] S_j^y + [S_{j+1}^y - a S_j^y] S_j^x\right) \\ & + \beta_1^{(1)}(\lambda) \left([S_{j+1}^z - b S_j^z] S_j^y + [S_{j+1}^y - b S_j^y] S_j^z\right) \Big].\end{aligned} \tag{3.12.6}$$

where the constants $a$ and $b$ are chosen so that all five terms are mutually orthogonal w.r.t. the scalar product induced by the trace (i.e. Hilbert-Schmidt) norm; this ensures the linear independence of the constituent terms. Note that the three imaginary-valued on-site terms correspond precisely to the imaginary-valued $\mathfrak{su}(2) \subsetneq \mathfrak{su}(3)$.

Adding ***Magnetization Conservation and Spin Inversion Symmetry*** further reduces the allowed terms in the series. Therefore, one has to restrict to three- and four-body terms:

$$\begin{aligned}\mathcal{X}(\{\zeta_l^{(k)}\}) \; = \; & \sum_j \zeta_0^{(2)}(\lambda) \left(i S_j^+ S_{j+1}^- S_{j+2}^z + i S_j^z S_{j+1}^- S_{j+2}^+ + \text{h.c.}\right) + \\ & \zeta_0^{(3)}(\lambda) \left(i S_j^- S_j^z S_{j+1}^+ S_{j+1}^z + i S_j^+ S_j^z S_{j+1}^- S_{j+1}^z + \text{h.c.}\right),\end{aligned} \tag{3.12.7}$$

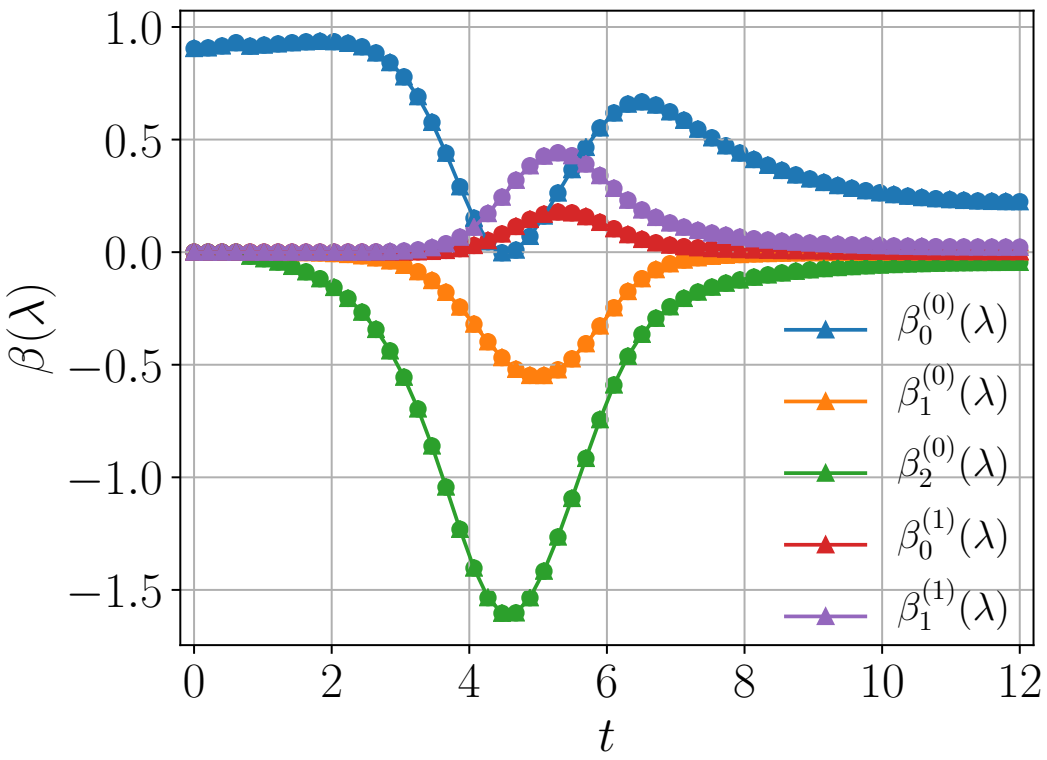



Figure 3.15: Spin-1 Ising chain. Time dependence of the optimal coefficients $\beta_l^{(k)}(\lambda(t))$ in the variational gauge potential (Eq. 3.12.6) with translation and reflection symmetry, determined from the procedure in Sec. 3.12. The total duration $T = 12$ with the time discretization step $\Delta t = 0.2$, and the system size $N = 8$. The protocol we used is $\lambda(t) = \sin^2\left(\frac{\pi t}{2T}\right)$. The other model parameters are the same as in Fig. 3.7.

Because these terms are multi-body and less local, we refrain from using them in CD-QAOA in the present study. We merely list them here for completeness.

As explained in the main text, to apply CD-QAOA for many-body ground state preparation, we consider the constituent terms in $\mathcal{X}$ as independent generators $\{H_j\}_{j=1}^{|\mathcal{A}|}$. This comes in contrast to the variational gauge potential method where the ratios between the coefficients $\beta_l^{(k)}$ play an important role.

**Variational Gauge Potential Ansatz for the Lipkin-Meshkov-Glick Model**

As explained in the main text, the Lipkin-Meshkov-Glick (LMG) Hamiltonian, cf. Eq. equation 3.4.3, models homogeneously interacting spin-1/2 particles on an all-to-all connected graph in the presence of an external field. Here, we compute the lowest-order terms appearing in the series for the variational gauge potential $\mathcal{X}$, going beyond Ref. [136].

The starting point is the LMG Hamiltonian

$$H = -\frac{J}{N}\left(S^x\right)^2 + h\left(S^z + N/2\right). \tag{3.12.8}$$

We introduce two bosonic modes, $s$ and $t$, where $S^z = t^\dagger t - N/2 = n_t - N/2$ and $S^+ = t^\dagger s$, and cast the LMG Hamiltonian in the form

$$H = ht^\dagger t - \frac{J}{4N}\left(t^\dagger s + s^\dagger t\right)^2. \tag{3.12.9}$$

Recalling once again that real-valued Hamiltonians have imaginary-valued gauge potentials, and that gauge potentials do not have diagonal matrix elements, we make the following ansatz:

$$\mathcal{X}(\{\beta_l^{(k)}\}) = \beta_0^{(0)}(\lambda)Y + \beta_1^{(1)}(\lambda)\hat{XY} + \beta_1^{(0)}(\lambda)\hat{ZY}, \tag{3.12.10}$$

where

$$\begin{aligned}
Y &= S^y = \frac{i}{2}\left(s^\dagger t - t^\dagger s\right), \\
\hat{XY} &= \frac{1}{N}(S^x S^y + S^y S^x) = -\frac{i}{2N}\left[(t^\dagger s)^2 - (s^\dagger t)^2\right], \\
\hat{ZY} &= \frac{1}{N}\left(\left(S^z + \frac{N}{2}\right)S^y + S^y\left(S^z + \frac{N}{2}\right)\right) \\
&= \frac{i}{2N}\left(s^\dagger t^\dagger t t - s t^\dagger t t^\dagger + s^\dagger t t^\dagger t - s t^\dagger t^\dagger t\right).
\end{aligned} \tag{3.12.11}$$

To compute the matrix elements of the gauge potentials, we define the basis

$$|N, n_t\rangle = \frac{\left(t^\dagger\right)^{n_t}\left(s^\dagger\right)^{N-n_t}}{\sqrt{n_t!(N-n_t)!}}|0\rangle, \quad \text{with} \quad n_t = 0, \ldots, N.$$

The gauge potentials have the following non-zero matrix elements (plus their conjugates to make the operators hermitian):

$$\begin{aligned}
\langle N, n_t|Y|N, n_t+1\rangle &= -\frac{i}{2}\sqrt{(n_t+1)(N-n_t)}, \\
\langle N, n_t|\hat{XY}|N, n_t+2\rangle &= \frac{i}{2N}\sqrt{(n_t+2)(n_t+1)(N-n_t-1)(N-n_t)}, \\
\langle N, n_t|\hat{ZY}|N, n_t+1\rangle &= \frac{i}{2N}(2n_t+1)\sqrt{(n_t+1)(N-n_t)}.
\end{aligned} \tag{3.12.12}$$

## Numerical Minimization to obtain the Variational CD Protocol

Since the action $\mathcal{S}$ in Eq. equation 3.12.4 is quadratic in the variational parameters $\beta_j$, it is possible to derive a generic linear system, whose solutions are the optimal parameters of the variational gauge potential within CD driving [226].

Suppose that $\mathcal{X} = \sum_{j=1}^{r} \beta_j H_j$ is given by a linear combination of $r$ gauge potential terms. Then, it is straightforward to see that

$$G(\mathcal{X}) = \partial_\lambda H + \sum_{j=1}^{r} i[H_j, H]\beta_j. \tag{3.12.13}$$

Defining the operator-valued quantities $B_0 = \partial_\lambda H$ and $B_j = i[H_j, H]$ and setting $\beta_0 = 1$, we arrive at the following expression for the variational action

$$\begin{aligned}
\mathcal{S}(\mathcal{X}) &= \left\langle \left( B_0 + \sum_j B_j\beta_j \right)^2 \right\rangle - \left( \langle B_0 + \sum_j B_j\beta_j \rangle \right)^2 \\
&= \sum_{i,j=0}^{r} \Big( \langle B_i B_j\rangle - \langle B_i\rangle\langle B_j\rangle \Big)\beta_i\beta_j,
\end{aligned} \tag{3.12.14}$$

which is a quadratic form in the unknown coefficients $\beta_j$. To find the minimum of $\mathcal{S}(\mathcal{X})$ w.r.t. $\beta_j$, we can take the derivative and set it to zero, to obtain the linear system of equations for the optimal $\beta_j$:

$$\sum_k \mathcal{M}_{jk}\beta_k = -\mathcal{M}_{0j} \tag{3.12.15}$$

where $\mathcal{M}_{jk} = \langle B_j B_k\rangle + \langle B_k B_j\rangle - 2\langle B_j\rangle\langle B_k\rangle$. Solving the system we obtain the minimum $\{\beta_j\}_{j=1}^{r}$ of the variational action $\mathcal{S}$.

The ground state expectation values in the above procedure, as well as the Hamiltonian $H(\lambda(t))$ depend implicitly on time $t \in [0, T]$ via the protocol $\lambda(t)$. Therefore, to find the time dependence of $\beta_j(t)$, we discretize the time interval $[0, T]$ into $N_T$ time steps, and repeat the procedure at every time step. This yields $\beta_j(t_i)$ at the time steps $t_i$. To recover the full functional dependence, we use a fine discretization mesh, and apply a linear interpolation to $\beta_j(t_i)$. Alternatively, notice that the coefficients $\beta_j = \beta_j(\lambda(t))$ depend on time $t$ only implicitly via the protocol $\lambda$. Therefore, it is also possible to discretize the range of $\lambda(t)$ instead.

For the spin-1 Ising model, the time-dependence of $\beta_j$ is shown in Fig. 3.15. This defines $H_{\text{CD}}$ which generates the CD evolution. In Sec. 3.5 and Sec. 3.13, we compare variational CD driving to CD-QAOA and conventional QAOA.

## Algorithm for Generating Gauge Potential Terms in the Presence of Lattice Symmetries

Finally, we also show the algorithm we used to determine the terms appearing in the gauge potential expansions in Eqn. equation 3.12.5, equation 3.12.6, and equation 3.12.7, which obey a fixed set of symmetries.

In general, one can represent any local operator of the kind $J_{i_1,\cdots,i_l} O_{i_1}^{\gamma_1} \cdots O_{i_l}^{\gamma_l}$ as a triple $(\mathcal{Y}, \mathcal{I}, J)$, where $J = J_{i_1,\cdots,i_l}$ is the coupling coefficient constant, $\mathcal{I} = (i_1, \cdots, i_l)$ is the set of sites the operators act on, and $\mathcal{Y} = (\gamma_1, \cdots, \gamma_l)$ defines the types of operators that act on the corresponding sites; the triple $(\mathcal{Y}, \mathcal{I}, J)$ can then be used to construct the operator.

In the following, we refer to the separate terms appearing in the gauge potential series as 'Hamiltonians' $H_j$, i.e. $\mathcal{X} = \sum_j \beta_j H_j$; a Hamiltonian is defined as $H = \sum_{(i_1,\cdots,i_l)\in\Lambda} J_{i_1,\cdots,i_l} O_{i_1}^{\gamma_1} \cdots O_{i_l}^{\gamma_l}$, where $\Lambda$ is the lattice graph. As we argued above, real-valued Hamiltonians have purely imaginary-valued gauge potentials; thus, the coefficient $J$ is chosen to be purely imaginary.

We build the series for the variational gauge potential $\mathcal{X}$ recursively: we first consider a set $\mathcal{L}_{\text{elem}}$ of elementary operators $O$ — the building blocks for the expansion: e.g., for the spin-1 chains, these can be the spin-1 operators $\mathcal{L}_{\text{elem}} = \{S^+, S^-, S^z\}$. We want to construct the terms in the expansion for $\mathcal{X}$ iteratively at a fixed order $l$, e.g. $l = 1$ comprises single-body terms, $l = 2$ – two-body terms, etc. We also assume that we have access to a routine which checks if a trial list of operators obeys a given lattice symmetry; if not, the same routine returns the missing operators to be added to the original list, so that the symmetry is now satisfied [e.g., such a routine is used in QuSpin [328, 329]].

The pseudocode we developed is shown in Algorithm 3. To construct multi-body terms at a fixed order $l$, we define combinations of the elementary operators, and store them in the list $\mathcal{L}_{\text{op}}$; the way these combinations are built can be used to implement constraints, such as particle/magnetization conservation, etc. This is implemented via the product operator (Line 2 of Algorithm 3). It generates all possible combinations of selecting $l$ elementary operators with replacement. The sets of lattice sites that the operators from $\mathcal{L}_{\text{op}}$ act on, are stored in the list $\mathcal{L}_{\text{sites}}$ (Line 3 of Algorithm 3). Then, for each trial triple ($\mathcal{Y}$, $\mathcal{I}$, $J$), we make use of the routine to check the symmetry and record any operators which do not respect it. We append these, so-called *missing operators*, to the original list, and we keep checking the symmetry condition until we obtain all operators that satisfy the symmetry (Line 10 -15 of Algorithm 3). The finite number of combinations guarantees a termination in a finite number of steps.

**Algorithm 3** Generation of variational gauge potential

**Input:** a list of required symmetries $\mathcal{L}_{\text{sym}}$, order $l$, a list of elementary operator types $\mathcal{L}_{\text{elem}}$.

1: Initialize empty list for gauge potential terms $\mathcal{L}_{\text{gauge}}$.
2: Generate all possible combinations of local operators at order $l$

$$\mathcal{L}_{\text{op}} = \text{product}(\mathcal{L}_{\text{elem}}, \text{repeat} = l).$$

3: Enumerate all possible combinations of lattice sites $\mathcal{L}_{\text{sites}}$ the $l$-th order operators act on.
4: **for** $\mathcal{Y}$ in $\mathcal{L}_{\text{op}}$ **do**
5:  **for** $\mathcal{I}$ in $\mathcal{L}_{\text{sites}}$ **do**
6:  Initialize an empty list $\mathcal{L}_H$
7:  Set $J = i$ ($i = \sqrt{-1}$).
8:  Append ($\mathcal{Y}$, $\mathcal{I}$, $J$) to $\mathcal{L}_H$.
9:  Set the flag $IsSym = False$.
10:  **while** $IsSym$ is $False$ **do**
11:  Set $IsSym = True$.
12:  **for** sym in $\mathcal{L}_{\text{sym}}$ **do**
13:  **if** exists missing operator ($\mathcal{Y}'$, $\mathcal{I}'$, $J'$) **then**
14:  Set $IsSym = False$.
15:  Append ($\mathcal{Y}'$, $\mathcal{I}'$, $J'$) to $\mathcal{L}_H$.
16:  **end if**
17:  **end for**
18:  **end while**
19:  Build Hamiltonian $H$ using the triplets in $\mathcal{L}_H$.
20:  **if** $H$ or equivalents not included in $\mathcal{L}_{\text{gauge}}$ **then**
21:  Append $H$ to $\mathcal{L}_{\text{gauge}}$ .
22:  **end if**
23:  **end for**
24: **end for**
25: **Return** the list of gauge potential basis $\mathcal{L}_{\text{gauge}}$.

`product`: Cartesian product, `equivalents`: equivalent mod scalar,
`missing operator`: the operator missed for the symmetry requirement

In order to avoid repeating previously identified Hamiltonians, we discard *equivalent* Hamiltonians (Line 20 of Algorithm 3): two Hamiltonians are called equivalent when one is a scalar times the other. Since here we consider imaginary-valued gauge potentials, the multiple constant should be real. To test whether the Hamiltonians $H_1$

and $H_2$ are equivalent in practice, it suffices to test whether $H_1$ is equal to $\pm\frac{\|H_1\|}{\|H_2\|}H_2$, where we use the Hilbert-Schmidt norm.

## 3.13 CD-QAOA for Many-Body State Preparation

Here, we provide a supplementary discussion on the performance of CD-QAOA for many-body pure state preparation using the quantum spin chains introduced in the main text. We refer the reader to the main text for the definition of various model parameters; the short-hand spin operator notation used is defined in Table 3.1.

### Spin-1/2 Ising Chain

First, we show results for the single-spin problem ($J = 0$):

$$H = H_1 + H_2, \qquad H_1 = h_z S^z, \quad H_2 = h_x S^x. \tag{3.13.1}$$

In Fig. 3.16, we clearly see that CD-QAOA [red curve] has a smaller quantum speed limit $T_{\rm QSL} \approx 4.0$ than conventional QAOA [blue]; this is anticipated, since CD-QAOA has a larger control space at its disposal. Moreover, we find that, for $T < T_{\rm QSL}$, CD-QAOA only makes use of a single $Y$ rotation by setting the durations $\alpha_j$ associated with any other unitaries from the set $\mathcal{A}$, to zero. As mentioned in the main text, conventional QAOA tries to represent this $Y$-rotation by means of Euler angles, i.e. composed of $X$ and $Z$ rotations; in general, this results in a higher duration cost to complete the population transfer (leading to a larger $T_{\rm QSL}$). However, for short durations $T$, a $Y$-rotation can be exactly obtained using a proper sequence of the $X$ and $Z$ terms. For these reasons, we find an exact agreement between the two curves for small values of $T \lesssim 3$.

Let us now switch on the spin-spin interaction strength $J > 0$; consider the spin-1/2 Ising chain

$$\begin{aligned} H &= H_1 + H_2, \\ H_1 &= \sum_{j=1}^{N} J S^z_{j+1} S^z_j + h_z S^z_j, \quad H_2 = \sum_{j=1}^{N} h_x S^x_j. \end{aligned} \tag{3.13.2}$$

Figure 3.17 [top] shows a comparison of the best learned energies, between conventional QAOA, and CD-QAOA for two sets $(\mathcal{A}, \mathcal{A}')$ with different number of unitaries: $|\mathcal{A}| = 5, |\mathcal{A}'| = 3$ [see caption]. We find that additionally using only the single-particle

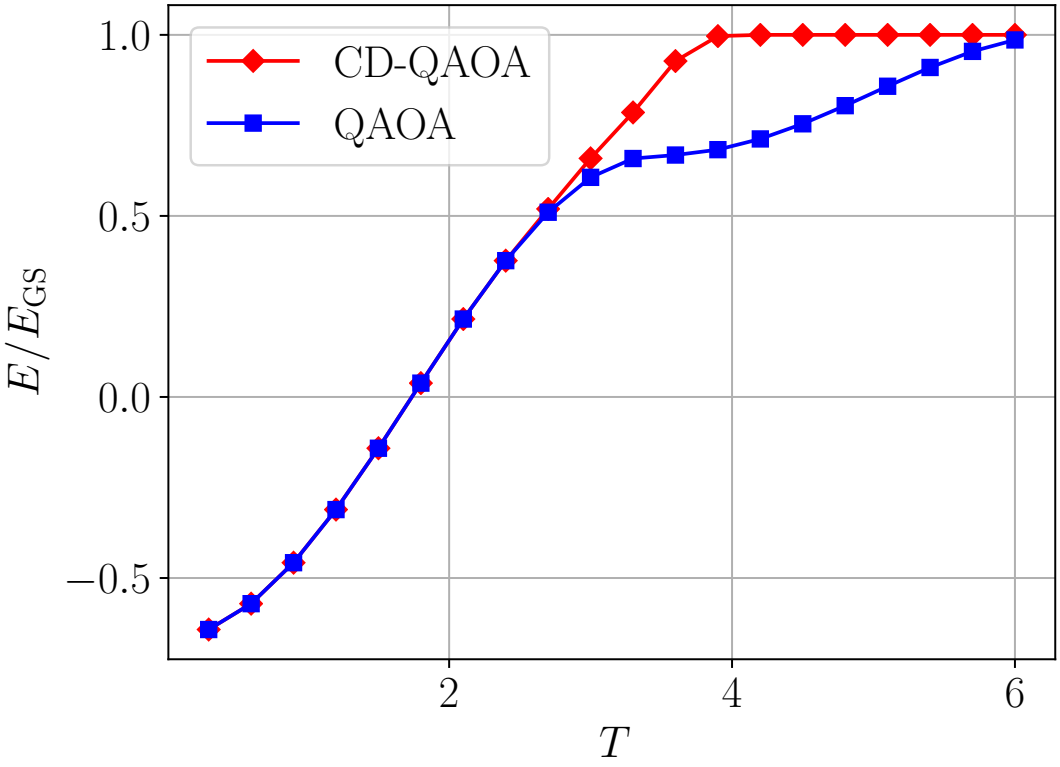



Figure 3.16: Single spin-1/2 state preparation: energy density against protocol duration for CD-QAOA with $\mathcal{A}_{\mathrm{CD-QAOA}} = \{Z, X, Y\}$ (red) and conventional QAOA with $\mathcal{A}_{\mathrm{QAOA}} = \{Z, X\}$ (blue). The values of $q$ is 3 for both methods. For conventional QAOA, we trained two possible alternating patterns $\big($i.e. $(Z \to X \to Z)$ and $(X \to Z \to X)\big)$ and pick the best one for the comparison. The model parameters are the same as in Fig. 3.1 with $J = 0$.

gauge potential term $Y$ [green line], typically accessible in experiments, one can already obtain a higher-fidelity protocol than QAOA to prepare the ground state. Interestingly, for short protocol durations $T$, the two-body gauge potential terms, present in $\mathcal{A}$ but not in $\mathcal{A}'$, do not contribute to improving the energy of the final state, as can be seen from the agreement of the red and green lines for $T \lesssim 1.5$. This suggests that single-particle processes dominate over many-body processes when it comes to lowering the energy of the $z$-polarized initial state, and implies that the target ground state is single-particle-like (i.e. close to a product state). The non-smooth behavior of the green curve at larger durations, is attributed to the

ruggedness of the control landscape, as different runs of the SLSQP optimizer may get stuck in one of the many suboptimal local minima [Sec. 3.11].

One may wonder if it is possible to prepare the ground state by straightforward fidelity maximization. We define the many-body fidelity to transfer the population to the target state using the unitary process $U(\{\alpha_j\}_{j=1}^q, \tau)$, with $\sum_{j=1}^q \alpha_j = T$, as

$$F_\tau(T) = F(\{\alpha_j\}_{j=1}^q, \tau) = |\langle\psi_*|U(\{\alpha_j\}_{j=1}^q, \tau)|\psi_i\rangle|^2. \tag{3.13.3}$$

The fidelity can be less relevant from the perspective of many-body physics because (i) the many-body fidelity is typically exponentially suppressed, and (ii) it requires a reference to the ground state itself (which we seek) in order to be computed. However, the fidelity of a quantum process is a widely used benchmark in quantum computing; it also provides a better measure (than energy density) for the distance between two states in the Hilbert space $\mathcal{H}$.

Figure 3.17 [bottom] shows the many-body fidelity for $N = 14$ spins. Unlike the inset of Fig. 3.1 from the main text (where we show the fidelity associated with the protocol obtained using energy density minimization), here we use the fidelity as a reward function for QAOA. We observe that optimizing the fidelity behaves quantitatively similar to optimizing the energy density. We would like to emphasize here once again the advantage of the gauge potential ansatz: the conventional QAOA simulation is done using $q = 80$ variational parameters $\alpha_j$ [yet no significant improvement is observed for $q \geq 4$, cf. Fig. 3.1], while CD-QAOA requires only $q = 3$ variational parameters.

Although the fidelity $F_\tau(T)$ is anticipated to vanish for $T < T_{\mathrm{QSL}}$ in the thermodynamic limit, the negative log-fidelity density, $-N^{-1} \log F_\tau(T)$, is more likely to. Figure 3.18 [inset] shows the finite size scaling of the fidelity curves. Similar to the energy density [Fig. 3.2], we obtain an almost perfect scale collapse. We verified that maximizing the fidelity produces similar results as minimizing the negative log-fidelity density for the spin-1/2 chain: at first sight, this is nontrivial because $F_\tau(T)$ is exponentially suppressed with the system size $N$ for $T < T_{\mathrm{QSL}}$; however, this behavior is likely explained by the generalization capabilities of the RL agent from small to large system sizes [cf. Sec. 3.6].

## Anisotropic Spin-1 Heisenberg Chain

Next, we discuss in detail the ground state preparation process in the anisotropic Heisenberg spin-1 chain:

$$\begin{aligned} H &= H_1 + H_2, \\ H_1 &= J\sum_{j=1}^{N}(S^x_{j+1}S^x_j + S^y_{j+1}S^y_j), \quad H_2 = \Delta\sum_{j=1}^{N} S^z_{j+1}S^z_j, \end{aligned} \tag{3.13.4}$$

where the model parameters are defined in the main text.

An important detail worth mentioning is that the ferromagnetic ground state at $\Delta/J = -2.0$ is two-fold degenerated (one state, corresponding to one of the two $z$-polarizations). While being a trivial observation, this requires certain care when analyzing the physics of the protocols the agent found. In particular, notice that energy minimization is insensitive to this degeneracy, and hence the final state can appear as an arbitrary superposition of the two ferromagnetic states, and still have the correct ground-state energy. This leads to ambiguity when computing the fidelity of being in the target state: related to this, the cost function landscape likely develops a continuous one-dimensional structure for the (degenerate) global minima. Because we are interested in energy minimization, here we define the fidelity using the projector to the ground state manifold $P = |\psi_*^{(1)}\rangle\langle\psi_*^{(1)}| + |\psi_*^{(2)}\rangle\langle\psi_*^{(2)}|$:

$$\begin{aligned} F_\tau(T) = F(\{\alpha_j\}_{j=1}^q, \tau) &= |\langle\psi_*^{(1)}|U(\{\alpha_j\}_{j=1}^q, \tau)|\psi_i\rangle|^2 \\ &\quad + |\langle\psi_*^{(2)}|U(\{\alpha_j\}_{j=1}^q, \tau)|\psi_i\rangle|^2 \end{aligned}$$

where $|\psi_*^{(1)}\rangle, |\psi_*^{(2)}\rangle$ are any two orthonormal states which span the doubly degenerate ground state manifold (e.g., the two FM ground states).

Figure 3.19 shows a comparison between CD-QAOA and conventional QAOA for FM, XY, and Haldane target states: the top row shows the result of energy density minimization [cf. Fig. 3.3]. The bottom row, on the other hand, displays the many-body fidelity associated with the same protocols. For $\Delta/J = 0.5$, CD-QAOA allows reaching the target topological Haldane state already faster, as compared to conventional QAOA. Notice also that the gauge potential ansatz appears essential for reaching the target for both the XY ($\Delta/J = -0.5$) and FM states ($\Delta/J = -2.0$); this becomes particularly obvious from the many-body fidelity curves. The latter also reveals an interesting detail: at $\Delta/J = 0.5$, a regime emerges around $T \approx 5$, where the QAOA fidelity is better than the CD-QAOA fidelity. However, this peculiarity below the quantum speed limit can be explained, recalling that the RL agent is given

the (negative) energy density as the reward signal, and not the fidelity (note that CD-QAOA does outperform QAOA in energy).

In order to investigate in detail in the protocols found by CD-QAOA, we fix a duration $T$, and consider the time evolution of the state, $|\psi(t)\rangle = U(\{\alpha_j\}_{j=1}^q, \tau)|\psi_i\rangle$, for three physical quantities:

(i) the energy

$$E(t) = \langle\psi(t)|H_*|\psi(t)\rangle$$

provides a measure of how far away in the cost function landscape the state is, at any given time $t \in [0, T]$.

(ii) the instantaneous fidelity

$$F_\tau(t) = |\langle\psi_*|\psi(t)\rangle|^2$$

(and its generalization to the doubly-degenerate ground state manifold), measures how far the current state is, from the target state $|\psi_*\rangle$ in the Hilbert space; typically, we choose the ground state as the target state $|\psi_*\rangle = |\psi_{\rm GS}(H)\rangle$.

(iii) the entanglement entropy of the half chain

$$S_{\rm ent}^{N/2}(t) = -{\rm tr}_A\left[\rho_A(t)\log\rho_A(t)\right],\ \rho_A(t) = {\rm tr}_{\bar{A}}|\psi(t)\rangle\langle\psi(t)|,$$

where $A$ denotes a contiguous spacial region with a complement $\bar{A}$ comprising half the periodic chain, and $\rho_A(t)$ is the reduced density matrix on $A$ at time $t$. For many-body systems, it is common to look at the entanglement entropy per site, which for spin-1 systems lies within the interval $2N^{-1}S_{\rm ent}^{N/2} \in [0, \log 3]$.

Figure 3.20 shows the time evolution of the energy, fidelity and entropy density, for all three target states of interest. For $\Delta/J = 0.5$, transferring the population from the AFM initial state to the Haldane state can be obtained equally well using either QAOA or CD-QAOA. Table 3.6(b) shows the optimal protocol found by the RL agent: notice the three vanishing durations $\alpha_2 = \alpha_{17} = \alpha_{18} = 0$; factoring them out, we recover precisely the conventional QAOA sequence (albeit with $q$ odd). Thus, we see that the CD-QAOA may converge to conventional QAOA whenever the latter provides a high-reward sequence. This result exemplifies our claim that CD-QAOA generalizes QAOA successfully. Of course, it is not clear whether this is the true global minimum of the cost function landscape (the RL agent does make use of the additional gauge potential terms for $T < 7$). Nevertheless, all physical quantities

are expected to be prepared with similar accuracy under both protocols: to see this, notice that the entanglement entropy density depends only on the quantum state (unlike expectation values of observables), and that its value at $t = T$ is close to the value for the target state (dashed horizontal line). Importantly, the entanglement remains area-law (as seen by the values being much smaller than the maximum entropy per site, log(3), suggesting the existence of a local effective Hamiltonian which generates the population transfer process dynamically.

The best sequence for targeting the XY state at $\Delta/J = -0.5$ is shown in Table 3.6(a). Although its structure is more complicated, factoring out the vanishing $\alpha_j$, we can discern two clear patterns: (i) the sequence starts and ends with two different single-particle basis rotations, and (ii) there is an alternating subsequence based on the subset $\{X|X+Y|Y,Y\} \subsetneq \mathcal{A}_{\mathrm{CD-QAOA}}$. Interestingly, the only gauge potential term used by the RL agent is the experimentally accessible single-particle $Y$ rotation, and it is sufficient to reach the target with a very high many-body fidelity. For comparison, conventional QAOA appears insufficient to prepare the target state for the circuit depth of $q = 18$ ($p = 9$). The advantage of CD-QAOA is also visible in the entanglement entropy density curve: QAOA can easily lead to volume-law entanglement, while CD-QAOA manages to generate as little entanglement as needed for the target state.

The discrepancy between conventional QAOA and CD-QAOA is best visible in the FM state preparation at $\Delta/J = -2.0$. In this case, a naïve application of QAOA with the set $\mathcal{A}_{\mathrm{QAOA}} = \{X|X+Y|Y,Z|Z\}$ is a priori doomed to fail: starting from the initial AFM state, which is orthogonal to the target FM manifold, the resulting QAOA unitaries leave the target AFM manifold invariant; in other words, transitions between the initial and the target states are forbidden by selection rules within the QAOA dynamics. Therefore, the many-body fidelity remains zero at all times during the QAOA evolution. The energy and entanglement entropy curves certify that the state does undergo nontrivial dynamics: similar to the XY state, QAOA creates volume-law entanglement and cannot reach the FM ground state manifold in energy, while CD-QAOA is well-behaved and sufficient to prepare the target. The CD-QAOA protocol sequence is shown in Table 3.5(b): while we do not discern an obvious pattern, we emphasize that this time the RL agent makes use of both single-particle and two-body gauge potential terms.

Last, we show the system-size scaling of the energy curves for the three target states in Fig. 3.21(b-d). Similar to the spin$-1/2$ Ising chain, we find very little system-size dependence for the Haldane (b) and XY states (c). However, we cannot extrapolate the results to the thermodynamic limit due to the relatively small system sizes we

were able to investigate. system-size effects are more pronounced for the ferromagnetic state (d), which is the one furthest away in Hilbert space from the initial perfect antiferromagnet.

Last, we mention in passing that we do not show results on preparing the AFM ground state at $\Delta/J=2.0$ since this problem is somewhat trivial: indeed, starting from a perfect AFM in the $z$-direction, the AFM ground state of the spin-1 Heisenberg model can be easily reached even using adiabatic evolution because it lies within the AFM phase.

## Lipkin-Meshkov-Glick model

In the main text, we also introduced the ferromagnetic Lipkin-Meshkov-Glick (LMG) model, described by the total spin Hamiltonian

$$H = -\frac{J}{N}(S^x)^2 + h\left(S^z + \frac{N}{2}\right).$$

Figure 3.22 shows the comparison between CD-QAOA and QAOA for two more values of $h/J = 0.1$ (deep in the ferromagnetic regime), and $h/J = 0.9$ (close to the critical point at $h/J = 1.0$). While the behavior for $h/J = 0.1$ is qualitatively similar to $h/J = 0.5$ (discussed in the main text), we do see that close to the critical point the two-body gauge potential terms $\hat{XY}$ and $\hat{ZY}$ may offer some degree of improvement below the quantum speed limit, as compared to using only using the single-body $\hat{Y}$ term. We mention in passing that we observed a stronger system-size dependence in the optimal protocol found by the RL agent in the immediate vicinity of the critical point $h_c/J = 1$.

## Spin-1 Ising Chain

Finally, let us turn to the spin-1 Ising chain:

$$\begin{aligned} H(\lambda) &= \lambda(t)H_1 + H_2, \\ H_1 &= \sum_{j=1}^{N} JS^z_{j+1}S^z_j + h_xS^x_j, \qquad H_2 = \sum_{j=1}^{N} h_zS^z_j, \end{aligned} \tag{3.13.5}$$

see main text for discussion of the model parameters. Using this model, we compare four state preparation techniques: CD-QAOA, conventional QAOA, CD-driving using a variational gauge potential, and adiabatic evolution.

In order to compare these four methods, we first investigate their energy budget, i.e. the amount of energy required by the corresponding protocols. This is necessary, since variational CD-driving does not put any constraints on the magnitude of the expansion parameters $\beta_j(\lambda)$ [cf. Sec. 3.12], and we know that larger energies (i.e. generators of unitaries $H_j$ with large norms) in general allow for a faster population transfer. To measure quantitatively the energy budget of a protocol, we use the average energy density along the protocol trajectory

$$\mathcal{N} = \frac{1}{T}\int_0^T \mathrm{d}t \frac{\|H(t)\|}{N}, \tag{3.13.6}$$

where $H(t)$ is a unified notation for the continuous protocols in the case of adiabatic or CD driving, and the piecewise-constant (in time) sequences in CD-QAOA and conventional QAOA; $\|H\|$ denotes the Hilbert-Schmidt norm of the operator $H$. Since we are interested in many-body systems, it is also natural to look at the energy density, i.e. $\|H(t)\|/N$. Figure 3.23 [bottom] shows that $\mathcal{N}$ is on a similar scale for all four methods within the range of durations of interest, which allows for a meaningful comparison between them. As expected, CD-driving approaches adiabatic driving at large $T$, since the gauge potential term comes with a pre-factor $\dot{\lambda}$ which vanishes for $T \to \infty$; in the opposite limit of $T \to 0$, the energy budget of CD-driving blows up, as a result of $\beta_j(\lambda)$ being unconstrained.

In Fig. 3.23 [top], we see that the many-body fidelity, associated with the protocols obtained using energy density minimization, increases the performance contrast between the performance of the different methods [cf. Fig. 3.7, main text]. Since the fidelity is defined as the overlap square of the final with the target states [Eq. equation 3.13.3], like the entanglement entropy, it is insensitive to any specific observable; this implies that CD-QAOA outperforms the other three methods on *all* observables, not just energy. This is anticipated, because CD-QAOA combines the variational power of QAOA with physical insights from CD driving. Despite its better performance, notice how CD-QAOA also has a smaller energy budget than either of CD- and adiabatic driving.

To demonstrate the nonequilibrium character of the optimal protocols found by the RL agent in this setup, we fix $T = 12$, and look at the time evolution of the energy, the fidelity, and the entanglement entropy within the learned protocol, cf. Fig. 3.24. While the protocol sequence [Table 3.5(a)] appears impenetrable, we remark that (i) the RL agent makes use of both single-particle and two-body gauge potential terms, and (ii) some step durations $\alpha_j$ are found to vanish identically, suggesting that the value of $q$ may be reduced. As anticipated, the behavior of the dynamics generated by the CD and adiabatic driving is smooth, in contrast to the circuit-like piece-wise

continuous curves of QAOA and CD-QAOA. The highly non-monotonic behavior of the energy curve shows that the CD-QAOA dynamics can be highly nonequilibrium: this likely stems from the RL objective [cf. Sec. 3.8] – the total expected return: the agent only cares about maximizing the reward at $t = T$ and is insensitive to any intermediate values. This allows the agent to drive the system through various states which are very far away from the target (e.g. w.r.t. the fidelity) [Curiously, these bad-energy states are all distinct, since they have different entanglement entropy, and the system does not visit the same quantum state twice during the evolution]. The non-smooth and non-monotonic behavior of the CD-QAOA solution raises the question about how robust the protocol is, to small external perturbations – a topic of future studies.

(a) **Ising** spin-1

| **Hamiltonian** | **Duration** |
|---|---|
| $X\|Y$ | 0.312 |
| $Y$ | 0.299 |
| $Z$ | 0.216 |
| $Y$ | 0.717 |
| $Z$ | 0.000 |
| $Y$ | 0.537 |
| $Z\|Z+X$ | 0.477 |
| $Y$ | 0.054 |
| $Z\|Z+X$ | 0.657 |
| $Z$ | 0.000 |
| $Z\|Z+X$ | 0.269 |
| $Y\|Z$ | 0.274 |
| $Z\|Z+X$ | 0.478 |
| $Y\|Z$ | 0.372 |
| $Z\|Z+X$ | 0.000 |
| $Z$ | 1.794 |
| $X\|Y$ | 0.072 |
| $Z$ | 0.039 |
| $Y$ | 1.007 |
| $Z$ | 4.426 |

(b) **Ferromagnetic** ($\Delta/J = -2.0$)

| **short-hand notation** | **Duration** |
|---|---|
| $Y\|Z-YZ$ | 0.122 |
| $X\|X+Y\|Y$ | 0.178 |
| $YZ$ | 0.027 |
| $Z\|Z$ | 0.376 |
| $Y\|Z-YZ$ | 0.234 |
| $Z\|Z$ | 0.000 |
| $X\|X+Y\|Y$ | 0.323 |
| $Z\|Z$ | 0.284 |
| $Y\|Z-YZ$ | 0.366 |
| $Z\|Z$ | 0.000 |
| $X\|X+Y\|Y$ | 0.314 |
| $Z\|Z$ | 0.188 |
| $Y\|Z-YZ$ | 0.535 |
| $Y$ | 0.001 |
| $X\|X$ | 0.342 |
| $Z\|Z$ | 0.105 |
| $Y\|Z-YZ$ | 0.538 |
| $X\|X$ | 0.208 |
| $Y$ | 0.000 |
| $Z\|Z$ | 0.051 |
| $Y$ | 0.658 |
| $Y\|Z-YZ$ | 0.002 |
| $Y$ | 0.900 |
| $Z$ | 0.771 |
| $Y$ | 0.005 |
| $X\|Y-XY$ | 0.474 |
| $Y\|Z-YZ$ | 0.000 |
| $X\|X+Y\|Y$ | 0.000 |

Table 3.5: Ising spin-1 chain and Anisotropic Heisenberg spin-1 chain: the protocol sequences and corresponding durations given by CD-QAOA. The protocol (a) correspond to Ising spin-1 in Fig. 3.24; the (b) corresponds to the Ferromagnetic phase in the same setting as Fig. 3.20. The short-hand notation is the same in Table 3.1. We use a shaded cell background whenever terms from the CD gauge potential are used in the protocol sequence. Terms of zero durations are marked in light grey.

(a) **XY** ($\Delta/J=-0.5$)

| **short-hand notation** | **Duration** |
|---|---|
| $Y$ | 0.795 |
| $X\|X$ | 0.000 |
| $Y$ | 0.772 |
| $X\|X+Y\|Y$ | 0.143 |
| $X\|X$ | 0.383 |
| $Y$ | 0.001 |
| $X\|X+Y\|Y$ | 0.284 |
| $X\|X$ | 0.180 |
| $X\|X+Y\|Y$ | 0.467 |
| $X\|X$ | 0.113 |
| $X\|X+Y\|Y$ | 0.635 |
| $X\|X$ | 0.097 |
| $X\|X+Y\|Y$ | 0.617 |
| $Y$ | 0.000 |
| $Z\|Z$ | 0.162 |
| $X\|X+Y\|Y$ | 0.265 |
| $X\|X$ | 0.092 |
| $Z$ | 1.995 |

(b) **Haldane** ($\Delta/J=0.5$)

| **short-hand notation** | **Duration** |
|---|---|
| $X\|X+Y\|Y$ | 0.149 |
| $X\|X$ | 0.000 |
| $X\|X+Y\|Y$ | 0.052 |
| $Z\|Z$ | 1.376 |
| $X\|X+Y\|Y$ | 0.313 |
| $Z\|Z$ | 0.668 |
| $X\|X+Y\|Y$ | 0.187 |
| $Z\|Z$ | 0.723 |
| $X\|X+Y\|Y$ | 0.289 |
| $Z\|Z$ | 0.528 |
| $X\|X+Y\|Y$ | 0.218 |
| $Z\|Z$ | 0.561 |
| $X\|X+Y\|Y$ | 0.254 |
| $Z\|Z$ | 0.684 |
| $X\|X+Y\|Y$ | 0.360 |
| $Z\|Z$ | 0.639 |
| $X\|X$ | 0.000 |
| $Z$ | 0.000 |

Table 3.6: Anisotropic Heisenberg spin-1 chain: the protocol sequences and corresponding durations given by CD-QAOA. The (a), (b) correspond to the two phases (XY and Haldane) in the same setting as Fig. 3.20. The short-hand notation is the same in Table 3.1. We use a shaded cell background whenever terms from the CD gauge potential are used in the protocol sequence. Terms of zero durations are marked in light grey.

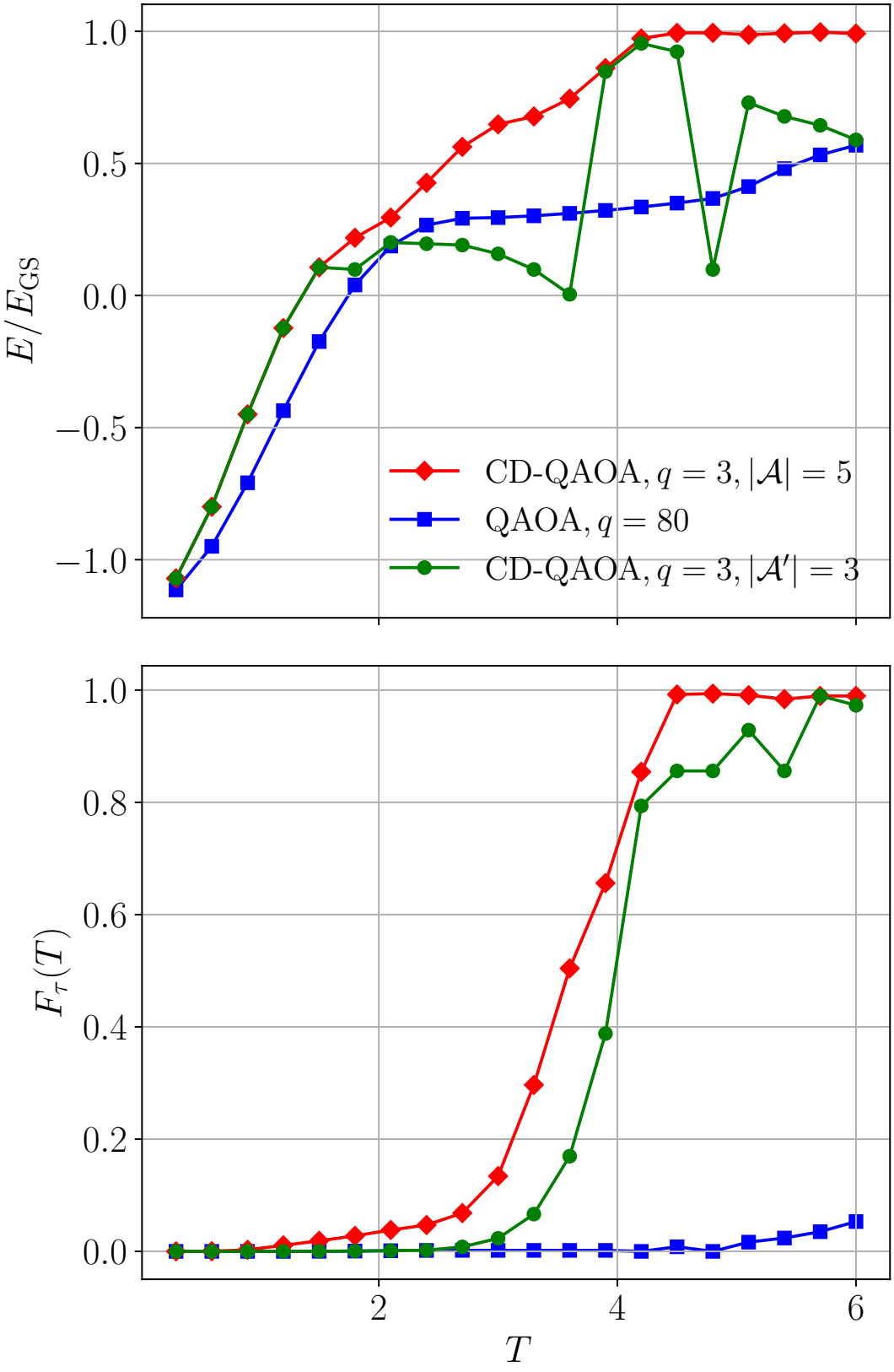



Figure 3.17: Spin-1/2 Ising model: energy minimization (top) and many-body fidelity maxi-

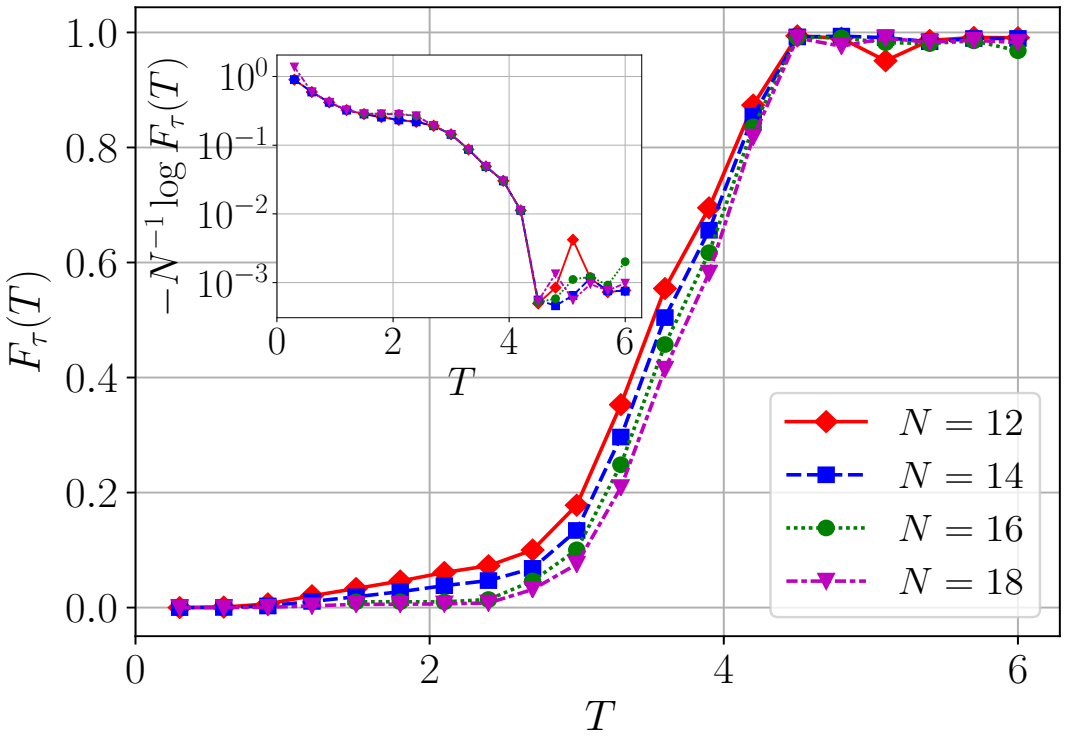



Figure 3.18: Spin-1/2 Ising model: many-body fidelity maximization and corresponding quantity [inset, log scale] against protocol duration $T$ for different system sizes $N$. The QAOA parameters are $q=3$ and $\mathcal{A} = \{Z|Z+Z, X; Y, X|Y, Y|Z\}$. The model parameters are the same as in Fig. 3.1.

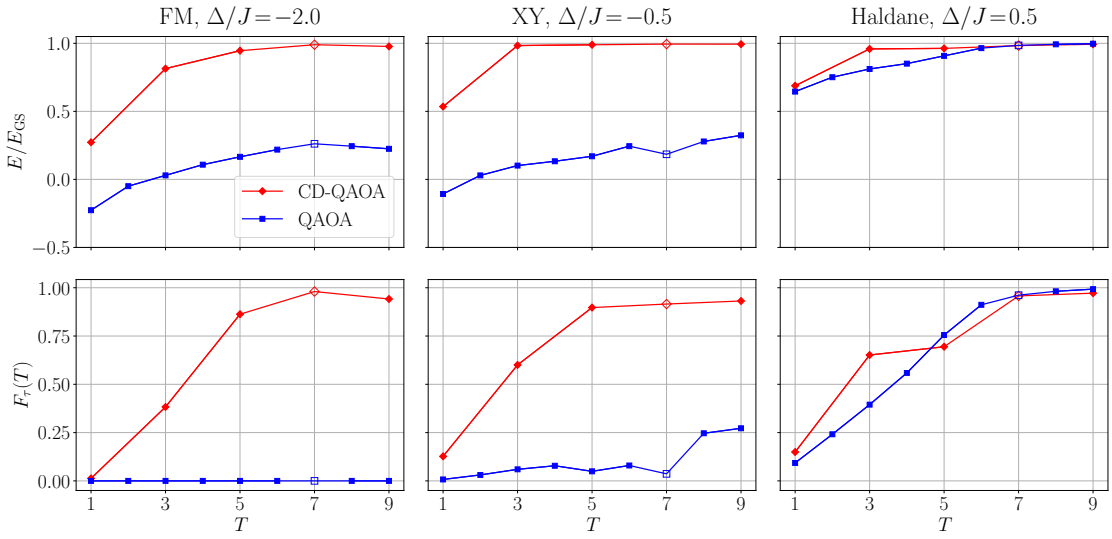



Figure 3.19: Anisotropic Heisenberg spin-1 chain: energy minimization against protocol duration $T$ — the corresponding energy (top row) and many-body fidelity (bottom row) for three ordered target states, corresponding to the ground state of the ferromagnetic (left, $\Delta/J=-2.0$), XY (middle, $\Delta/J=-0.5$), and Haldane (right, $\Delta/J=0.5$) target states, respectively. The empty symbols mark the duration at which we show the evolution of the system in Fig. 3.20. The model parameters are the same as in Fig. 3.3.

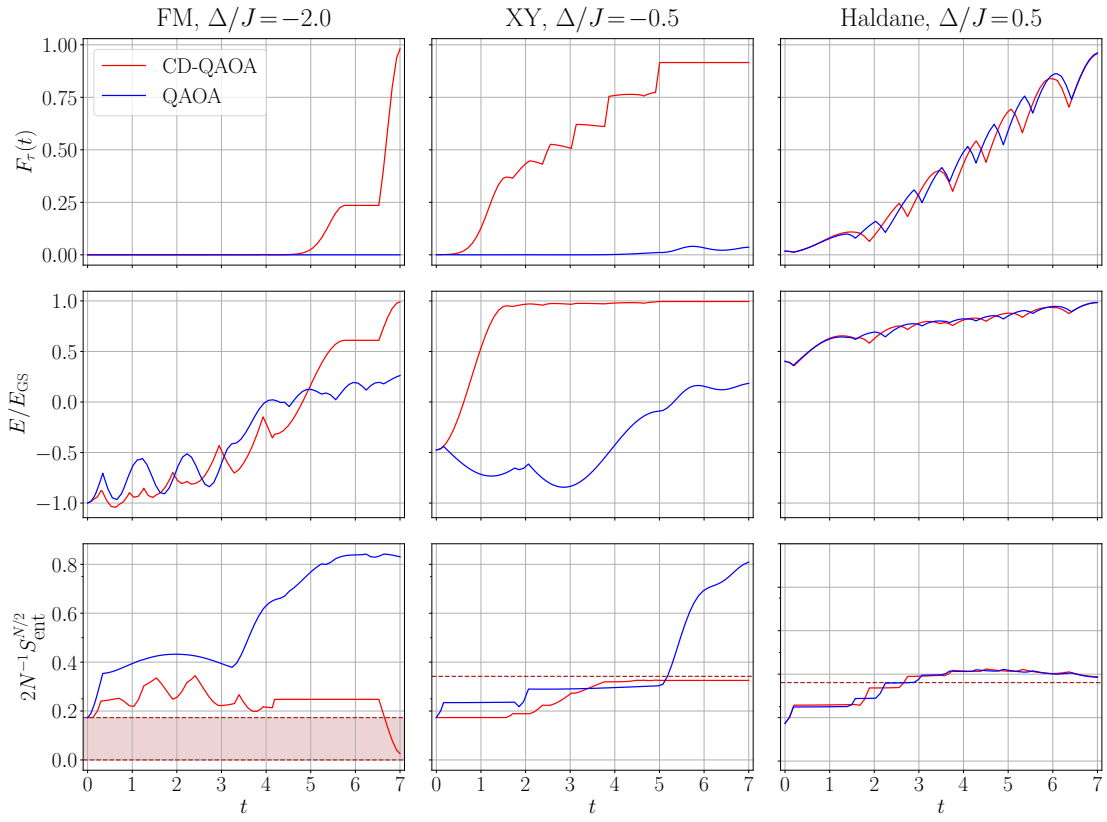



Figure 3.20: Anisotropic Heisenberg spin-1 chain: time evolution generated by the protocol given by CD-QAOA (blue line), and conventional QAOA (red line) for the three target states, corresponding to the ferromagnetic ($\Delta/J=-2.0$), XY ($\Delta/J=-0.5$), and Haldane ($\Delta/J=0.5$) target state, respectively. Three quantities are shown: many-body fidelity (first row), energy ratio (second row), and the entanglement entropy density of the half chain (third row). The horizontal dashed line in the entanglement entropy curve shows the value in the target state, while the shaded area for the FM state denotes that in the span of the doubly degenerate ground state manifold. The protocols correspond to the duration $T = 7$ in Fig. 3.3. The related CD-QAOA protocol sequences are given in Table 3.5(b) [ferromagnetic ($\Delta/J=-2.0$)], Table 3.6(a) [XY ($\Delta/J=-0.5$)] and Table 3.6(b) [Haldane ($\Delta/J=0.5$)]. The simulation parameters are the same as in Fig. 3.3.

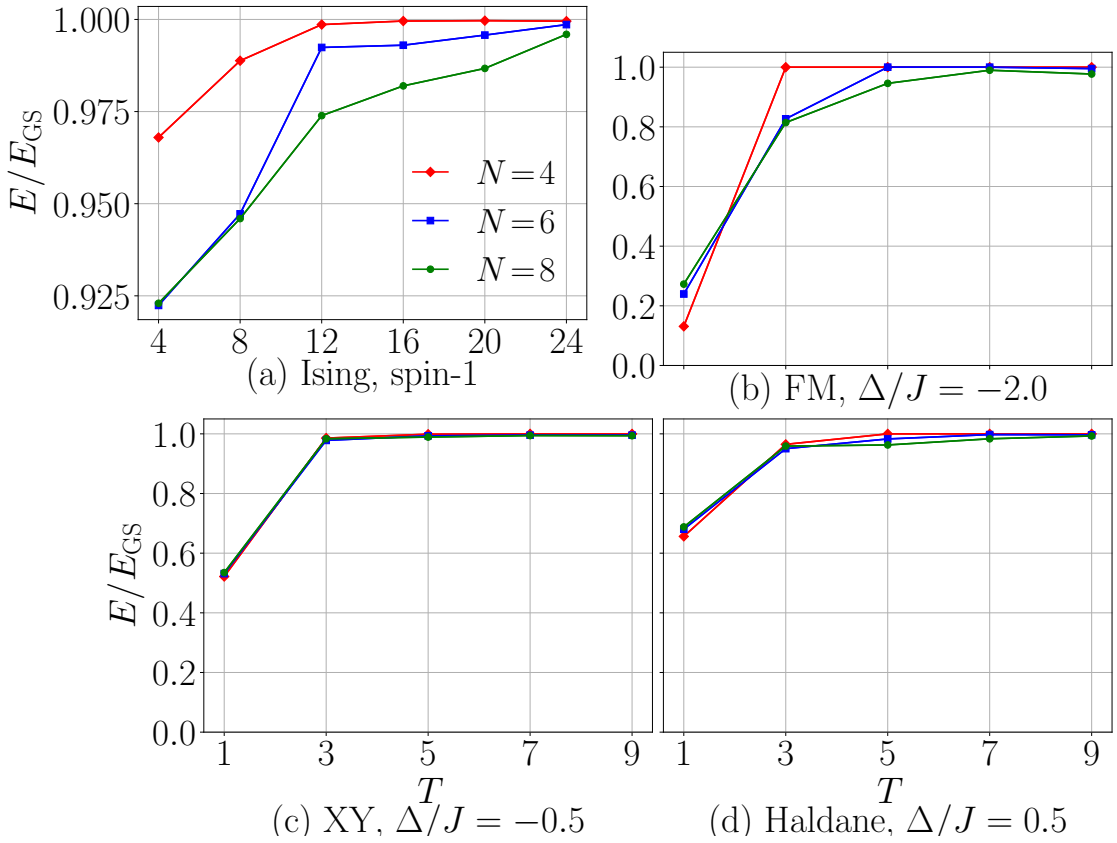



Figure 3.21: system-size scaling of the energy minimization against protocol duration $T$ for different system sizes $N$: (a) spin-1 Ising chain, (b-d) anisotropic Heisenberg spin-1 chain for $\Delta/J=-2.0$, $\Delta/J=-0.5$, $\Delta/J=0.5$, respectively. Note that the $y$-axis scale is different for the spin-1 Ising model in panel (a). The model parameters are the same as in (a) Fig. 3.7 and (b-d) Fig. 3.3 correspondingly.

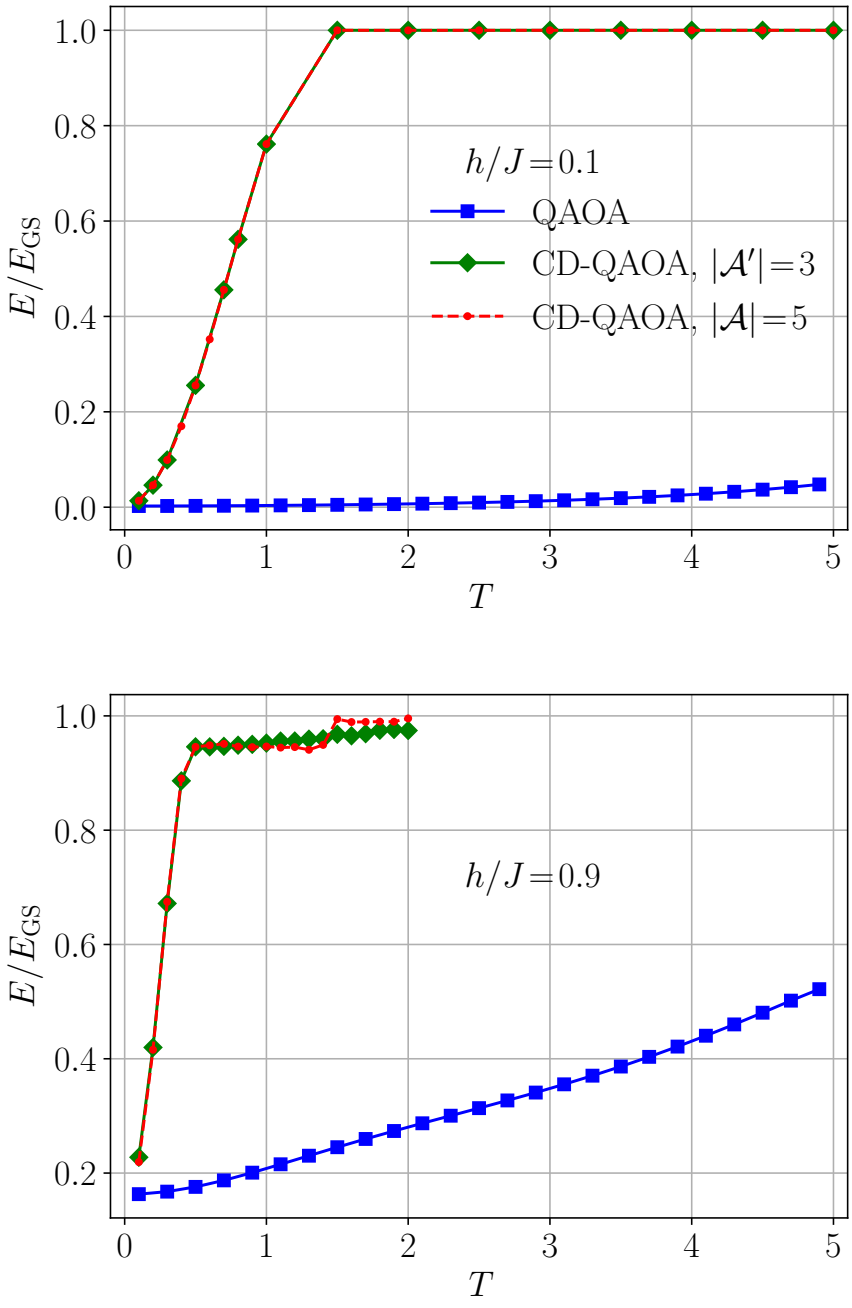



Figure 3.22: LMG model: energy minimization against protocol duration $T$ using conventional QAOA (blue square) and CD-QAOA (red dashed line, green solid line). The model parameters are the same from the settings in Fig. 3.5 but for $h/J=0.1$ (top panel), and $h/J=0.9$ (bottom panel).

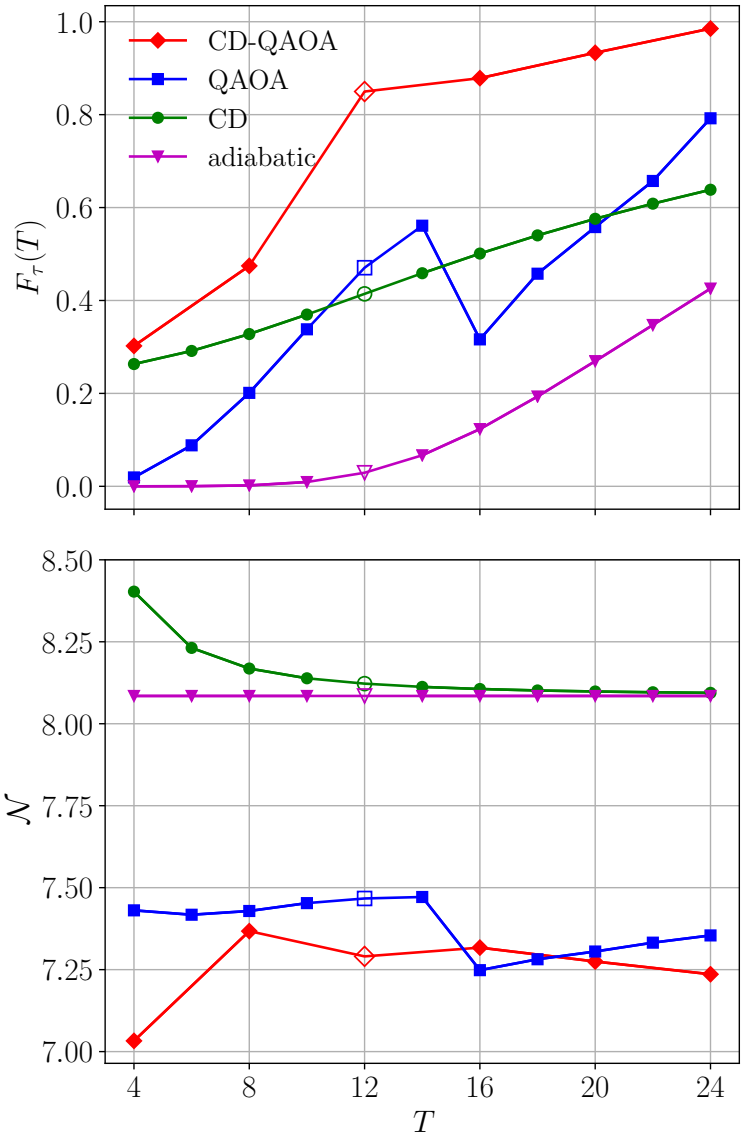



Figure 3.23: Spin-1 Ising model: energy minimization against different protocol duration $T$ for four different optimization methods: CD-QAOA (red line), conventional QAOA (blue line), variational gauge potential (green) and adiabatic evolution (magenta). Two associated quantities are shown: many-body fidelity $F_\tau$ (top) and normalized time-averaged energy density $\mathcal{N}$ over the protocol (bottom). The empty symbols mark the duration for which the evolution of physical quantities is shown in Fig. 3.24. The parameters are the same as in Fig. 3.7.

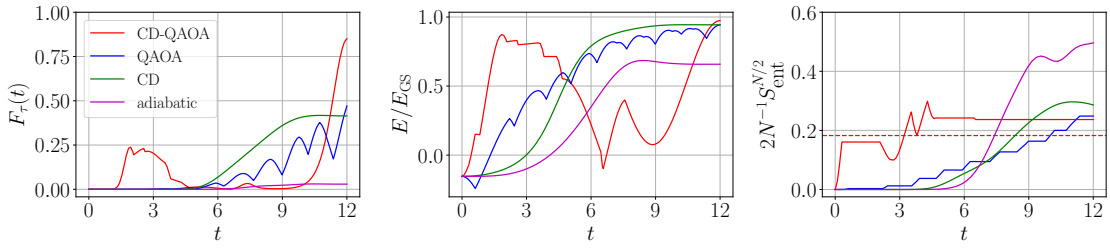



Figure 3.24: Spin-1 Ising model: time evolution generated by the four different methods: CD-QAOA (red), conventional QAOA (blue), CD driving using the variational gauge potential (green) and adiabatic evolution (magenta). The three quantities are shown: the many-body fidelity (left), energy (middle), and entanglement entropy of the half chain (right). The protocols correspond to the empty symbols during $T = 12$ in Fig. 3.7. We compare The horizontal dashed line in the entanglement entropy curve shows the value in the target state. The CD-QAOA protocol sequence is given in Table 3.5(a). The model parameters are the same as in Fig. 3.7.

# Chapter 4

# Noise-Robust Deep Autoregressive Policy Networks based variational algorithms

Variational quantum eigensolvers have recently received increased attention, as they enable the use of quantum computing devices to find solutions to complex problems, such as the ground energy and ground state of strongly-correlated quantum many-body systems. In many applications, it is the optimization of both continuous and discrete parameters that poses a formidable challenge. Using reinforcement learning (RL), we present a hybrid policy gradient algorithm capable of simultaneously optimizing continuous and discrete degrees of freedom in an uncertainty-resilient way. The hybrid policy is modeled by a deep autoregressive neural network to capture causality. We employ the algorithm to prepare the ground state of the nonintegrable quantum Ising model in a unitary process, parametrized by a generalized quantum approximate optimization ansatz: the RL agent solves the discrete combinatorial problem of constructing the optimal sequences of unitaries out of a predefined set and, at the same time, it optimizes the continuous durations for which these unitaries are applied. We demonstrate the noise-robust features of the agent by considering three sources of uncertainty: classical and quantum measurement noise, and errors in the control unitary durations. Our work exhibits the beneficial synergy between reinforcement learning and quantum control.

## 4.1 Introduction

The last decade has seen impressive breakthroughs in Machine Learning (ML), ranging from image classification [268, 180] to mastering complex video and board games [221, 284]. ML algorithms have opened the door to solving major scientific challenges, hitherto considered intractable, such as protein modelling [256] and folding [152], or molecular dynamics simulations [202].

Deep learning tools and methods quickly found their way into the field of physics [98, 215, 55, 56]: Supervised learning was found efficient in identifying phase transitions and analyzing experimental data [57, 313, 36, 262]. Unsupervised learning brought a new class of variational many-body wavefunctions [54], as well as methods to perform tomography on many-body quantum states [309], find conservation laws from data [147], identify phase transitions [321, 177], Hamiltonian learning [311], etc. Reinforcement learning (RL) [295] brought strategies for navigating turbulent flows [261, 73, 25], and even exploring the string landscape [131].

The variational character of ML models combined with their intrinsic optimization procedure, provide a natural playground for applications in quantum control [273, 318, 271, 105, 228, 4, 287, 336, 339, 10]. Due to the close relationship between control theory and reinforcement learning, the control of quantum systems has become a major application area of RL algorithms in physics. Notable examples include policy gradient [237, 107, 19, 249, 325, 346, 292], Q-learning [65, 48, 50, 288, 37] and AlphaZero [78].

Over the years, the physics community has also developed a number of successful quantum control algorithms [161, 53, 244, 79, 204, 205], including GRAPE, CRAB, and VQE. A prominent example of the latter is Quantum Approximate Optimization Algorithm (QAOA) [102], whose versatility allows for solving complex combinatorial problems using quantum computers [110, 97, 160, 159, 347, 300, 41]. Quantum control algorithms, such as CRAB or QAOA, come with an ingenious physics-informed variational ansatz for the structure of control protocols. RL algorithms, on the other hand, are model-free and resilient to uncertainty. Hence, a natural question emerges as to how one can combine the benefits offered by RL and quantum control in a unified framework.

In this paper, our aim is to deploy a generalized QAOA ansatz in combination with an end-to-end deep RL algorithm for a versatile continuous-discrete quantum control [Sec. 4.2]. We adopt the continuous degrees of freedom of QAOA which offer an increased control accuracy. Additionally, we consider an enhanced variational control ansatz which contains a larger space to select the building blocks of the

protocols from; this introduces a second, discrete combinatorial optimization problem. The resulting algorithm, RL-QAOA, realizes greater gains by striking a balance between robustness and versatility: it is resilient to various kinds of uncertainty, a property shared with PG-QAOA [346]; at the same time, RL-QAOA has access to the more general variational counter-diabatic (CD) driving ansatz [86, 208, 122] through CD-QAOA [347].

However, RL-QAOA presents a number of new challenges, cf. Sec. 4.3. It requires a mixed continuous-discrete action space so that the RL agent can construct a control protocol by optimizing the order in which unitaries appear in the control sequence; simultaneously, the agent has to also choose the continuous duration to apply each unitary. This requires the use of a suitable ML model to approximate the policy, which allows us to build in temporal causality. Therefore, an essential building block of RL-QAOA is a novel monolithic deep autoregressive policy network that handles continuous and discrete actions on equal footing. To train our RL agent, we derive an extension of Proximal Policy Optimization (PPO) [278] to hybrid discrete-continuous policies.

We apply RL-QAOA to find the ground state of a nonintegrable chain of interacting spin-1/2 particles (a.k.a. qubits) in a fixed amount of time, cf. Sec. 4.4. The mixed discrete-continuous degrees of freedom allow the RL agent to construct a short protocol sequence away from the adiabatic regime. We test the agent's behavior in a strongly stochastic environment, by considering three different kinds of noise: classical and quantum measurement noise, and errors in the control unitary gate duration. In Sec. 4.5, we demonstrate that RL-QAOA is insensitive to the types of noise applied, and outperforms previously developed algorithms based on QAOA in the regime of strong noise.

## 4.2 Preliminaries

We start the discussion by introducing the QAOA ansatz used in quantum control. Following a short overview of reinforcement learning terminology, we review two RL-based QAOA algorithms — PG-QAOA and CD-QAOA — which we aim to blend into a homogeneous hybrid in Sec. 4.3. The resulting new algorithm combines the benefits of the generalized variational QAOA ansatz, with an RL algorithm performing both continuous and discrete control simultaneously.

## QAOA for Ground State Preparation

Of particular interest in the quest for designing new materials with novel features (such as room-temperature superconductors, or topological quantum computers), is the study of ground state properties in quantum many-body physics. Quantum simulators provide an ideal platform to bring together both theory and experiment; yet, they require the ability to prepare a system in its ground state – a formidable challenge for modern quantum computing devices, due to the presence of various sources of uncertainty and noise. The Quantum Approximate Optimization Algorithm (QAOA) [102] provides a widely used state-of-the-art ansatz for this purpose.

Consider a quantum system of $N$ qubits, described by the Hamiltonian $H$. Starting from an initial quantum state $|\psi_i\rangle$, in QAOA we apply two alternating unitary evolution operators (i.e. quantum gates) [102]:

$$|\psi(T)\rangle = U(\{\alpha_j,\beta_j\}_{j=1}^{p})\,|\psi_i\rangle = e^{-iH_2\beta_p}e^{-iH_1\alpha_p}\cdots e^{-iH_2\beta_1}e^{-iH_1\alpha_1}\,|\psi_i\rangle\,. \tag{4.2.1}$$

The dynamics are generated by the time-independent operators $H_1$ and $H_2$, applied for a duration of $\alpha_j \geq 0$ and $\beta_j \geq 0$, respectively ($j = 1, 2, \cdots, p$ with $p \in \mathbb{N}$). We refer to $q = 2p$ as the total circuit depth. In order to apply QAOA to many-body systems [141], the protocol durations $\{(\alpha_j,\beta_j)\}_{j=1}^{p}$ are variationally optimized to minimize the expected value of the energy density $\mathcal{E}(\{\alpha_j,\beta_j\}_{j=1}^{p}) = N^{-1}\,\langle\psi(T)|H|\psi(T)\rangle$ :

$$\{\alpha_j^*,\beta_j^*\}_{j=1}^{p} = \underset{\{\alpha_j,\beta_j\}_{j=1}^{p}}{\arg\min}\ \mathcal{E}(\{\alpha_j,\beta_j\}_{j=1}^{p}), \quad \sum_{j=1}^{p}(\alpha_j+\beta_j) = T. \tag{4.2.2}$$

The additional constraint $\sum_{j=1}^{p}(\alpha_j+\beta_j) = T$ is required for the resulting protocol to remain in the regime of practical applications, and also for a fair comparison between different algorithms.

As a concrete example to keep in mind, consider the spin-1/2 Ising Hamiltonian

$$H = H_1 + H_2, \qquad H_1 = \sum_{i=1}^{N} J S_{i+1}^z S_i^z + h_z S_i^z, \quad H_2 = \sum_{i=1}^{N} h_x S_i^x, \tag{4.2.3}$$

where $[S_k^\alpha, S_j^\beta] = i\delta_{kj}\varepsilon^{\alpha\beta\gamma}S_j^\gamma$ are the spin-1/2 operators. We are interested in preparing the ground state of $H$, starting from a spin-up polarized initial product state. More details about the physical system are discussed later on in Sec. 4.4 and Sec. 4.9.

## Reinforcement Learning (RL)

While QAOA defines a variational ansatz to prepare ground states in a unitary process, it does not yet provide a self-contained optimization procedure to find the optimal protocol durations. A universal optimization framework is presented by RL [295].

Reinforcement learning comprises a powerful set of algorithms designed to solve control problems. In RL, an agent aims to find a policy $\pi$ which solves a specific task in a trial-and-error approach based on interactions with the agent's environment. Consider a finite-horizon Markov Decision Process (MDP) defined by the tuple $(\mathcal{S}, \mathcal{A}, p, r)$ where $\mathcal{S}$ and $\mathcal{A}$ are the state and action spaces, respectively, and $p : \mathcal{S} \times \mathcal{S} \times \mathcal{A} \to [0, 1]$ defines the transition probability which governs the environment dynamics. Upon selecting an action $a \in \mathcal{A}$, the environment transitions to a new state $s \to s' \in \mathcal{S}$, and emits a reward $r : S \times A \to \mathbb{R}$, which the RL agent uses to select subsequent actions. The action $a_j \in \mathcal{A}$ to be selected in a given state $s \in \mathcal{S}$ is determined probabilistically by the instantaneous policy $\pi(a_j|s_j) : \mathcal{A} \times \mathcal{S} \to [0, 1]$. For a given policy $\pi$, this process generates a trajectory $\tau = (s_1, a_1, ...., a_q, s_{q+1})$ with probability $\tau \sim \mathbb{P}^\pi(\cdot)$. Here, $\mathbb{P}^\pi(\tau) = p_0(s_1)\pi(a_1|s_1)p(s_2|s_1, a_1) \cdots \pi(a_q|s_q)p(s_{q+1}|s_q, a_q)$, the episode/trajectory length is $q$, and $p_0$ is the initial state distribution. The objective in RL is to find the optimal policy, i.e. the policy which maximizes the total expected return: $\mathbb{E}_{\tau\sim\mathbb{P}^\pi}\left[\sum_{j=1}^{q} r(s_j, a_j)\right]$.

## Policy Gradient Quantum Approximate Optimization Algorithm (PG-QAOA)

A reinforcement learning based approach to QAOA was recently introduced in Ref. [346], using a policy gradient algorithm. The basic idea behind PG-QAOA is to let the RL agent select the durations $\{\alpha_j, \beta_j\}$, which constitute a continuous action space $\mathcal{A}^c$. However, casting the quantum control problem within the RL framework comes with certain challenges. The first challenge is that quantum states cannot be directly measured in experiments, which poses questions about the proper definition of the RL state space. To remedy this in an environment following deterministic Schrödinger dynamics, it was suggested to fix the initial quantum state, and define the RL state as the trajectory of actions $s_j = (a_1^c, \cdots, a_{j-1}^c) = (\alpha_1, \beta_1, \cdots)$ up to episode step $j$ [48]; this definition is inferior to using the full state, but it allows to accommodate the experimental constraint, so we adopt it in this study as well. Alternatively, one could use the expectation values of observables to define an RL state [325]. The second challenge is the sparsity of the reward signal – a quantum measurement is allowed only once at the end of each episode, since projective measurements collapse the

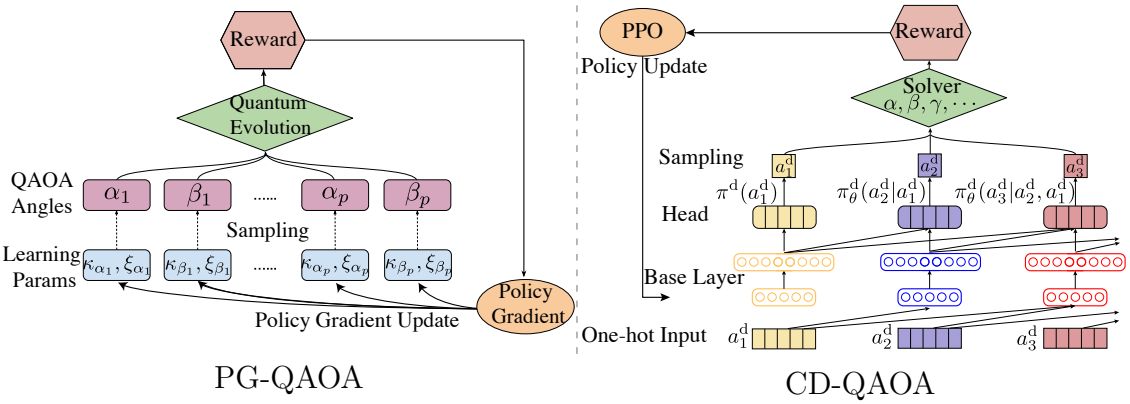



Figure 4.1: Schematic diagram for PG-QAOA [left, see Sec. 4.2] and CD-QAOA [right, see Sec. 4.2]. The PG-QAOA samples continuous QAOA-angles from its policy and variationally updates the policy parameters via policy gradient; CD-QAOA autoregressively samples the gate sequences for the generalized QAOA ansatz and employs the gradient-free solver (Powell algorithm) to solve for their corresponding durations. The policy network is updated via Proximal Policy Optimization (PPO). For a comparison with RL-QAOA, cf. Fig. 4.2.

quantum wavefunction and the quantum state is lost irreversibly.

Since the protocol durations are continuous degrees of freedom, we need an RL method for *continuous* optimization. PG-QAOA defines the simplest ansatz: $q=2p$ independent Gaussian distributions to parameterize the policy, one for each duration $\{\alpha_j, \beta_j\}_{j=1}^{p}$ in Eq. equation 4.2.2. Since a Gaussian distribution is uniquely determined by its mean $\mu$ and standard deviation $\sigma$, we need a total of $2p$ independent variational parameters $\boldsymbol{\theta} = \{\mu_{\alpha_j}, \sigma_{\alpha_j}, \mu_{\beta_j}, \sigma_{\beta_j}\}_{j=1}^{p}$ to parametrize the policy $\pi_{\boldsymbol{\theta}}$ as:

$$\pi_{\boldsymbol{\theta}}(\{\alpha_j, \beta_j\}_{j=1}^{p}) = \prod_{j=1}^{p} \pi(\alpha_j; \kappa_{\alpha_j}, \xi_{\alpha_j}) \pi(\beta_j; \kappa_{\beta_j}, \xi_{\beta_j}), \tag{4.2.4}$$

where $\kappa_{\alpha_j} = \mu_{\alpha_j}$, $\kappa_{\beta_j} = \mu_{\beta_j}$ are the means, and $\xi_{\alpha_j} = \sigma_{\alpha_j}$, $\xi_{\beta_j} = \sigma_{\beta_j}$ are the variances of the Gaussian policy. The actual protocol durations are thus sampled according to $\alpha_j \sim \mathcal{N}(\mu_{\alpha_j}, \sigma_{\alpha_j}^2)$, and similarly for $\beta_j$. As was shown in Ref. [346], despite its simplicity, PG-QAOA defines a particularly noise-robust algorithm. In the presence of various kinds of noise, it readily outperforms a number of alternative gradient-free optimization algorithms.

For this study, the original PG-QAOA implementation [346] is not directly applicable, and a modification is required. First, the extensive scaling with increasing the number of qubits suggests us to use as a cost function the energy density, rather than the

many-body fidelity; in doing this, the algorithm no longer requires an explicit reference to the target ground state we are searching for. Second, the original PG-QAOA algorithm does not support an easy implementation of the protocol duration constraint $\sum_{j=1}^{p}(\alpha_j+\beta_j)=T$. Here, in order to do a fair comparison among different algorithms, we enforce this constraint. Note that this is a non-trivial task for policy gradient, for three reasons: (i) protocol durations are sampled from a Gaussian distribution which has unbounded support, (ii) a Gaussian policy supports negative as well as positive samples (yet we require $\alpha_j, \beta_j \geq 0$ for a physical time duration), and (iii) sampled values, even if bounded and nonnegative, are always random, and hence one needs to additionally fix their total sum. We consider two different approaches to resolving (i) and (ii), and apply a normalization trick to fix (iii).

The first approach we consider is to define the policy using the Beta distribution [72], i.e. $\alpha_j, \beta_j \sim \mathrm{B}(\kappa, \xi)$, instead of a Gaussian, and learn the two nonnegative parameters $\kappa, \xi$. Since the Beta distribution is defined on the interval $\mathcal{A}^{\mathrm{c}} = [0, 1]$, it solves the boundedness and positivity problems. The policy is given by Eq. equation 4.2.4 with $\pi(x; \kappa, \xi) = \frac{\Gamma(\kappa+\xi)}{\Gamma(\kappa)\Gamma(\xi)} x^{\kappa-1}(1-x)^{\xi-1}$ the probability density for the Beta distribution; $\Gamma$ denotes the Gamma function. Note that the number of independent variational parameters $\boldsymbol{\theta} = \{\kappa_{\alpha_j}, \xi_{\alpha_j}, \kappa_{\beta_j}, \xi_{\beta_j}\}_{j=1}^{p}$, remains equal to $4p$.

In the second approach, we pass the output of the Gaussian distribution through a sigmoid activation function [126]. Due to the boundedness of the sigmoid function, this restricts the range of all actions/durations to the nonnegative interval $\mathcal{A}^{\mathrm{c}} = [0, 1]$. Hence, our policy is given by Eq. equation 4.2.4 where $\pi(x; \kappa, \xi) = \frac{1}{x(1-x)} \frac{1}{\sqrt{2\pi\xi^2}} \exp\left(-\frac{(\mathrm{logit}(x)-\kappa)^2}{2\xi^2}\right)$ is the probability density for the Sigmoid Gaussian distribution $\mathcal{SN}(\kappa, \xi^2)$[1]. Here, the logit function, $\mathrm{logit}(x) = \log x - \log(1-x)$, is the inverse of the sigmoid function $f(x) = 1/(1+\exp(-x))$, and the factor $\frac{1}{x(1-x)}$ is the inverse Jacobian of $x = f(y)$ over $y$ [cf. Sec. 4.8]. Notice how, the action output of this policy is forced within the interval $[0, 1]$ by construction, without changing the total number of independent variational parameters $\boldsymbol{\theta}$.

Finally, to fix the total protocol duration, (iii), we normalize the sum of durations manually according to $\alpha_j = \frac{\alpha_j}{\sum_{j=1}^{p}(\alpha_j+\beta_j)} T$, $\beta_j = \frac{\beta_j}{\sum_{j=1}^{p}(\alpha_j+\beta_j)} T$. We note that the normalization procedure is considered part of the RL environment, i.e. no gradients are passed through it. In essence, it becomes part of the reward function. This requires us to slightly re-define the meaning of the policy: it generates the bare protocol durations before the normalization; to minimize energy, the durations need the extra

[1] $\mathcal{SN}(\kappa, \xi^2)$ is short-hard notation for Gaussian distribution $\mathcal{N}(\kappa, \xi^2)$ under the sigmoid transformation $f(x)$.

normalization. We mention in passing that this is not the only way to hold the protocol duration $T$ fixed: alternatives include using the constraint to fix the last protocol duration $\beta_p$, or the addition of an extra penalty term to the cost function.

## Quantum Approximate Optimization Ansatz based on Counter-Diabatic Driving

In conventional QAOA, there are two possible gates, corresponding to the two unitaries $U_j = \exp(-i\alpha_j H_j), j = 1, 2$. Therefore, there exist only two distinct sequences of unitaries: $\tau_1^{\mathrm{d}} = U_1U_2U_1U_2\cdots$ and $\tau_2^{\mathrm{d}} = U_2U_1U_2U_1\cdots$. A generalization of this ansatz was considered in Ref. [347], where an RL agent was given the complex combinatorial task to construct the sequence of unitaries $\tau^{\mathrm{d}}$, out of a predefined set $\mathcal{A}^{\mathrm{d}}$ of $|\mathcal{A}^{\mathrm{d}}|$ gates/unitaries. As for the gate duration notation, we will use $\alpha_j$ for all durations instead of alternating $\alpha_j$, $\beta_j$ due to a general ansatz. This set can, in principle, be chosen arbitrarily; however, one can also make a more physics-informed choice, e.g., inspired by counter-diabatic driving in the case of quantum-many-body systems [cf. Sec. 4.2]. In the latter case, the resulting generalized algorithm, called CD-QAOA, was demonstrated to drastically enhance the variational ansatz of QAOA when applied to many-body quantum chains, allowing for shorter circuit depths at no cost in performance [347].

Similar to PG-QAOA, CD-QAOA does *not* use the quantum wavefunction to perform the optimization, and the state is $s_j = (a_1^{\mathrm{d}}, \cdots, a_{j-1}^{\mathrm{d}})$ at episode step $j$. Rewards are given once per episode, in the end, and are defined by the (negative) energy density. However, the action space is given by the set of $|\mathcal{A}^{\mathrm{d}}|$ unitary gates from which the protocol sequence $\tau^{\mathrm{d}}$ are selected; it does not involve the continuous protocol durations which are found as part of the RL environment; to do this, in this study, we use the gradient-free Powell algorithm [250] instead of the gradient-based SLSQP algorithm [178] presented in the original CD-QAOA paper.

Apart from the low-level optimization mentioned above, CD-QAOA adopts a two-level optimization schedule [194, 216]: high-level discrete optimization is used to construct the optimal sequence $\tau^{\mathrm{d}}$ out of the available set of unitaries. For this purpose, in Ref. [347], it was suggested to employ Proximal Policy Optimization[278] (PPO), an advanced variant of policy gradient, aided by a deep autorgressive neural network to implement causality:

$$\pi_{\boldsymbol{\theta}}^{\mathrm{d}}\left(a_1^{\mathrm{d}}, a_2^{\mathrm{d}}, \cdots, a_q^{\mathrm{d}}\right) = \pi_{\boldsymbol{\theta}}^{\mathrm{d}}\left(a_1^{\mathrm{d}}\right) \prod_{j=2}^{q} \pi_{\boldsymbol{\theta}}^{\mathrm{d}}\left(a_j^{\mathrm{d}} \mid a_1^{\mathrm{d}}, \cdots, a_{j-1}^{\mathrm{d}}\right). \tag{4.2.5}$$

Each factor in the product above is a categorical distribution over the action space. We point out that the search for the optimal sequence $\tau^{\rm d}$ represents a *discrete* optimization problem. This should be contrasted with the low-level continuous optimization employed by QAOA to find the optimal durations $\{\alpha_j\}_{j=1}^q$, carried out using the Powell solver. Since the Powell solver only deals with the bounded optimization, we apply the same normalization trick to enforce the total duration constraint.

Given the complete protocol sequence $\tau^{\rm d} = (a_1^{\rm d}, \cdots, a_q^{\rm d})$, we can construct the unitary process

$$U(\{\alpha_j\}_{j=1}^q, \tau^{\rm d}) = \prod_{j=1}^{q} U_{\tau_j^{\rm d}}(\alpha_j) \tag{4.2.6}$$

which we use as a generalized QAOA ansatz. The sequence $\tau^{\rm d}$, and the durations $\{\alpha_j\}_{j=1}^q$ are found by minimizing the energy density, cf. Eq. equation 4.2.2. In doing so, we impose an extra constraint that the same action cannot be taken twice in a row, for otherwise one can add the corresponding durations and optimize them together.

### Possible Choice of Unitaries based on Counter-Diabatic Driving

In order to construct the unitary $U(\{\alpha_j\}_{j=1}^q, \tau^{\rm d}) = \prod_{j=1}^q \exp\Big(-i\alpha_j H_{\tau_j^{\rm d}}\Big)$ from Eq. equation 5.2.1, the RL agent needs to select the sequence $\tau^{\rm d}$ of subprocess generators $\mathcal{A}^{\rm d}$. Hence, at every step $j$ in the RL episode, the agent's action consists of a choice of a Hermitian operator $H_{\tau_j^{\rm d}} \in \mathcal{A}^{\rm d}$

The set of discrete actions $\mathcal{A}^{\rm d}$ consists of the available possible controls in an experiment. In Refs. [347, 140, 90], it was shown that a particularly suitable choice of actions for ground state preparation in quantum many-body systems, is given by terms appearing in the series of the variational adiabatic gauge potential, designed for many-body counter-diabatic driving [280]. These terms provide shortcuts in the Hilbert space that may significantly decrease the time required to prepare the ground state. For brevity, below we just list the generator set $\mathcal{A}^{\rm d}$ for the spin$-1/2$ Ising model, cf. Eq. equation 4.2.3, which the RL agent has access to, and refer the interested readers to Ref. [347] for more details: $\mathcal{A}^{\rm d} = \{H_1, H_2, Y, X|Y, Y|Z\}$, with $Y = \sum_i S_i^y$, $X|Y = \sum_i S_i^x S_{i+1}^y + S_i^y S_{i+1}^x$, and $Y|Z = \sum_i S_i^y S_{i+1}^z + S_i^z S_{i+1}^y$; $H_1, H_2$ are defined in Eq. equation 4.2.3.

We emphasize that this is just one particular choice for $\mathcal{A}^{\rm d}$. In practice, the algorithm is agnostic to the discrete action space which is determined by the available controls

| Method | QAOA | PG-QAOA | CD-QAOA | RL-QAOA |
|---|---|---|---|---|
| protocol sequence optimization (discrete) | ✗ | ✗ | $\nabla$-free | $\nabla$-free |
| gate durations optimization (continuous) | $\nabla$-free | $\nabla$-free | $\nabla$-free | $\nabla$-free |
| RL optimization | ✗ | continuous | discrete | continuous & discrete |
| noise-robust | ✗ | ✓ | ✗ | ✓ |
| autoregressive | ✗ | ✗ | ✓ | ✓ |

Table 4.1: Comparison between all four algorithms: QAOA, PG-QAOA, CD-QAOA and RL-QAOA.

for the system of interest: e.g., on a quantum computer, these can be the set of local gates, etc.

## 4.3 Mixed Discrete-Continuous Policy Gradient using Deep Autoregressive Networks

Although RL is used as an optimizer in both PG-QAOA and CD-QAOA, it serves two fundamentally different purposes. In PG-QAOA it is employed for continuous optimization of the protocol durations $\{\alpha_j\}$, while in CD-QAOA it is used to find the solution to the discrete combinatorial task of ordering the unitaries in the protocol sequence. In this section, we illustrate how to combine the two aspects together into a unified RL-based algorithm.

We have seen that with the help of RL one can tremendously enhance the properties of the QAOA ansatz in very different ways, cf. Table 5.1: For instance, PG-QAOA has the important desired property that it is robust to noise. Moreover, it does a completely gradient-free optimization of the continuous protocol durations. On the other hand, CD-QAOA, enhances the variational ansatz itself by offering the appealing ability to select the order in which three or more unitaries can be applied in the protocol sequence. Moreover, it also introduces an autoregressive deep neural network to encode causality (i.e., which unitary is optimal at a given episode step depends on the unitaries chosen hitherto). The imminent question arises as to whether we can design an algorithm which makes the best of both worlds.

## Autoregressive Policy Ansatz for Hybrid Discrete-Continuous Action Spaces

Recently, a number of studies have considered the problem of simultaneous discrete/continuous control using RL [181, 101, 326, 137, 84, 31, 343, 101, 326, 233]. Following the notation of Ref. [207], we describe the RL problem within the framework of parametrized-action Markov decision processes (PAMDPs). The major difference, compared to ordinary MDPs, is the definition of the action space: $\mathcal{A} = \bigcup_{a^{\mathrm{d}} \in \mathcal{A}^{\mathrm{d}}, a^{\mathrm{c}} \in \mathcal{A}^{\mathrm{c}}} (a^{\mathrm{d}}, a^{\mathrm{c}})$, $\mathcal{A}^{\mathrm{d}} = \{H_j\}_{j=1}^{|\mathcal{A}^{\mathrm{d}}|}$, $\mathcal{A}^{\mathrm{c}} = [0,1]$, where $|\mathcal{A}^{\mathrm{d}}|$ denotes the cardinality of the discrete action set. As before, the state space contains all possible sequences of actions, and the reward is the (negative) energy density of the quantum state, given once at the end of the protocol.

In this section, we present a unified continuous-discrete quantum control algorithm, called RL-QAOA, based on a hybrid policy which optimizes simultaneously the discrete and continuous degrees of freedom in the policy. The policy can be decomposed as a product of two coupled auxiliary policies – one for the continuous actions, $\pi^{\mathrm{c}}_{\boldsymbol{\theta}}$, and the other for the discrete actions, $\pi^{\mathrm{d}}_{\boldsymbol{\theta}}$:

$$\pi_{\boldsymbol{\theta}}(\tau) = \pi^{\mathrm{c}}_{\boldsymbol{\theta}}\left(\tau^{\mathrm{c}}\right) \pi^{\mathrm{d}}_{\boldsymbol{\theta}}\left(\tau^{\mathrm{d}}\right), \tag{4.3.1}$$

where $\tau^{\nu} = (a^{\nu}_1, \ldots, a^{\nu}_q)$, $\nu \in \{\mathrm{c}, \mathrm{d}\}$ defines the discrete/continuous subsequence of actions in each trajectory of length $q$. Denoting, as before, the RL state by $s_j = (a_1, \cdots, a_{j-1})$ with the hybrid action $a_* = (a^{\mathrm{c}}_*, a^{\mathrm{d}}_*)$, we define a generalized continuous/discrete autoregressive model for the policy, following Eq. equation 4.2.5. Adopting the short-hand notation $\pi^{\nu}_{\boldsymbol{\theta}}\left(a^{\nu}_j \mid s_j\right) = \pi^{\nu}_{\boldsymbol{\theta}}\left(a^{\nu}_j \mid a_1, \cdots, a_{j-1}\right)$, the policy can be written as

$$\pi_{\boldsymbol{\theta}}\left(a_1, a_2, \cdots, a_q\right) = \prod_{j=1}^{q} \pi^{\mathrm{d}}_{\boldsymbol{\theta}}\left(a^{\mathrm{d}}_j \mid s_j\right) \pi^{\mathrm{c}}_{\boldsymbol{\theta}}\left(a^{\mathrm{c}}_j \mid s_j, a^{\mathrm{d}}_j\right). \tag{4.3.2}$$

As expected, at every step $j$, the action $a^{\mathrm{c}}_j$ is sampled from a continuous distribution, whose parameters depend on the discrete action $a^{\mathrm{d}}_j$ selected at the same step $j$. This is natural, since different discrete actions may require different corresponding continuous distribution parameters $\kappa, \xi$.

Additionally, similar to CD-QAOA, we impose a further restriction that no discrete action can occur in the trajectory consecutively. We use a Sigmoid-Gaussian distribution to bound the samples for the continuous actions, and normalize the durations $\alpha_j \propto a^{\mathrm{c}}_j \sim \pi^{\mathrm{c}}_{\boldsymbol{\theta}}$ to fix the total protocol duration to $\sum_{j=1}^{q} \alpha_j = T$; using the Beta distribution instead results in a similar performance.

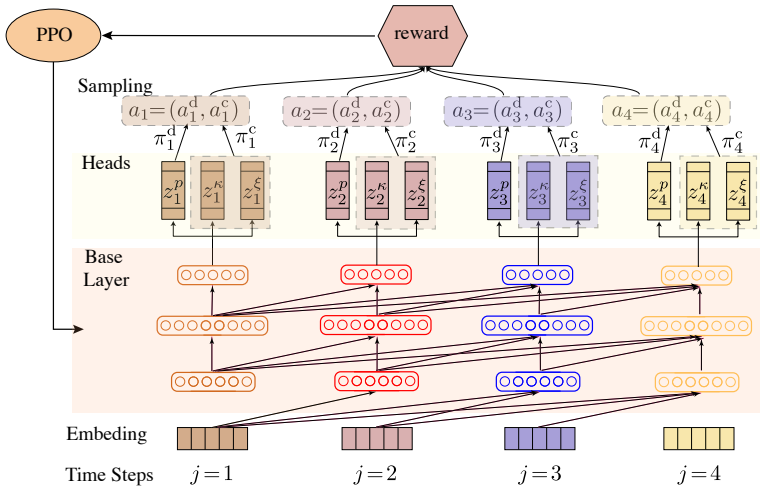



Figure 4.2: Schematic representation of RL-QAOA and the deep autoregressive network for $q = 4$ (see text). The time step $j$ also corresponds to the gate index. The policy network is composed of (i) an embedding layer to encode the continuous and discrete actions as input. (ii) The base layer implements the causal autoregressive structure (see arrows). (iii) The heads are three-fold, one for the discrete distribution parameters, and two for the continuous distribution parameters. A batch of actions are sampled to evolve the quantum state and compute the negative energy density as a reward. Proximal hybrid Policy Optimization (PPO) is used to update the policy network. The pseudocode for RL-QAOA is shown in Algorithm 4.

## Deep Autoregressive Policy Network

We implement the policy ansatz variationally, using a deep neural network called the policy network. In Fig. 4.2, we show a cartoon of the model for illustration purposes. The network consists of base layers with intermediate output $\boldsymbol{y}$, followed by three independent head layers with outputs $\boldsymbol{z}^p, \boldsymbol{z}^\kappa, \boldsymbol{z}^\xi$, respectively. The three heads learn the discrete probability distribution $\pi^{\mathrm{d}}$, and the parameters $\kappa, \xi \in \mathbb{R}^+$ which define the continuous probability distribution $\pi^{\mathrm{c}}$. Each head outputs a vector of size $|\mathcal{A}^{\mathrm{d}}|$ – so that the model can learn a set $(\kappa, \xi)$ for every distinct discrete action. Notice that each head output depends on the joint base layer parameters $(\boldsymbol{W}, \boldsymbol{b})$, but not on the parameters $(\boldsymbol{V}, \boldsymbol{c})$ of any of the other two heads; thus, the base layers are shared by all three heads. In practice, we find that a base layer, comprised of two hidden layers, can already achieve a good performance; one can in principle add more layers for enhanced expressivity.

The above description focuses on a single episode step $j$ out of a total of $q$ steps in

an episode. The autoregressive feature of the ansatz can then be built-in, by allowing the outputs of the base layers from previous steps to become inputs into the layers at subsequent episode steps [Fig. 4.2].

Let us denote the input to the autogressive network by $(x_1, x_2, \cdots, x_q)$, and the weights and bias parameters of the base layer by $W_j \in \mathbb{R}^{d_\text{h} \times (i-1)|\mathcal{A}^\text{d}|}$ and $b_j \in \mathbb{R}^{d_\text{h}}$, respectively, where $d_\text{h}$ is the hidden dimension. Then, the intermediate output $(y_1, y_2, \cdots, y_q)$ of the base layer reads as

$$y_j = g(W_j x_{<j} + b_j), \quad j = 1, 2, \cdots q, \tag{4.3.3}$$

where $x_{<j} = (x_{j-1}, \cdots, x_1)^T \in \mathbb{R}^{(j-1)|\mathcal{A}^\text{d}|}$ denotes the input of all previous steps preceding step $j$; for $j=1$, we set $W_j x_{<j} + b_j = b_j$.[2] We use ReLU nonlinearities $g(\cdot)$.

The output of the base layer $(y_1, y_2, \cdots, y_q)$ can be viewed as an input to the three-head layer. The three-head layer contains three heads with independent weights $V_j^p, V_j^\kappa, V_j^\xi \in \mathbb{R}^{|\mathcal{A}^\text{d}| \times j d_\text{h}}$ and biases $c_j^p, c_j^\kappa, c_j^\xi \in \mathbb{R}^{|\mathcal{A}^\text{d}|}$. The three-head layer output, $(z_1, z_2, \cdots, z_q)$, are the parameters for the discrete and continuous distributions: $z_j^p$ are the categorical distribution parameters; $z_j^\kappa$ and $z_j^\xi$ are the two parameters for the sigmoid-Gaussian distribution [cf. Sec. 4.8]:

$$z_j^p = \log\left(\text{SoftMax}(V_j^p y_{\leq j} + c_j^p)\right), \qquad z_j^\kappa = V_j^\kappa y_{\leq j} + c_j^\kappa, \quad z_j^\xi = \exp\Big(V_j^\xi y_{\leq j} + c_j^\xi\Big), \tag{4.3.4}$$

where $y_{\leq j} = (y_j, \cdots, y_1)^T \in \mathbb{R}^{j d_\text{h}}$.[3] To define a categorical distribution, we use a SoftMax[4] nonlinarity: $\text{SoftMax}(v)[i] = \exp(v[i]) / \sum_{k=1}^{|\mathcal{A}^\text{d}|} \exp(v[k])$, where $v = V_j^p y_{\leq j} + c_j^p \in \text{R}^{|\mathcal{A}^\text{d}|}$, and $[\cdot]$ takes the index; we learn the log-probability to achieve a resolution over a few orders of magnitude, and to stabilize the learning process.

We apply ancestral sampling to draw actions from the autoregressive policy. Starting from the heads layer at step $j = 1$, we first sample $a_1^\text{d} \sim \pi(a_1^\text{d}) = \text{Categorical}(\exp(z_1^p))$; we use the sampled discrete action $a_1^\text{d}$ to look-up the corresponding parameters $\kappa = z_1^\kappa[a_1^\text{d}]$ and $\xi = z_1^\xi[a_1^\text{d}]$[5] for the continuous action distribution. Then we sample the duration $\alpha_1 \propto a_1^\text{c} \sim \pi(a_1^\text{c}|a_1^\text{d}) = \mathcal{SN}\left(z_1^\kappa[a_1^\text{d}], (z_1^\xi[a_1^\text{d}])^2\right)$. The sampling output is

[2] In practice, implementing the autoregressive constraint $_{<j}$ can be achieved using masks (one for each set of weights).

[3] Note that here we are able to use the "=" sign because the previous layer of operation has already filtered out the "=" sign for those steps.

[4] Note that this function is not operated element-wise like the others; it is applied on the whole vector of dimension $|\mathcal{A}^\text{d}|$).

[5] Here, $[a_1^\text{d}]$ means taking the component by index.

passed as an input at the second step $j = 2$. To do this, we use as an embedding for $(a_1^{\mathrm{d}}, a_1^{\mathrm{c}})$ represented by the variable $x_1$, where $x_1[i] = a_1^{\mathrm{c}}$ if $i = a_1^{\mathrm{d}}$, and $x_1[i] = 0$ otherwise. Going on, we repeat the process: we sample successive actions $a_2^{\mathrm{d}}, a_2^{\mathrm{c}} \sim \pi(a_2^{\mathrm{d}}|x_1), \pi(a_2^{\mathrm{c}}|x_1, a_2^{\mathrm{d}})$. The sampling, or forward pass, through the network is then repeated $q$ times, until we reach the end of the episode; thus, at step $j$ we have $a_j^{\mathrm{d}}, a_j^{\mathrm{c}} \sim \pi(a_j^{\mathrm{d}}|x_{<j}), \pi(a_j^{\mathrm{c}}|x_{<j}, a_j^{\mathrm{d}})$. This gives the trajectory $\tau$ of mixed discrete-continuous actions. Note that the time complexity of the process is $\mathcal{O}(q \times |\mathcal{A}^{\mathrm{d}}|)$.

## Proximal Hybrid Policy Optimization

The set of all weights and biases, $\boldsymbol{\theta} = \{W_j, b_j, V_j^p, V_j^\kappa, V_j^\xi, c_j^p, c_j^\kappa, c_j^\xi\}_{j=1}^q$, defines the learnable parameters of the autoregressive policy network. We now discuss how to compute the policy gradients and define an update rule for $\boldsymbol{\theta}$.

Our goal is to maximize the RL objective within the trust region [279] for the continuous and discrete policy

$$\mathbb{E}_\tau \left[ \frac{\pi_{\boldsymbol{\theta}}(\tau)}{\pi_{\boldsymbol{\theta}_{\mathrm{old}}}(\tau)} A_{\boldsymbol{\theta}_{\mathrm{old}}}(\tau) \right], \quad \text{subject to} \quad \mathbb{E}_\tau \left[ \mathrm{D}_{\mathrm{KL}} \left[ \pi^\nu_{\boldsymbol{\theta}_{\mathrm{old}}}(\cdot), \pi^\nu_{\boldsymbol{\theta}}(\cdot) \right] \right] \leq \delta^\nu, \tag{4.3.5}$$

where $\mathrm{E}_\tau[\,\cdot\,]$ is a shorthand notation for $\mathrm{E}_{\tau=(a_1,\cdots,a_q)\sim\pi_{\boldsymbol{\theta}_{\mathrm{old}}}}[\,\cdot\,]$. The Kullback–Leibler (KL) divergence is defined as $\mathrm{D}_{\mathrm{KL}}(\pi^{\mathrm{c}}_{\boldsymbol{\theta}}, \pi^{\mathrm{c}}_{\boldsymbol{\theta}_t}) = \int_{x\in\mathcal{A}^{\mathrm{c}}} \pi^{\mathrm{c}}_{\boldsymbol{\theta}}(x) \log\left(\frac{\pi^{\mathrm{c}}_{\boldsymbol{\theta}}(x)}{\pi^{\mathrm{c}}_{\boldsymbol{\theta}_t}(x)}\right) \mathrm{d}x$, and similarly for $\nu = \mathrm{d}$; $\delta^\nu$ defines a constraint on the size of the discrete policy or continuous policy updates in distribution space. Here, $\boldsymbol{\theta}_{\mathrm{old}}$ denotes the parameters before the update; $A_{\boldsymbol{\theta}_{\mathrm{old}}}(\tau) = R(\tau) - b$ is the advantage function – the return (negative energy density) for a given trajectory w.r.t. the baseline $b$.

In practice, we utilize a clipped surrogate RL objective [278] with two clipping parameters $\epsilon^\nu$. The idea is to update the continuous and discrete policies using different $\epsilon^\nu$ during policy optimization. This allows for the discrete policy $\pi^{\mathrm{d}}_{\boldsymbol{\theta}}$ to change more quickly/more slowly as compared to the continuous policy $\pi^{\mathrm{c}}_{\boldsymbol{\theta}}$. Hence, the hybrid PPO RL objective reads as

$$\mathcal{J}(\boldsymbol{\theta}) = \mathbb{E}_\tau \Big[ \mathcal{G}^{\mathrm{d}}(\tau^{\mathrm{d}}; \boldsymbol{\theta}, \epsilon^{\mathrm{d}}) + \mathcal{G}^{\mathrm{c}}(\tau^{\mathrm{c}}; \boldsymbol{\theta}, \epsilon^{\mathrm{c}}) \Big] + \beta_S^{-1}(\mathcal{S}^{\mathrm{d}} + \mathrm{SC}), \tag{4.3.6}$$

with

$$\mathcal{G}^\nu(\tau^\nu; \boldsymbol{\theta}, \epsilon^\nu) = \min \left\{ \rho^\nu_{\boldsymbol{\theta}}(\tau^\nu) A^\nu_{\boldsymbol{\theta}_{\mathrm{old}}}(\tau^\nu),\ \mathrm{clip}\left(\rho^\nu_{\boldsymbol{\theta}}(\tau^\nu), 1-\epsilon^\nu, 1+\epsilon^\nu\right) A^\nu_{\boldsymbol{\theta}_{\mathrm{old}}}(\tau^\nu) \right\}, \tag{4.3.7}$$

where $\rho^{\nu}_{\boldsymbol{\theta}}(\tau^{\nu}) \coloneqq \frac{\pi^{\nu}_{\boldsymbol{\theta}}(\tau^{\nu})}{\pi^{\nu}_{\boldsymbol{\theta}_{\text{old}}}(\tau^{\nu})}$ is the importance weight ratio of two policies associated with trajectory $\tau^{\nu}$. The clip function, defined as $\text{clip}(\rho, x, y) = \max\big(\min(\rho, x), y\big)$ sets the value of $\rho_{\boldsymbol{\theta}}$ to be within the interval $[x, y]$, and constrains the likelihood ratio from Eq. equation 4.3.5 to the range $[1-\epsilon, 1+\epsilon]$. The entropy terms [right-most part of Eq. equation 4.3.6] are discussed below. Our goal is to find those parameters $\boldsymbol{\theta}$ which maximize $\mathcal{J}(\boldsymbol{\theta})$.

To understand the hybrid PPO algorithm, consider two limiting cases first. In the extreme case when $\epsilon^{\text{d}} \rightarrow 0$, i.e. the discrete policy $\pi^{\text{d}}_{\boldsymbol{\theta}}$ is kept fixed, our algorithm reduces to PG-QAOA. On the other hand, when $\epsilon^{\text{c}} \rightarrow 0$, the continuous policy is kept fixed; if this fixed policy additionally corresponds to the greedy "expert policy" defined by the Powell optimizer, the algorithm is reduced to CD-QAOA. In this sense, for finite values of $\epsilon^{\text{c}}, \epsilon^{\text{d}} > 0$, RL-QAOA can be viewed as a smooth interpolation between PG-QAOA and CD-QAOA.

In order to incentivize the agent to explore the action space during the early stages of training, we also added entropy to the RL objective, cf. Eq. equation 4.3.6. The entropy for a discrete/continuous policy is defined as $\mathcal{S}^{\text{d}}(\pi^{\text{d}}) = -\sum_{x\in\mathcal{X}} \pi^{\text{d}}(x) \log \pi^{\text{d}}(x)$ or $\text{SC}(\pi^{\text{c}}) = -\int_{x\in\mathcal{X}} \pi^{\text{c}}(x) \log\big(\pi^{\text{c}}(x)\big)\,\text{d}x$, respectively. The coefficient $\beta_S^{-1}$ in Eq. equation 4.3.6 defines an effective temperature, which we anneal with increasing the number of iterations. It is easy to see that the total entropy $\mathcal{S} = \mathcal{S}^{\text{d}} + \text{SC}$ associated with the hybrid policy consists of a discrete $\mathcal{S}^{\text{d}} = \sum_{j=1}^{q} \text{E}_{a_{<j}\sim\pi_{\boldsymbol{\theta}}} \mathcal{S}^{\text{d}}\big(\pi^{\text{d}}_{\boldsymbol{\theta}}(\,\cdot\,|a_{<j})\big)$, and a continuous $\text{SC} = \sum_{j=1}^{q} \text{E}_{a_{<j}\sim\pi_{\boldsymbol{\theta}}, a^{\text{d}}_j\sim\pi^{\text{d}}_{\boldsymbol{\theta}}} S^{\text{c}}\big(\pi^{\text{c}}_{\boldsymbol{\theta}}(\,\cdot\,|a_{<j}, a^{\text{d}}_j)\big)$ contribution. The RL agent has to maximize the total expected return while also maximizing the entropy associated with the policy.

In RL, there are two common ways to incorporate entropy in practice [191]: (i) whenever one can compute a closed-form expression for the entropy, entropy is added as a separate term to the objective which can be thought of as entropy regularization. Note that it is the autoregresssive structure that makes it possible to obtain the exact value for the entropy $\mathcal{S}^{\text{d}}$: for $\pi^{\text{d}}_{\boldsymbol{\theta}}(\,\cdot\,|a_{<j}) = \text{Categorical}(\exp\big(z^p_j\big))$, the entropy is $\mathcal{S}^{\text{d}}\big(\pi^{\text{d}}_{\boldsymbol{\theta}}(\,\cdot\,|a_{<j})\big) = -\sum_{k=1}^{|\mathcal{A}^{\text{d}}|} z^p_j[k] \cdot \exp\big(z^p_j[k]\big)$. (ii) Often times it is not possible to compute the value for the entropy, since the expression is not analytically tractable; in such cases, the maximum entropy formulation [126, 127, 125] still allows us to add to the reward a sample estimate of the entropy, known as an entropy bonus: $R^{\text{c}}(\tau) \leftarrow R^{\text{c}}(\tau) + \beta_S^{-1} \text{E}_{a^{\text{c}}\sim\pi^{\text{c}}_{\boldsymbol{\theta}}}\big[-\log \pi^{\text{c}}_{\boldsymbol{\theta}}\big]$. In this study, we add an entropy bonus to take into account the entropy of the continuous policy $\pi^{c}$.

## 4.4 Application: Quantum Ising Model in the Presence of Noise

To test the performance of RL-QAOA, we investigate the ground state preparation problem for a system of $N$ interacting qubits (i.e. spin-1/2 degrees of freedom), described by the Ising Hamiltonian introduced in Eq. equation 4.2.3. We use periodic boundary conditions and work in the zero momentum sector of positive parity, which contains the antiferromagnetic ground state. We emphasize that this model is non-integrable, i.e., it does not have an extensive number of local integrals of motion; as a consequence, no closed-form analytical description is known for its eigenstates and eigenenergies. Moreover, the lack of integrability results in chaotic quantum dynamics.

In the following, $J=1$ sets the energy unit, $h_z/J=0.4523$ and $h_x/J=0.4045$. In the thermodynamic limit, $N\to\infty$, these parameters are close to the critical line of the model, where a quantum phase transition occurs in the ground state between an antiferromagnet and a paramagnet; for the finite system sizes we can simulate, the critical behavior is smeared out over a small finite region. In Ref. [209], using QAOA it was shown that this region of parameter space appears most challenging in the noise-free system.

We initialize the system in the $z$-polarized product state $|\psi_i\rangle=|\uparrow\cdots\uparrow\rangle$, and aim to prepare the ground state of $H$. We use the negative energy density $-\mathcal{E}=-E/N$ as a reward for the RL agent, cf. Eq. equation 4.2.2, which is an intensive quantity as the number of qubits $N$ increases. In this study, we are mostly interested in exploring the behavior of the system, subject to various kinds of noise/uncertainty. Our primary focus is quantifying the effects of noise on the achievable fidelity, w.r.t. the noise-free values. We deliberately select a fixed duration of $JT=10$ far from the adiabatic regime, such as to exhibit the benefits of the CD-QAOA ansatz over QAOA [cf. Sec. 4.9].

We point out that, working at a fixed duration $T$, it is not always possible to achieve high-fidelity ground states. This is easy to see for decoupled qubits, where the magnitude of the spin precession frequency on the Bloch sphere (so-called Larmor precession frequency) is set by the fixed strength of the magnetic field $(h_x,0,h_z)$: hence, fixing the total protocol duration $T$, it may be physically impossible to reach the target state in the allotted time. This behavior leads to the notion of the quantum speed limit (QSL) – the minimum time required to prepare the ground state with unit fidelity.

## Three Noise Models

When operating present-day quantum devices, one is confronted with various sources of uncertainty. Since the exact form and details depend on the peculiarities and particularities of the underlying experimental platform, it is desirable to construct algorithms, capable of learning such details without extra human input. In this study, our RL agent learns in a simulator. To mimic the diversity of uncertain processes that can occur, we consider three types of noise.

### Classical Measurement Gaussian Noise

Noise naturally occurs due to imperfect measurements. For instance, the measurement signal is often present in a form of currents and voltages, whose values can only be determined within the resolution of the measurement apparatus. In practice, experimentalists perform a large number of measurements and average the result in the end to obtain an estimate for the value of an observable. By the central limit theorem, in the limit of large sample sizes, the statistics of the measurement data is approximated by a Gaussian distribution. To model this behavior, we use small Gaussian noise to add uncertainty in the reward signal: $\mathcal{E}_\gamma(\{\alpha_i\}_{i=1}^q, \tau^{\mathrm{d}}) = \mathcal{E}(\{\alpha_i\}_{i=1}^q, \tau^{\mathrm{d}}) + \epsilon_\gamma$, where $\epsilon_\gamma \sim \mathcal{N}(0, \gamma^2)$.

### Quantum Measurement Noise

In quantum mechanics, there is another, intrinsic, kind of noise, which arises due to the quantum nature of the controlled system. Consider the evolved state $|\psi(T)\rangle = U(\{\alpha_j\}_{j=1}^q, \tau)|\psi_i\rangle$ at the end of the protocol. The expected measurement for the energy density $\mathcal{E} = N^{-1}\langle\psi(T)|H|\psi(T)\rangle$ is obtained within a *quantum* uncertainty, $\Delta\mathcal{E} = N^{-1}\sqrt{\langle\psi(T)|H^2|\psi(T)\rangle - \langle\psi(T)|H|\psi(T)\rangle^2}$, set by the energy variance in the final state. In the limit of a large number of measurements, quantum noise can be simulated using a Gaussian distribution $\mathcal{E}_Q(\{\alpha_i\}_{i=1}^q, \tau^{\mathrm{d}}) = \mathcal{E}(\{\alpha_i\}_{i=1}^q, \tau^{\mathrm{d}}) + \epsilon_Q$, where $\epsilon_Q \sim \mathcal{N}(0, \Delta\mathcal{E}^2)$. Note that the width of the Gaussian depends on the final state $|\psi(T)\rangle$: in the early stages of training, $|\psi(T)\rangle$ is typically far away from any of the eigenstates of $H$; therefore, the energy variance $\Delta\mathcal{E}$ will be large and finite. However, towards the later training stages, when the agent learns to prepare a state close to the target ground state, the energy variance will go down. Hence, one can think of the quantum noise as a Gaussian noise with a time-dependent strength.

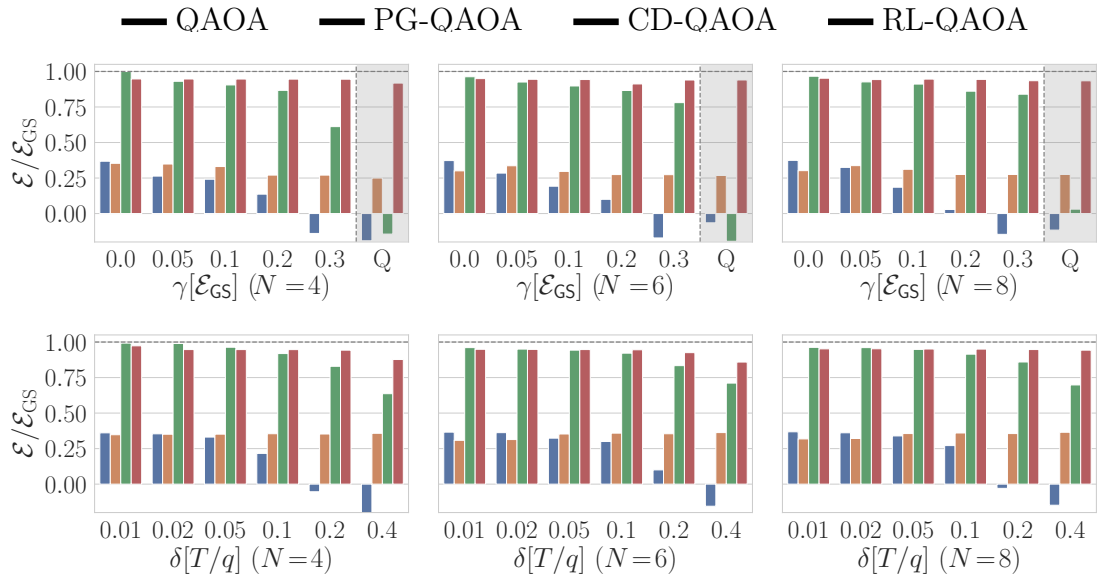



Figure 4.3: Energy minimization against different noise levels with circuit depths $p=q/2=4$ and protocol duration $JT=10$ for four different optimization methods: QAOA, PG-QAOA, CD-QAOA (red line), RL-QAOA. The initial and target states are $|\psi_i\rangle=|\uparrow\cdots\uparrow\rangle$ and $|\psi_*\rangle=|\psi_{\rm GS}(H)\rangle$ for $h_z/J=0.4523$ and $h_x/J=0.4045$. The alternating unitaries for conventional QAOA and PG-QAOA are generated by $\mathcal{A}^{\rm d}=\{H_1,H_2\}$; for CD-QAOA and RL-QAOA, we extend this set using adiabatic gauge potential terms to $\mathcal{A}^{\rm d}=\{H_1,H_2;Y,X|Y,Y|Z\}$. The system sizes are $N=4,6,8$.

### Noise arising from Gate Rotation Errors

Finally, we also consider the uncertainty in implementing the unitaries $U_i$. We focus on gate rotation errors [293], caused by imperfections in the durations $\alpha_i$: $\mathcal{E}_\delta(\{\alpha_i\}_{i=1}^q,\tau)=\mathcal{E}(\{\alpha_i+\epsilon_i\}_{i=1}^q)$, where $\epsilon_i\sim\mathcal{N}(0,\delta^2)$. This defines a simplified error model for coherent control, an important source of errors in present-day state-of-the-art quantum computing hardware [17], and which is especially pertinent to the case of quantum computers which are utilized frequently but calibrated only periodically.

## 4.5 Numerical Experiments and Results

To evaluate the performance of the trained agent, we eliminate the uncertainty associated with the probabilistic nature of the policy: we take the discrete action

which maximizes the categorical distribution $\pi^{\mathrm{d}}$, and only keep the mean of the continuous distribution $\pi^{\mathrm{c}}$, setting its width to zero. This defines a natural greedy policy to test the ability of the RL agent.

We performed a number of numerical experiments to study the effect of the noise on the performance of the four algorithms QAOA, PG-QAOA, CD-QAOA and RL-QAOA, for the three different sources of uncertainty: classical and quantum measurement noise, and gate rotation noise. We vary both the noise strength, and we look at three different system sizes for two protocol durations each. The results of these experiments can be summarized, as follows.

Figure 4.3 shows the best achievable energy at a protocol duration $JT = 10$ against different noise types and system sizes: the top row shows data for various measurement noise strengths, with the shaded area marking the special case of quantum noise; the noise strength is measured in percentages of the achievable ground state energy density: e.g., a noise strength of $\gamma = 0.3$ corresponds to an average deviation from the actual energy of about 30%. The bottom row displays results when varying the gate noise strength. Here, the noise strength is defined as a percentage of the mean gate duration $T/q$. The three columns correspond to system sizes $N = 4$ (left), $N = 6$ (middle) and $N = 8$ (right).

When $T < T_{\mathrm{QSL}}$ is chosen below the QSL, we find that QAOA and PG-QAOA fail to reach the ground state in the time allotted, as a result of having an overconstrained control space $\mathcal{A}^{\mathrm{d}} = \{H_0, H_1\}$. Nonetheless, the noise-robust character of PG-QAOA becomes pronounced at increased values of the noise strength. Since the initial quantum state is far away from the target ground state, the best ratio $E/E_{\mathrm{GS}}$ found by QAOA can even be negative. The $JT = 10$ duration exhibits the advantage of using the generalized QAOA ansatz brought in by CD-QAOA: suitably enlarging the discrete action space $\mathcal{A}^{\mathrm{d}} = \{H_1, H_2, Y, X|Y, Y|Z\}$ unlocks paths in Hilbert space which are inaccessible to QAOA. Hence, CD-QAOA and RL-QAOA find the largest rewards in the noise-free case. A large noise strength reduces visibly the ability of CD-QAOA to find the ground state, with the performance being particularly bad for quantum measurement noise (Q). However, the hybrid policy optimizer allows RL-QAOA to emerge as a noise-robust algorithm, agnostic to the source of noise applied to the system.

## 4.6 Conclusion and Outlook

In summary, we presented RL-QAOA – a versatile and noise-robust quantum control algorithm based on the QAOA variational ansatz. The algorithm inherits valuable

features from its ancestors: (i) the noise-robust property of PG-QAOA allows us to find optimal durations probabilistically. (ii) the generalized QAOA ansatz of CD-QAOA makes it possible to select the order in which a set of unitaries appears in the control sequence. While we focused on physically-motivated unitaries, we emphasize that the ansatz is completely general and applicable to a large variety of unitaries/quantum gate sets useful for both theoretical and experimental studies. We had to modify these "ancestors" accordingly: in PG-QAOA we introduced a mechanism to fix the total protocol duration, and introduced a stochastic policy based on the compactly supported Beta function; in CD-QAOA we changed the low-level optimizer to gradient-free Powell, as opposed to the gradient-based SLSQP which did not give a reasonable performance in the presence of noise. RL-QAOA extends PG-QAOA and CD-QAOA with both the use of a generalized autoregressive architecture which incorporates the parameters of the continuous policy, and the derivation of an extension of Proximal Policy Optimization applicable to hybrid continuous-discrete policies.

We tested the performance of RL-QAOA using the unitary dynamics of a quantum Ising chain subject to various sources of noise: classical and quantum measurement noise as well as uncertainty leading to errors in the application of quantum unitary gates. In particular, we demonstrated that RL-QAOA successfully outperforms its ancestors in the highly-constrained non-adiabatic regime, irrespective of the noise model selected. Thus, RL-QAOA is not only noise-robust but also agnostic to the physical source of noise. This opens up the exciting possibility of using machine learning to 'learn' the particularities of noisy experimental environments, which often depend on the chip architecture and can even change in the course of exploitation. However, the presented results are obtained using numerical simulations based on certain theoretical assumptions; it remains to test the performance of RL-QAOA on realistic noisy intermediate-scale quantum computing devices.

The RL-QAOA is a versatile method that can be extended along several directions. For instance, the current version of RL-QAOA defines a fixed sequence/protocol length. However, the algorithm is versatile enough to accommodate a variable length of the protocols after a slight modification. To do so, one can simply add a "stop" action to the discrete action set $\mathcal{A}^d$. If the agent happens to choose the stop action, then the episode comes to an end immediately and we measure the energy of the evolved quantum state.

There also exist a number of exciting alternatives for the policy network architecture to explore. Although it has to incorporate temporal causality, notice that the architecture is not limited to the autoregressive choice used in this study; e.g., it can be generalized

to a recurrent neural network (RNN), a Long Short Term Memory (LSTM) network, or a transformer with the attention mechanism [314] and all its modern variants [168, 71, 323, 305]. In the present study, we chose the autoregressive network for its sheer simplicity. Moreover, the continuous policy head can be generalized to capture distributions with more than two modes using the normalizing flow method, which would additionally boost the expressivity of the policy [303].

## 4.7 Pseudocode and Algorithm Hyperparameters

The pseudocode for RL-QAOA is outlined in Algorithm equation 4. The agent samples a batch of actions from the autoregressive network. Then, the corresponding expected energy density is computed using a classical simulator for the quantum dynamics. Below, we focus on the noise-free case; dealing with noise requires a trivial modification following Sec. 4.4. The baseline for the reward is estimated through an exponential moving average. Finally, proximal policy optimization is applied to update the agent's policy.

We also conducted coarse hyperparameter sweeps to find the optimal values for the hyperparameters of RL-QAOA, cf. Table 4.2. We use a batch size of 128 to train the policy. The policy network is optimized using Adam. The initial learning rate is set to $5 \times 10^{-4}$, which is typical when training autoregressive networks; we employ a learning rate decay schedule which decreases by 98% every 50 iterations. The Autoregressive network is implemented using uniform masks and dense layers [112]. The base layer (see Fig. 4.2) consists of two hidden layers with 100 neurons each and the heads contain $3|\mathcal{A}^{\mathrm{d}}|$ neurons in total.

The agent is trained via proximal policy gradient (PPO). We use four PPO updates to the policy network parameters per iterations. The clipping parameters are set as $\epsilon^{\mathrm{c}} = 0.1$ for the continuous policy, and $\epsilon^{\mathrm{d}} = 10^{-3}$ for the discrete policy. We include entropy bonus to increase exploration; the corresponding temperature schedule $\beta_{\mathcal{S}}^{-1}$ starts at $1 \times 10^{-1}$, and drops by 99% every 50 iterations.

A typical learning curve in the noisy setting is shown in Fig. 4.4. Three quantities are recorded to measure the performance of the agent. In the noise setting, these quantities correspond to the ideal noise-free case. We use them only for the purpose of evaluation; during the training, the RL agent only has access to the noisy rewards. These quantities are shown in terms of the energy ratio with respect to the target ground state so the possible maximum is upper bounded by one; since the energy of a state can be either positive or negative, while the GS has a negative value, negative ratios are possible. Figure 4.4 shows that the agent starts to pick up the learning

**Algorithm 4** Autoregressive network based reinforcement learning: RL-QAOA

**Input:** batch size $M$, learning rate $\eta_t$, total number of iterations $T_{\text{iter}}$, exponential moving average coefficient $m$, entropy coefficient $\beta_S^{-1}$, PPO gradient steps $K$.

1: Initialize the autoregressive network and initialize the moving average $\hat{R}=0$.
2: **for** $t = 1, .., T_{\text{iter}}$ **do**
3: Autoregrssively sample batch $B$ hybrid actions of size $M$:

$$a_1^{\{k\}}, a_2^{\{k\}}, \cdots, a_q^{\{k\}} \sim \pi_{\boldsymbol{\theta}}\left(a_1, a_2, \cdots, a_q\right), \ k = 1, 2, \cdots, M.$$

4: Measure the observables and use the negative energy density as the return and compute the moving average of the return

$$R_k = -\mathcal{E}_k = -\frac{1}{N}\langle\psi_i|U^\dagger(\{a_j^{\{k\}}\}_{j=1}^q)HU(\{a_j^{\{k\}}\}_{j=1}^q)|\psi_i\rangle,$$

$$\hat{R} = m \cdot \hat{R} + (1-m) \cdot \frac{1}{M}\sum_{k=1}^{M} R_k.$$

5: Compute the advantage estimates $A_k = R_k - \hat{R}$
6: **for** $k = 1, .., K$ **do**
7: Evaluate the samples' likelihood using the parameter from the last iterations, i.e. $\pi_{\boldsymbol{\theta}_t}(a_1^{\{k\}}, a_2^{\{k\}}, \cdots, a_q^{\{k\}})$ and compute the importance weight $\rho_k^\nu$
8: Using the advantage estimation and importance weight to compute $\mathcal{G}_k^d, \mathcal{G}_k^c, \mathcal{S}_k^{\text{d}}, \text{SC}_k$.
9: Compute the RL-QAOA objective Eq (4.3.6) and backpropagate to get the gradients.

$$\nabla_{\boldsymbol{\theta}}\mathcal{J}(\boldsymbol{\theta}) = \frac{1}{M}\sum_{\{a_j^{\{k\}}\}_{j=1}^q \in B} \nabla_{\boldsymbol{\theta}}\Big[\mathcal{G}_k^d + \mathcal{G}_k^c + \beta_S^{-1}(\mathcal{S}_k^{\text{d}} + \text{SC}_k)\Big].$$

10: Update weights $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \eta_t \nabla_{\boldsymbol{\theta}}\mathcal{J}(\boldsymbol{\theta})$.
11: **end for**
12: **end for**

Table 4.2: RL-QAOA Hyperparameters.

| Hyperparameter | Value |
|---|---|
| Optimizer | Adam [163] |
| Learning rate | $5 \times 10^{-4}$ |
| Likelihood ratio clip, $\epsilon^{\nu}$ | 0.1 ($\epsilon^{\mathrm{c}}$) |
| | 0.001 ($\epsilon^{\mathrm{d}}$) |
| PPO Epochs | 4 |
| Hidden units (masked dense layer) | $[100, 100]$ |
| Activation function | ReLU |
| baseline exponential moving average ($m$) | 0.95 |
| Learning rate annealing steps | 50 |
| Learning rate annealing factor | 0.98 |
| Learning rate annealing style | Staircase |
| Entropy bonus temperature ($\beta^{-1}_{S,\{0\}}$) | $1 \times 10^{-1}$ |
| Entropy bonus temperature decay steps | 50 |
| Entropy bonus temperature decay factor | 0.99 |
| Entropy bonus temperature decay style | Smooth |
| Minibatch size | 128 |

signal around two thousand iterations. After that, it slightly modifies the policy in order to achieve a higher reward. Here, the mean reward stands for the sample mean of energy density at every iteration; the max reward is the maximum over the sample; the history best is the best-encountered reward during the entire training process.

## 4.8 A comparison of compactly supported distributions defining continuous actions

### Sigmoid Gaussian Distribution

In order to enforce the bounds for the duration output from the distribution, we apply the sigmoid function. This kind of finite bound of the action does help a lot in practice when we then normalize the actions to have the finite sum; otherwise, we observe a large variance when we normalize the total protocol duration to $T$ (see main text). To this end, we apply the sigmoid function to the Gaussian distribution. In the following formula, we have $x = f(y)$, where $f(y) = \frac{1}{1+e^{-y}}$ is the sigmoid. We denote

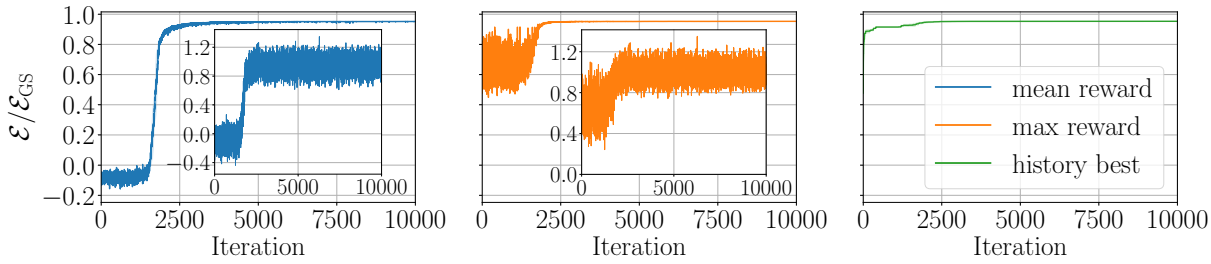



Figure 4.4: Spin-1/2 Ising model: training curves for RL-QAOA with energy minimization as a cost function. The quantities in the main figure are noiseless evaluation, while in the inset are noisy measurement. The noiseless quantities are only for the evaluation's purpose, and the agent can only access the noisy quanties (in the inset). The mean reward (blue curve) is the average energy ratio across the minibatch sampled from the autoregressive policy; the max reward (orange curve) is taking the maximum across the minibatch; the history best bookkeeps the best ever max reward during the training. The total duration is $T=10$ and the number of spin-1/2 particles is $N=8$. The discrete RL-QAOA action space is $\mathcal{A}^{\mathrm{d}} = \{H_1, H_2; Y, X|Y, Y|Z\}$, and we use $q=8$. Here, the noise is classic gaussian noise, with the noise level $\gamma=0.1$.

the original distribution as $\pi_0(y;\kappa,\xi)$ and the distribution after the transformation, as $\pi(x;\kappa,\xi)$.

$$\pi(x;\kappa,\xi) = \pi_0(y;\kappa,\xi)\left|\det\left(\frac{\mathrm{d}x}{\mathrm{d}y}\right)\right|^{-1}$$

For example, if we choose $\pi_0$ to be Gaussian distribution according to $\mathcal{N}(\kappa,\xi^2)$, then

$$\log\pi(x;\kappa,\xi) = -\log\xi - \frac{1}{2}\log(2\pi) - \frac{1}{2}\left(\frac{\mathrm{logit}(x)-\kappa}{\xi}\right)^2 - \log(x\,(1-x)) \tag{4.8.1}$$

Here, the logit function, $\mathrm{logit}(x) = \log x - \log(1-x)$, is the inverse of the sigmoid function $f(x) = 1/(1+\exp(-x))$.

Thus, the derivative with respect to the parameters (i.e. $\kappa$ and $\xi$) can be computed analytically, and reads

$$\frac{\partial\log\pi(x;\kappa,\xi)}{\partial\kappa} = \frac{\mathrm{logit}(x)-\kappa}{\xi^2}, \tag{4.8.2}$$

$$\frac{\partial\log\pi(x;\kappa,\xi)}{\partial\xi} = -\frac{1}{\xi} + \frac{1}{\xi}\left(\frac{\mathrm{logit}(x)-\kappa}{\xi}\right)^2 \tag{4.8.3}$$

We use this log probability in the policy gradient formula.

### Beta Distribution

The Beta distribution's probability density function is defined as:

$$\pi(x;\kappa,\xi) = \frac{\Gamma(\kappa+\xi)}{\Gamma(\kappa)\Gamma(\xi)} x^{\kappa-1}(1-x)^{\xi-1},$$

where the Gamma function is $\Gamma(z) = \int_0^\infty x^{z-1}e^{-t}\,dt$. Here, the $\kappa$ and $\xi$ are the parameters of the Beta distribution, which can be learned by the autoregressive policy network. The corresponding log-probability reads as

$$\log \pi(x;\kappa,\xi) = \log\Gamma(\kappa+\xi) - \log\Gamma(\kappa) - \log\Gamma(\xi) + (\kappa-1)\log(x) + (\xi-1)\log(1-x), \tag{4.8.4}$$

Thus, the derivative with respect to the parameters (i.e. $\kappa$ and $\xi$) reads

$$\frac{\partial \log \pi(x;\kappa,\xi)}{\partial \kappa} = \psi(\kappa+\xi) - \psi(\kappa) + \log(x), \tag{4.8.5}$$

$$\frac{\partial \log \pi(x;\kappa,\xi)}{\partial \xi} = \psi(\kappa+\xi) - \psi(\xi) + \log(1-x), \tag{4.8.6}$$

where the digamma function is defined as the logarithmic derivative of the gamma function:

$$\psi(x) = \frac{d}{dx}\ln\big(\Gamma(x)\big) = \frac{\Gamma'(x)}{\Gamma(x)}. \tag{4.8.7}$$

Hence, the gradient can be used to compute the policy gradient using analytical expressions.

## 4.9 Choosing the protocol duration $T$

Finally, let us explain the choice of protocol durations $JT = 10$ used in our study. Figure 4.5 shows a scan of the best energy over the protocol duration $T$ in the noise-free case for $N = 4$ qubits for three methods: QAOA, CD-QAOA and adiabatic driving. For the adiabatic driving, we consider the driven spin-1/2 Ising model:

$$H(\lambda) = \lambda(t)H + (1-\lambda(t))\tilde{H}, \tag{4.9.1}$$

where $\lambda(t) = \sin^2\left(\frac{\pi t}{2T}\right)$ $t \in [0,T]$, is an smooth protocol satisfying the boundary conditions: $\lambda(0)=0$, $\lambda(T)=1$, $\dot{\lambda}(0)=0=\dot{\lambda}(T)$. The initial state is the ground state at $t=0$, i.e. $|\psi_i\rangle = |\uparrow\cdots\uparrow\rangle$, while the target state is the ground state of the Ising model

at $t = T$ for $h_z/J = 0.4523$ and $h_x/J = 0.4045$. Here, $H$ is the target Hamiltonian defined in Eq. equation 4.2.3, and $\tilde{H} = -\sum_{i=1}^{N} S_i^z$.

The value $JT = 10$ is selected to achieve a compromise: on one hand, it is large enough for CD-QAOA to reach close enough to the ground state; on the other hand, it is small enough for a discrepancy between the performance of CD-QAOA and QAOA to become clearly visible. Hence, $JT = 10$ exemplifies nicely the benefits of using the generalized QAOA ansatz, as compared to QAOA. Last, we emphasize that both $JT = 10$ are far away from the adiabatic regime, as shown by the adiabatic curve.

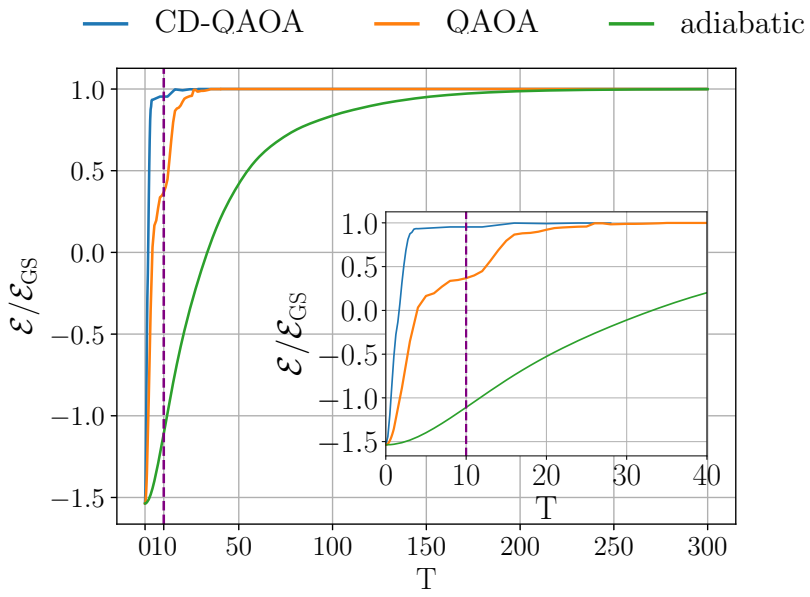



Figure 4.5: Spin-1/2 model: energy minimization at different protocol duration $T$ for three different methods in the nose-free setup: CD-QAOA (blue line), QAOA (red line), adiabatic evolution (green line). The physics model and the setting are the same as in Sec. 4.4. For the adiabatic driving simulation, we used the protocol function $\lambda(t) = \sin^2\left(\frac{\pi t}{2T}\right)$, $t \in [0, T]$. The quantum dynamics was solved for numerically, using a step size of $\Delta t = 1 \times 10^{-3}$. The system size is $N = 4$. The vertical purple dashed line corresponds to $JT = 10$.

# Chapter 5

# Monte Carlo Tree Search based variational quantum algorithms

Variational quantum algorithms stand at the forefront of simulations on near-term and future fault-tolerant quantum devices. While most variational quantum algorithms involve only continuous optimization variables, the representational power of the variational ansatz can sometimes be significantly enhanced by adding certain discrete optimization variables, as is exemplified by the generalized quantum approximate optimization algorithm (QAOA). However, the hybrid discrete-continuous optimization problem in the generalized QAOA poses a challenge to the optimization. We propose a new algorithm called MCTS-QAOA, which combines a Monte Carlo tree search method with an improved natural policy gradient solver to optimize the discrete and continuous variables in the quantum circuit, respectively. We find that MCTS-QAOAhas excellent noise-resilience properties and outperforms prior algorithms in challenging instances of the generalized QAOA.

## 5.1 Introduction

Quantum computing provides a fundamentally different way for solving a variety of important problems in scientific computing, such as finding the ground state energy in computational chemistry, and the MaxCut problem in combinatorial optimization. Variational quantum circuits are perhaps the most important quantum algorithms on near term quantum devices [251], mainly due to the tunability and the relatively short circuit depth [62], as exemplified by the variational quantum eigensolver (VQE) [244, 213] and the quantum approximate optimization algorithm (QAOA) [102]. A common thread in these algorithms is to variationally optimize a parameterized quantum

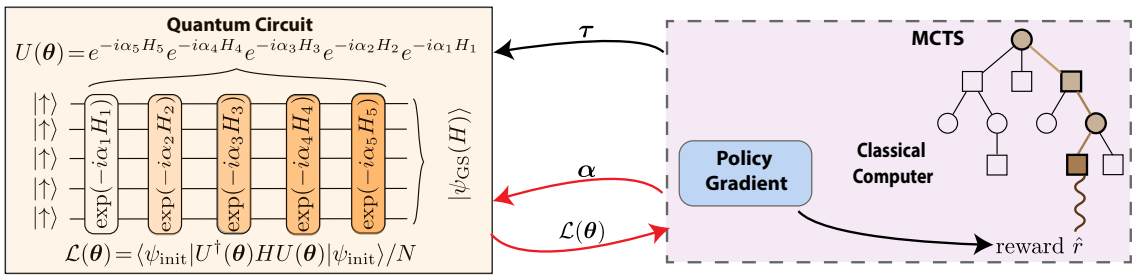



Figure 5.1: **The schematics of MCTS-QAOA**: MCTS provides promising paths for the discrete optimization search; the inner loop (highlighted in red) Policy Gradient (PG) solver evaluates the discrete sequence in a noise-robust way; the reward obtained is then propagated back through the search tree and used to improve the tree policy.

circuit using classical methods to obtain an approximate ground state. For instance, in combinatorial optimization, QAOA encodes the classical objective function into a quantum Hamiltonian, and constructs a quantum circuit with a set of two alternating quantum gates. The continuous adjustable parameters are the duration or phases of the gates.

For quantum many-body problems, the expressivity of the QAOA ansatz may be limited: the exponentially large (in the number of qubits) Hilbert space may not be efficiently navigated by the dynamics generated by the alternating gate sequence. This can lead to circuit depths that grow with the system size [141], or render the target ground state outside the scope of accessible states altogether, thus fundamentally precluding its preparation. To address these problems, various versions of a generalized QAOA ansatz have been presented in recent works [365, 347, 64], where additional control Hamiltonians are used to generate the variational circuits. In general, these Hamiltonians are tailored to the many-body system whose ground state we seek to prepare, and the extended Hamiltonian pool is often constructed using ideas from variational counter-diabatic (CD) driving [280]. When the optimization of the parameterized circuit is performed successfully, the generalized ansatz produces a closer approximation to the ground state than the original alternating QAOA ansatz. The generalized QAOA may also significantly reduce the total protocol duration $T$ and therefore the depth of the quantum circuit while giving a high fidelity with respect to the ground state [347].

However, the ansatz of the generalized QAOA also results in a more challenging optimization problem. The original QAOA only involves optimization of continuous parameters. The generalized QAOA ansatz, in contrast, leads to a hybrid optimization

problem that involves both the discrete variables (the choice of quantum gates) and the continuous variables (the duration of each gate). To solve this hybrid optimization problem, we propose a novel algorithm that combines the Monte Carlo Tree Search (MCTS) algorithm [74, 43, 3, 284, 285] – a powerful method in exploring the discrete sequence, with an improved noise-robust natural policy gradient solver for the continuous variables of a fixed gate sequence.

**Contributions:**

- We propose the MCTS-QAOAalgorithm which combines the MCTS algorithm and a noise-robust policy gradient solver. We show that it is not only efficient in exploring the quantum gate sequences but also robust in the presence of different types of noise.
- The proposed MCTS-QAOA algorithm produces accurate results for problems that appear difficult or infeasible for previous algorithms based on the generalized QAOA ansatz, such as RL-QAOA [350]. In particular, MCTS-QAOA shows superior performance in the large protocol duration regime, where the hybrid optimization becomes challenging.
- In order for the MCTS-QAOA algorithm to produce reliable optimal results, it is crucial that the inner loop solver finds the optimal continuous variables with high accuracy. Compared to the original PG-QAOA solver introduced in [346], we improve the inner loop solver with entropy regularization and the natural gradient method, and implement it in Jax [40], which offers more accurate, stable, and efficiently computed solutions during the continuous optimization.
- For the physics models considered in this paper, we observe that there can be many "good" gate sequences. This means that for a large portion of gate sequences, the energy ratio obtained is not far away from the optimal energy ratio obtainable with the generalized QAOA ansatz, given that the continuous variables are solved with high quality. This phenomenon has not been recorded in the literature to the best of the authors' knowledge.

**Related works:**

**Quantum control and variational quantum eigensolver**: Traditional optimal quantum control methods, often used in prior works, are GRAPE [161] and CRAB [53]. More recently, success has been seen by the combination of traditional methods with machine learning [273, 318, 271, 105, 228, 4, 287, 336, 339, 10, 76], and especially reinforcement learning [237, 107, 19, 249, 325, 346, 292, 65, 48, 50, 288, 37, 78, 219]).

Among them, Variational quantum eigensolver or VQE [63, 308] provides a general framework applicable on noisy intermediate-scale quantum (NISQ) devices [251] to variationally tune the circuit parameters and improve the approximation. In the fault tolerant setting, there are also possibilities of error mitigation via the variational quantum optimization [293, 16].

QAOA [102] can be viewed as a specific variational quantum algorithm, and can be extended to the generalized QAOA ansatz [365, 347, 64]. Prior works optimize the generalized QAOA greedily and progressively for each circuit layer or end-to-end as a large autoregressive network. The present work differs from these methods; we take advantage of the MCTS structure and formulate the problem as a two-level optimization.

**MCTS and RL**: Monte Carlo tree search (MCTS) has been one major workhorse behind the recent breakthrough of reinforcement learning algorithm, especially AlphaGo algorithms and variants [284, 285, 283, 276, 352]. MCTS [43, 124] makes use of a discrete hierarchical structure to figure out a better exploration in high dimensional search problems. While it is typically applied to discrete search, it has also been used in the continuous setting [322], where the partition space of the whole space is viewed as branching of the tree. In the context of quantum computing, applications of MCTS have been recently emerged such as the Quantum Circuit Transformation [363], the quantum annealing schedules [67], and the quantum dynamics optimization [78].

Further related works in hybrid optimization, counter-diabatic driving methods, and architecture search can be found in Section 5.7.

## 5.2 Generalized QAOA ansatz

The generalized QAOA ansatz [347] constructs a variational quantum circuit via the composition of a sequence of parameterized unitary operators:

$$U(\boldsymbol{\theta}) = \prod_{j=1}^{q} U(\tau_j, \alpha_j) = \prod_{j=1}^{q} \exp\big(-i\alpha_j H_{\tau_j}\big). \tag{5.2.1}$$

Here the circuit parameters $\boldsymbol{\theta} = (\boldsymbol{\alpha}, \boldsymbol{\tau})$ contain two components: i) the *discrete* variables $\boldsymbol{\tau} = (\tau_1, \tau_2, \ldots, \tau_q)$ define a sequence of Hamiltonians with length $q$, while ii) the *continuous* variables $\boldsymbol{\alpha} = \{\alpha_j\}_{j=1}^{q}$ represent the duration that each corresponding gate is applied for. It is further assumed that each Hamiltonian $H_{\tau_j}$ is selected from a fixed Hamiltonian pool $\mathcal{A} = \{H_1, H_2, \cdots, H_{|\mathcal{A}|}\}$, and consecutive gates are not repeated, i.e., $\tau_j \neq \tau_{j+1},\ 1 \leq j \leq q-1$. The total number of possible sequences is

thus $|\mathcal{A}|(|\mathcal{A}|-1)^{q-1}$, which grows exponentially with $q$, rendering exhaustive search intractable.

After applying the circuit to an initial quantum state $|\psi_{\text{init}}\rangle$, one obtains the final quantum state $|\psi\rangle = U(\boldsymbol{\theta})\,|\psi_{\text{init}}\rangle$. To prepare a high quality approximation of the ground state $|\psi_{\text{GS}}\rangle$ of the target Hamiltonian $H$, the continuous and discrete variables are solved for by minimizing the following objective function:

$$\mathcal{L}(\boldsymbol{\theta}) = E(\boldsymbol{\theta})/N = \langle\psi_{\text{init}}|U^{\dagger}(\boldsymbol{\theta})HU(\boldsymbol{\theta})|\psi_{\text{init}}\rangle/N. \tag{5.2.2}$$

Note that the energy $E$ in the objective function is divided by the number of particles $N$ in the physical model, e.g. the number of qubits. This scaled objective function has a well-behaved limit when increasing the number of qubits, as required for larger-scale computations. Here, the energy function $E(\boldsymbol{\theta})$ is always lower-bounded by the ground state energy $E_{\text{GS}} = \langle\psi_{\text{GS}}|H|\psi_{\text{GS}}\rangle$. It is also worth noticing that the quantum states $|\psi\rangle$ are unknown to the optimization algorithm (they cannot be measured), which increases the difficulty of the optimization algorithm.

## 5.3 Reinforcement learning setup

After defining the optimization problem posed by the generalized QAOA, let us briefly cast it within the RL framework.

### Quantum constraints on the RL environment

Beyond classical physics, quantum mechanics imposes counterintuitive constraints on the state and reward spaces, which need to be embedded in a realistic RL environment.

First, the quantum state (or wavefunction) is not a physical observable by itself, and inference of the information of the full quantum state from experiments (called quantum state tomography) can require exponential resources. This fact is intimately related to the expected superior performance of quantum computers against their classical counterparts on certain tasks. To embed this quantum behavior into our environment simulator, we define the RL state as the sequence of actions applied [48] rather than the quantum state. Starting from a fixed initial state, the quantum state is uniquely determined (though still unmeasurable) by the Hamiltonian sequence applied.

Second, (strong) quantum measurements lead to a collapse of the quantum wavefunction. This means that, once a measurement has been performed, the state itself is irreversibly lost. Therefore, a second constraint for our quantum RL environment

is the sparsity of rewards. Indeed, only after the RL episode comes to an end, can we measure the energy and obtain the reward. In Sec. 5.4, we exploit this fact to introduce MCTS into the algorithm which does not evaluate the protocol $\boldsymbol{\tau}$ during the construction of it. As a result, the evaluation is delegated to the noise-robust PG-QAOA solver.

### The reinforcement learning environment

In the language of reinforcement learning (RL), the choice of quantum gates corresponds to the action of the learner, and the quantum circuit is completed after $q$ actions, which marks the end of the RL session/episode. The reward signal is provided by the inner loop solver which aims to compute the lowest possible energy that can be reached by the fixed chosen gate sequence. To be more specific, the action space $\mathcal{A} = \{H_j : 1 \leq j \leq |\mathcal{A}|\}$ is a set of Hamiltonians; the state space $\mathcal{S} = \{(\tau_1, \tau_2, \ldots, \tau_t) : \tau_j \in \mathcal{A}, 0 \leq j \leq t, 1 \leq t \leq q\}$ is the set of sequences of Hamiltonians with length no larger than $q$. In particular, a session always starts with the empty sequence $s_0$, and ends with a state given by a Hamiltonian sequence of length $q$. When $s_t = (\tau_1, \tau_2, \ldots, \tau_t)$ is not a terminal state, i.e., $t < q$, the next state $s_{t+1}$ is obtained by appending the $(t+1)$-th action $\tau_{t+1}$ at the end of $s_t$, i.e., $s_{t+1} = (\tau_1, \tau_2, \ldots, \tau_t, \tau_{t+1})$.

The reward $r(s)$ only depends on the state $s$, and it is set as 0 whenever $s$ is not a terminal state. As explained in the previous section, this implements the physical constraint reflecting the inability to perform a strong quantum measurement without destroying the quantum state. When $s$ is a terminal state $\boldsymbol{\tau} = (\tau_1, \tau_2, \ldots, \tau_q)$, we define

$$r(s) = r(\boldsymbol{\tau}) = -\min_{\alpha} E(\{\alpha_j\}_{j=1}^q, \boldsymbol{\tau})/N, \tag{5.3.1}$$

where $\{\alpha_j\}_{j=1}^q$ are the duration obtained by the inner loop continuous optimizer, and the energy $E$ is defined in equation 5.2.2.

## 5.4 Monte Carlo tree search with improved policy gradient solver

In this section, we introduce MCTS-QAOA, an algorithm that solves the hybrid optimization problem defined by the generalized QAOA ansatz, using a combination of MCTS and an improved policy gradient solver. In the combined algorithm, MCTS serves as the solver for the outer optimization problem: it is used to search for high quality gate sequences $\boldsymbol{\tau}$. At the same time, we design an improved policy gradient

**Algorithm 5** MCTS-QAOA

**Input:** UCB bound coefficient $c$, number of outer loop iterations $T_{\text{iter}}$, number of random initialization $T_{\text{init}}$.

Initialize the Monte Carlo tree.
**for** $t = 1, .., T_{\text{iter}}$ **do**
  Pick a node according to the tree policy $\pi_{\text{tree}}$, cf. Eq. equation 5.4.1, using the UCB bound with parameter $c$.
  **if** the tree node is not the terminal state **then**
    Randomly roll out from the current tree node to obtain a terminal state $\boldsymbol{\tau}_t$.
  **end if**
  **for** $i = 1, .., T_{\text{init}}$ **do**
    Run natural policy gradient method (see Algorithm 6) to obtain the estimated reward $r_t^{[i]}$.
  **end for**
  Choose the best gate sequence durations according to the maximum reward $\hat{r}_t = \max_i r_t^{[i]}$ across different random intialization of policy gradient.
  Back-propagate the reward $\hat{r}_t$ from the node up to the root and update the statistics $(Q, N)$ on each node.
**end for**

solver to produce the optimal gates duration $\boldsymbol{\alpha}$ for the discrete sequence provided by MCTS. Finally, the outcome of the evaluation is propagated back through the nodes of the MC tree to improve the tree policy before the next iteration.

## Discrete optimization: Monte Carlo tree search

MCTS-QAOA strikes an efficient balance between exploration and exploitation of the RL states, by leveraging the statistics recorded in a search tree. Each node of this tree corresponds to a state $s$; the child nodes denote all possible states $s'$ following the state $s$. For the problem considered in this paper, trajectories are loop-free, since each child state $s'$ has one more action attached than its parent state $s$. Thus, we refer to a given node by its corresponding state. In particular, the root node corresponds to the empty state $s_0$, which has $|\mathcal{A}|$ children, one for each action; any other non-terminal state $s$ has $|\mathcal{A}|-1$ children, reflecting the constraint that no action can follow itself, and a terminal state has none. During the search process, each node keeps track of the statistics of two quantities: i) $N(s, a)$ counts the selection of action $a$ at state $s$; ii) $Q(s, a)$ is the expected reward after taking action $a$ at state $s$. Intuitively, the

average $Q(s,a)/N(s,a)$ is an estimate of how promising a child node is. Finally, a node $s$ is called fully expanded, if all its children are visited in the search, i.e., if $N(s,a)\geq 1$ for all $a\in\mathcal{A}$; otherwise, $s$ is called an expandable node, and is the focus of exploration.

In each MCTS iteration, the tree and the node statistics are updated as follows:

1. *Forming a search path.* Starting from the root node, if the current node is fully expanded, then one of its children is chosen according to the following Upper Confidence Bound (UCB) [18]:

$$\pi_{\text{tree}}(s)=\arg\max_{a\in\mathcal{A}}\left(\frac{Q(s,a)}{N(s,a)}+c\sqrt{\frac{2\log N(s)}{N(s,a)}}\right), \tag{5.4.1}$$

   until reaching a terminal state or an expandable node; here $\pi_{\text{tree}}(s)$ denotes the tree policy. Then an unvisited child of the current node is chosen at random, unless the current node is a terminal state. After that, a simulation is rolled out with a uniform policy until reaching a terminal state.

2. *Evaluation and backup.* The reward $\hat{r}$[1] of the terminal state is evaluated by the inner loop solver and the tree statistics are updated using $Q(s,a)\leftarrow Q(s,a)+\hat{r}, N(s,a)\leftarrow N(s,a)+1$ for each visited edge $(s,a)$.

For the generalized QAOA ansatz, the real challenge lies in the evaluation step. On the one hand, the overall minimization of the energy depends on the potential of the trajectory selected by the MCTS, whose role is to find the optimal trajectory sequence. On the other hand, if the accuracy of the evaluation is low, then the searching process can be stuck at a severely suboptimal solution. Similarly, if the evaluation is not efficient enough, then the benefit obtained by using quantum computation strategy will also be lost. And last but not least, if the evaluation results are not robust to noise, then the algorithm can hardly be carried out on quantum devices. Hence, the inner loop solver used to implement the evaluation must be able to efficiently offer high accuracy results while being robust to different kinds of noise. The above considerations refer to the generic case; in practice, the optimization dynamics of the algorithm is set by the properties of the optimization landscape.

[1] In order to distinguish the estimated reward from the true reward $r(s)$ in the presence of noise, we denote the estimated reward as $\hat{r}$.

## Continuous optimization: natural policy gradient solver

For each terminal state $\boldsymbol{\tau} = (\tau_1, \tau_2, \ldots, \tau_q)$ reached in the MCTS process, an inner loop solver is invoked to produce the optimal duration $\boldsymbol{\alpha} = \{\alpha_j\}_{j=1}^q$ and the reward $-E(\{\alpha_j\}_{j=1}^q, \boldsymbol{\tau})/N$ which are then back-propagated through the tree to update the tree statistics. In order to ensure that the duration obtained has a practical magnitude and to allow for a fair comparison between algorithms, we further assume that the total duration of all gates is fixed as $T$, which can be seen as a protocol for the circuit depth.

The continuous optimization problem for the inner-loop solver in the reward-evaluation step is thus

$$\min_{\{\alpha_j\}_{j=1}^q} \left\{ E(\{\alpha_j\}_{j=1}^q, \boldsymbol{\tau}) : \sum_{j=1}^q \alpha_j = T;\ 0 \leq \alpha_j \leq T \right\}. \tag{5.4.2}$$

In order to avoid using explicit derivatives of the energy $E$, we instead optimize the expectation of the energy $E$ over a parameterized probability distribution of $\boldsymbol{\alpha}$; this is also crucial to to make the algorithm resilient to noise. More specifically, we set $\alpha_j = \frac{T\tilde{\alpha}_j}{\sum_k \tilde{\alpha}_k}$ to ensure the constraints on $\alpha_j$, where $\tilde{\alpha}_j$ is a random variable drawn from the sigmoid Gaussian distribution $\mathcal{SN}(\mu_j, \sigma_j)$[2]. It can be parameterized as $\tilde{\alpha}_j = \mathfrak{g}(\delta_j)$, where $\delta_j \sim \mathcal{N}(\mu_j, \sigma_j)$ is a Gaussian random variable and $\mathfrak{g}(x) = \frac{1}{1+\exp(-x)}$ is the sigmoid function. Adding a Shannon entropy regularizer to the total expected reward we obtain the regularized objective function:

$$\mathcal{J}(\{\mu_j, \sigma_j\}_{j=1}^q) = \mathbb{E}_{\delta_j \sim \mathcal{N}(\mu_j, \sigma_j)} \left[ R(\boldsymbol{\delta}) \right] + \beta_S^{-1} \sum_{j=1}^q \log \sigma_j, \tag{5.4.3}$$

which is maximized over the parameters $\{\mu_j, \sigma_j\}_{j=1}^q$. Here

$$R(\boldsymbol{\delta}) = -E\left( \left\{ \frac{T\mathfrak{g}(\delta_j)}{\sum_k \mathfrak{g}(\delta_k)} \right\}_{j=1}^q, \boldsymbol{\tau} \right) / N,$$

and $\beta_S^{-1}$ denotes the temperature, which controls the trade-off between exploration and exploitation: higher temperature $\beta_S^{-1}$ leads to a larger weight on the entropy term, and thus encourages exploration, while smaller $\beta_S^{-1}$ reduces exploration. The entropy term $\sum_{j=1}^q \log \sigma_j$ can be derived from the definition of Shannon entropy, cf. Section 5.10.

[2] $\mathcal{SN}(\mu, \sigma)$ denotes the sigmoid Gaussian distribution with parameters $\mu$ and $\sigma$, i.e., the distribution of the Gaussian random variable $\mathcal{N}(\mu, \sigma)$ under the sigmoid transformation. It is also called the logit-normal distribution

The inner loop solver is then constructed with a natural policy gradient (NPG) method applied to the regularized objective function $\mathcal{J}$ using the natural gradient direction $F^{-1}\nabla\mathcal{J}$, where $F$ is the Fisher information matrix for the joint distribution of $\{\delta_j\}_{j=1}^q$ and $\nabla\mathcal{J}$ is the gradient of $\mathcal{J}$ with respect to the parameters. This procedure is different from the solver established in PG-QAOA [346], where the standard gradient is used to update the parameters and no regularization is used. Using independent standard normal variables $\xi_j$, the natural gradient direction can be approximated by unbiased estimators:

$$F_j^{-1}\begin{bmatrix}\frac{\partial\mathcal{J}}{\partial\mu_j}\\ \frac{\partial\mathcal{J}}{\partial\log\sigma_j}\end{bmatrix} \approx \begin{bmatrix}\sigma_j R(\boldsymbol{\delta})\xi_j\\ \frac{1}{2}R(\boldsymbol{\delta})(\xi_j^2-1)+\frac{1}{2}\beta_S^{-1}\end{bmatrix}, \tag{5.4.4}$$

where $\delta_j = \mu_j + \sigma_j\xi_j$ and $F_j$ is the $j$-th 2-by-2 diagonal block of the Fisher information matrix, since $F$ is a block diagonal matrix, cf. Section 5.10. In practice, we update $\log\sigma$ instead of $\sigma$ to ensure the positivity of $\sigma$, and we use the average of the unbiased estimators in equation 5.4.4 within a batch of size $M$ to give the approximation of the natural gradient direction.

The first term in the objective function $\mathcal{J}$ can also be viewed as a smoothed reward function obtained with Gaussian perturbation. The parameter $\{\sigma_j\}_{j=1}^q$ determines the distance between $\mathcal{J}(\mu_j,\sigma_j)$ and $E(\mu_j)$ [231]. If $\sigma$ is too large, then $\mathcal{J}$ is far from $E$, and yields suboptimal solutions of $\mu_j$ since too much details are lost after the Gaussian smoothing. To avoid this, we propose to use a tempering technique (see for example [[]Sec. 5]klink2020self, abdolmaleki2018maximum, haarnoja2018soft). More specifically, after a certain number of NPG iterations, we reduce the temperature $\beta_S^{-1}$, and in the final stage of entropy adjustment (cf. line 10-12 in Algorithm 6), we discard the entropy term. In this way, the policy is less susceptible to highly suboptimal local maxima in the beginning of the inner loop optimization thanks to the entropy regularization. At the end of the optimization, the variance $\sigma_j$ decreases, since the temperature is reduced and the algorithm is able to achieve a higher precision as the smoothed problem becomes a better approximation to the original one. As a result, many policy gradient updates can be saved compared to the original policy gradient method in [346], and the quality of solutions is improved.

When the optimization by the inner loop solver is completed, the parameters $\{\mu_j\}_{j=1}^q$ are used to evaluate the reward to be back-propagated through the MC tree. More specifically, the gate sequence $\tau$ with duration $\left\{\frac{T\mathfrak{g}(\mu_j)}{\sum_i\mathfrak{g}(\mu_i)}\right\}_{j=1}^q$ is applied and a reward is obtained. In order to deal with noisy rewards, the evaluation is repeated $m$ times, and the average reward is sent to the discrete solver. The details of the inner loop algorithm is summarized in Algorithm 6.

**Algorithm 6** Improved policy gradient solver

**Input:** Action sequence $\boldsymbol{\tau}$, number of restarts $R$, batch size $M$, learning rates $\eta_t$, total number of iterations $K$, the number of evaluation repeats $m$, the total gate duration $T$, the initial temperature $\beta_S^{-1}$, the rate of temperature decrease $0 < \gamma_T < 1$.

1: Randomly initialize the mean $\{\mu_j\}_{j=1}^q$ and variance $\{\sigma_j\}_{j=1}^q$.
2: **for** $t = 1, .., R \times K$ **do**
3: Sample a batch of variables $\{\tilde{\alpha}_j^l\}_{j=1}^q, l = 1, 2, \cdots, M$ of size $M$ from sigmoid Gaussian distributions $\mathcal{SN}(\mu_j, \sigma_j)$.
4: Normalize the generalized QAOA parameter $\alpha_j = T\tilde{\alpha}_j / \sum_i \tilde{\alpha}_i$.
5: Compute the approximate NPG direction using Eq. equation 5.4.4.
6: Update the parameters with the gradient and learning rate $\eta_t$.
7: **if** $t$ mod $K = 0$ and $t < (R-1)K$ **then** $\beta_S^{-1} \leftarrow \gamma_T \beta_S^{-1}$.
8: **if** $t = (R-1)K$ **then** $\beta_S^{-1} \leftarrow 0$.
9: **end for**
10: Apply the circuit $m$ times with gate sequence $\tau$ and durations $\left\{\frac{T\mathfrak{g}(\mu_j)}{\sum_i \mathfrak{g}(\mu_i)}\right\}_{j=1}^q$, collect the rewards $\{r_k\}_{k=1}^m$, and estimate the reward $\hat{r}$ by $\hat{r} = \frac{1}{m}\sum_{k=1}^m r_k$.

**Output:** The mean and variance parameters $\{\mu_j\}_{j=1}^q$ and $\{\sigma_j\}_{j=1}^q$; the estimated reward $\hat{r}$.

### Relation to previous algorithms used to optimize the generalized QAOA ansatz

We finish this section by a comparison of MCTS-QAOA with previous methods solving the QAOA problem. As shown in Table 5.1, the CD-QAOA method adopts Scipy solver for the continuous optimization, which cannot be applied to problems with noise, and the RL-QAOA method can produce suboptimal solutions in certain regimes, which we verify with numerical experiments in the next section. Moreover, note that, due to the large neural network used in RL-QAOA, it is infeasible to apply the natural gradient methods as in Section 5.4.

## 5.5 Numerical experiments

To benchmark the performance of MCTS-QAOA, we consider three physics models: the 1-dimensional Ising model, the 2-dimensional Ising model on a square lattice, and the Lipkin-Meshkov-Glick (LMG) model. The description of the models and the

| Method | CD-QAOA | RL-QAOA | MCTS-QAOA |
|---|---|---|---|
| optimization (discrete) | AutoReg+PG | AutoReg+PG | MCTS |
| optimization (continuous) | SciPy | | PG |
| performance without noise | ✓ | ✓ | ✓ |
| performance with noise | ✗ | ✓ | ✓ |

Table 5.1: Comparison among the three algorithms for the generalized QAOA ansatz: CD-QAOA, RL-QAOA and MCTS-QAOA. In this table, AutoReg+PG stands for the policy gradient algorithm with the autoregressive neural network as a policy [350]; ✓means the algorithm can fail in certain challenging regimes (e.g., large total duration $T$).

additional Hamiltonians inspired from the counter-diabatic theory can be found in Section 5.8. In addition, in order to test the noise-resilience of MCTS-QAOA, we consider three types of noise models: classical measurement Gaussian noise, quantum measurement noise, and gate rotation error, cf. Section 6.11.

We compare the performance of MCTS-QAOA with that of RL-QAOA, and provide an analysis on why RL-QAOA might fail in certain regimes. Further analysis of the energy landscape of the discrete optimization reveals a surprising phenomenon: for generalized QAOA with optimal choices of the continuous degrees of freedom, there can be a large number of discrete protocols producing relatively accurate energies.

## Comparison with RL-QAOA

For the methods solving the generalized QAOA problem summarized in Table 5.1, the CD-QAOA algorithm cannot be applied to problems with noise since the continuous solver is not noise-resilient, while the RL-QAOA algorithm has been shown to be effective with relatively short total duration $JT$ (using unnormalized Hamiltonians [350]). Therefore, we use RL-QAOA as a baseline when evaluating the performance of MCTS-QAOA, and we focus on the more challenging regime of large $JT$ with normalized Hamiltonians[3].

[3]The Hamiltonians used in this work are normalized by their operator norm $\|H\|$, i.e., we use $H/\|H\|$ instead of the original Hamiltonian $H$. The reason for introducing the normalized Hamiltonian is that the dependence of the cost of performing a Hamiltonian evolution $e^{-iH\alpha}$ on a quantum device – $\Omega(\|H\|\alpha)$ – scales with the norm [28, 201]. Interested readers can refer to Section 5.8 for more details.

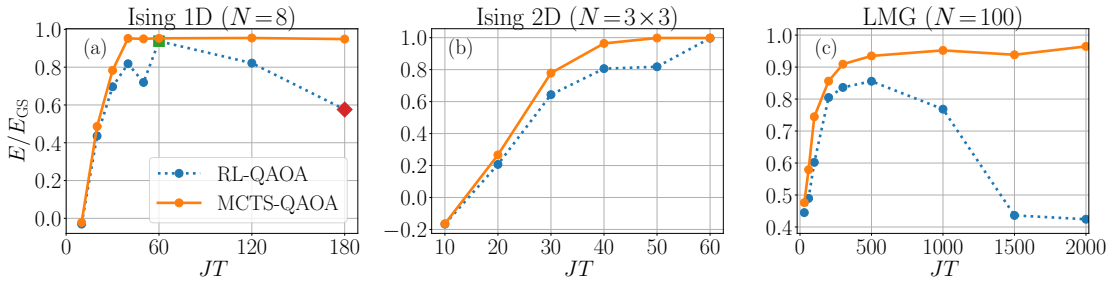



Figure 5.2: (**Quantum noise experiment**) comparison between MCTS-QAOA and RL-QAOA with quantum measurement noise (Section 6.11). (a): 1D spin-1/2 Ising chain ($N=8$); (b): 2D spin-1/2 Ising chain ($N=3\times3$); (c): LMG model ($N=100$) at $h/J=0.9$. The blue dotted line and the orange solid line display the energy ratio $E/E_{\rm GS}$ obtained by RL-QAOA and MCTS-QAOA. The green square shape and the red diamond shape in the left panel approximately corresponds to $JT=10$ (an example in the small $T$ regime) and $JT=28$ (an example in the large $T$ regime) with unnormalized Hamiltonians, respectively. The horizontal axis represents the total duration $JT$. MCTS-QAOA outperforms RL-QAOA in all tests.

We first compare the performance of MCTS-QAOA against that of RL-QAOA for the physical systems discussed in Section 5.8 in the presence of quantum noise. Detailed numerical results for the noiseless experiments and other noise models can be found in Section 6.11. In order to compare the performance of different optimizers, noisy rewards are offered to the optimizers during the training process, and the exact rewards are only used in evaluating the protocols found by the optimizers. For MCTS-QAOA, the protocol evaluated is given by a greedy search, i.e., a searching process with the exploration coefficient $c=0$ in Eq. equation 5.4.1.

Figure 5.2 shows the energy ratio evaluated for the protocols obtained by the optimizers across different lengths of total duration $JT$. For all three physics models, we find that the performance of MCTS-QAOA is at least as good as that of RL-QAOA for all protocol durations. In particular, for the 1D Ising model, MCTS-QAOA gives protocols that find close approximations to the true ground state when $JT \gtrsim 40$; RL-QAOA gives inferior solutions in these settings. For the 2D Ising model, while the performance of RL-QAOA is similar with that of MCTS-QAOA at $JT=60$, the performance of RLQAOA at $JT=30, 40$ and 50 is still inferior to that of MCTS-QAOA. For the LMG model, the quality of the gate sequence found by RL-QAOA further decreases when $JT>500$, and MCTS-QAOA is significantly more robust.

The inferior performance of RL-QAOA is directly related to the joint parameterization used in RL-QAOA for the continuous and discrete policies. Since RL-QAOA optimizes

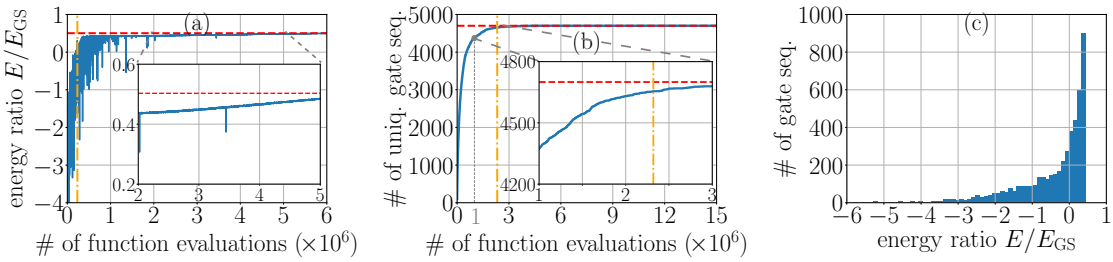



Figure 5.3: **Analysis of RL-QAOA using the LMG test**: (a): Energy ratio versus number of function evaluations; (b): Number of unique gate sequences encountered versus number of function evaluations; (c): Histogram of the rewards received by the algorithm in the first 5000 iterations. The horizontal red line in the left / middle panel represents the maximal energy ratio and the maximal number of unique gate sequences encountered during the optimization, respectively. The orange line marks the transition between two stages of the training process.

the continuous and discrete variables simultaneously, for each discrete sequence, the level of accuracy of the continuous optimization can be relatively low. Consequently, the optimizer can get stuck at a suboptimal discrete sequence.

To illustrate this behavior, we analyze the training of RL-QAOA using the LMG model with $(JT, N, q) = (1500, 100, 8)$ and noiseless rewards. Figure 5.3 summarizes the performance of RL-QAOA. Here by function evaluation we mean the computation of the objective function in equation 5.2.2. From Figure 5.3(a) and Figure 5.3(b), it is clear that the training process can be divided into two distinct stages, and the transition between the two stages is marked by the dashed-dotted vertical lines[4]. In stage I, which is to the left of the vertical lines, the number of unique gate sequences encountered by RL-QAOA quickly increases, while the energy ratio keeps oscillating below zero, which suggests that RL-QAOA focuses on exploration and the continuous optimization is done only very roughly within stage I. In stage II, which is to the right of the vertical lines, the number of unique gate sequences encountered by RL-QAOA stops to grow, while the energy ratio obtained grow above zero and eventually gets stuck at around 0.5, which means that the algorithm stops its exploration and focuses on the optimization of the continuous variables for a fixed gate sequence with stage II. The overall performance of RL-QAOA can highly depend on the discrete gate sequence that the RL-QAOA agent decides to exploit. In the next section, we demonstrate that

[4]These vertical lines are drawn at the point where the number of discrete protocol gate sequences drops to 10% of the total number within a single mini-batch

both the exploration and the exploitation phases in RL-QAOA can be suboptimal in this example, but the main issue is related to the suboptimal discrete sequences found in the exploration phase.

## Landscape of the discrete optimization and comparison with random search

In order to further understand the relative importance of continuous optimization versus discrete optimization for the generalized QAOA, we study the energy landscape of discrete optimization. For each discrete gate sequence, we perform numerical optimization to identify the *best* continuous parameters $\{\alpha_j\}$, and record the corresponding energy ratio.

**Energy landscape of discrete optimization.** – A profile of the discrete optimization landscape can be given by solving the corresponding continuous optimization individually on a random subset of all possible gate sequences; if the total number of possible gate sequences $|\mathcal{A}|(|\mathcal{A}|-1)^{q-1}$ is relatively small, this subset can actually be chosen to include all sequences. In our numerical implementation, each discrete gate sequence is sent to the natural policy gradient solver described in Section 5.4, and the continuous variables are solved for different $JT$ regime. Histograms for the energy ratios obtained can then be drawn.

Figure 5.4 shows the discrete landscape for the LMG model and the 1D Ising model, respectively, where the parameters of the ansatz are $(|\mathcal{A}|, q) = (5, 8)$, and the total number of gate sequences is thus 81920. From the histogram plot, most gate sequences are concentrated at the right-most peak in the large $JT$ regime. Far from searching "a needle in a haystack", this showcases that there are plenty of "good"[5] gate sequences assuming that each continuous optimization parameter is well solved. Note that the behavior is significantly different from the discrete-only optimization, where the landscape has been shown to feature transitions between glassy, correlated and uncorrelated phases [82]. To the best of our knowledge, the existence of many good discrete gate sequences in the QAOA-type variational quantum algorithms has not been reported in the literature.

For the LMG model with total gate duration $JT = 1500$ (cf. Figure 5.4), while most energy ratios fall into the cluster above 0.9, there is a smaller cluster located at 0.5. The energy ratio obtained by RL-QAOA (cf. Figure 5.3) falls into this cluster, which is depicted by the red dashed line, while the green dashed line shows the energy

[5] "Good" gate sequences here means the optimized energy ratio is close to the optimal energy ratio obtainable within the generalized QAOA ansatz.

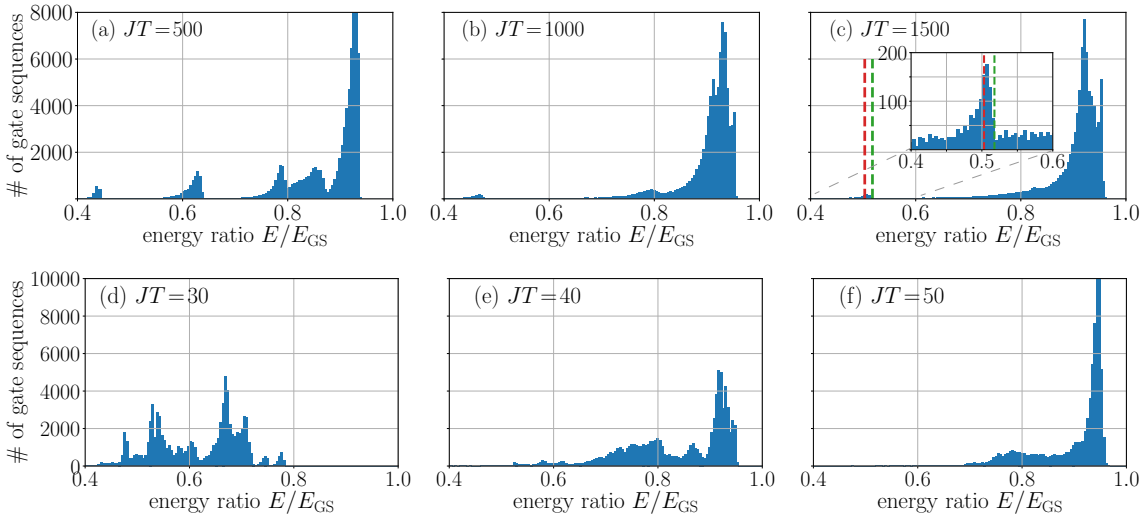



Figure 5.4: **Discrete landscape of the LMG model (a, b, c) and the 1D Ising model (d, e, f)**: Histograms of the energy ratio optimized by the improved natural gradient solver for $JT = 500, 1000, 1500$, respectively. $N_{\text{hist}} = 81920$ samples are chosen from the discrete gate sequences of generalized QAOA with parameters $q = 8$, $|\mathcal{A}| = 5$ and $N = 100$ (LMG) or $N = 8$ (1D Ising). The dashed red line in the top right panel shows the energy ratio achieved by RL-QAOA in Figure 5.3; the green dashed line shows the energy ratio obtained by the NPG solver for the same gate sequences.

ratio obtained by the natural policy gradient solver with the same gate sequence. The green line corresponds to a higher energy ratio than the red line, which means that the optimization of the continuous variables in the second stage of RL-QAOA is not as good as the NPG solver, and the difference between the two lines indicates the suboptimality caused by the exploitation. However, the suboptimality of the RLQAOA solution is mainly due to the exploration stage, since the discrete sequence that RL-QAOA chooses to exploit represents a suboptimal local optimum that belongs to a cluster much smaller than the rightmost one in the histogram. The top right panel of Figure 5.4 also verifies the claim that RL-QAOA only does a rough optimization on the continuous variables before it stops exploration, since the energy ratios displayed there are mostly above 0.4, while the energy ratio obtained in optimization stage I is mainly negative. While the landscape of the hybrid optimization is challenging for RL-QAOA, the proposed method MCTS-QAOA is able to deal with it by using a noise-resilient solver for the continuous variables (NPG), and by exploring the discrete variables constantly using MCTS.

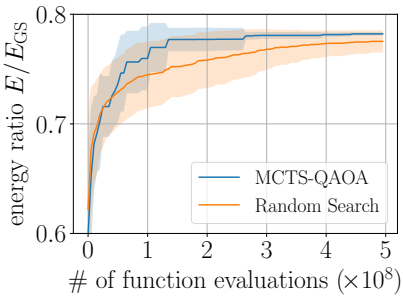



Figure 5.5: **Comparison between MCTS-QAOAand Random Search**: The blue and orange curves correspond to MCTS-QAOA and random search, respectively. The physics system is the 1D Ising model with duration $JT = 30$, which corresponds to Figure 5.2 (a). The generalized QAOA parameters are $q = 8$ and $|\mathcal{A}| = 5$. The horizontal axis is the number of function evaluations (with the function evaluation in the continuous optimization taken into account), and the vertical axis is the energy ratio. The shaded area for both algorithms represents the standard deviation across ten different random initializations.

For the 1D Ising model with total gate duration $JT = 30$ shown in the bottom left panel, where the rightmost cluster is not the largest. This means that in this setting, it is more difficult to find a gate sequence in the rightmost cluster when the random search is used. We examine the performance of random search and MCTS-QAOA using this example in the next part.

**Comparison with random search.** – A recent work [206] points out that advanced RL methods *need not* outperform simpler methods such as random search. In fact, if there is no specific structure in a problem, a random search algorithm might be as efficient as any sophisticated algorithm. In addition, from the landscape illustrated in the previous histograms, one can see that, *for the models we investigated*, there are lots of gate sequences with relatively high energy ratios *provided that* the continuous protocols are optimized. Therefore, it is natural to compare MCTS-QAOAagainst the random search algorithm[6]. For a fair comparison, we assume that the continuous optimization in both cases is solved by the natural policy gradient algorithm, and the difference only lies in the discrete optimization. In Figure 5.5, the best energy ratio in the training history is shown for the two methods, and one sees that MCTS-

[6]The random search algorithm also uses a two-level optimization, where the continuous optimization is solved by the policy gradient algorithm and the discrete optimization uses the random search. Since we assume no prior knowledge, the random search would be uniformly random on the discrete search space.

QAOAconsistently outperforms the random search across different random seeds. MCTS-QAOAnot only finds better gate sequences much faster, but also gives a smaller variance across different realizations. It is clear that instead of doing the search uniformly and treating each protocol as equally important, the tree statistics in MCTS-QAOAbetter guides into a more promising search direction.

## 5.6 Conclusion and discussions

In this paper, we study a continuous-discrete variational quantum algorithm for the generalized QAOA ansatz. To solve this hybrid optimization problem, we design a novel algorithm that combines the Monte Carlo tree search (MCTS) algorithm, a powerful method in exploring the discrete sequence, with an improved noise-robust policy gradient solver for the continuous duration variables of a fixed gate sequence. The proposed algorithms effectively generate robust quantum control where the prior methods fail.

In this context, we expect that random search algorithms cannot efficiently determine the best gate sequence if noisy rewards are used, while MCTS-QAOAis able to mitigate the noise and provide robust choice of gate sequence with the help of the tree structure it maintains. Moreover, it is possible for MCTS-QAOA to further reduce the number of evaluations by assigning different number of iterations for different gate sequences, e.g., it can assign more iterations for the more promising gate sequences. Also, MCTS-QAOA allows for the application of transfer learning using the tree statistics, which is not possible for the random search.

There are a number of possible ways to extend the problem presented in this paper:

**Learning based guided search.** – MCTS can be possibly guided by a learned functional approximator, such as neural networks or tensor networks. We have also tried the implementation of AlphaZero in the same experimental settings. However, the neural network based method does not work better than the simple MCTS. We find that the value function mapping from the discrete gate sequences to the score was quite hard to learn. One reason might be that the continuous policy gradient will try the best to optimize the energy ratio to the highest, thus making this mapping from discrete sequences to score, highly non-linear. Also, in terms of sampling efficiency, the neural network based approach needs lots of samples to fit the function, which is a heavy overhead compared to the simple MCTS approach. Nevertheless, the question remains open as to how to upgrade MCTS to a guided search.

**Amortized computation.** – The computation within the policy gradient solver for

different gate sequences can possibly be amortized. Currently, the continuous and discrete optimizations are separated. If some functional can be learned by replaying the data during the policy gradient iteration, the number of function evaluations can be further reduced. However, one difficulty in the quantum setting is that we do not have access to the quantum state, and thus we cannot learn a mapping taking the quantum state as input, unless we apply non-trivial quantum tomography. Therefore, how to reuse the past information and make the MCTS-QAOA algorithm quickly adaptive in physical setting remains to be investigated. Advanced algorithms like meta learning can be explored in the future work.

**Budget-aware variational quantum algorithms.** – A point of high interest is the design of budget-aware variational quantum algorithms. The importance of sample efficiency in the quantum setting can never be overemphasized. Each run of a quantum circuit can be expensive and quantum decoherence noise is usually not stationary over time. The budget-awareness property can be naturally incorporated in the MTCS framework. Making use of the tree structure, the adaptive algorithm would distribute more function evaluation budget to the most-visited or more promising nodes. The current algorithm likely operates in a budget-sufficient regime and uses the same amount of budget for each discrete gate sequences. We hope the adaptive algorithm can hit the sweet spot in the middle, i.e., use the right amount of computational budget and still compute the best possible gate sequence design. We hope that the present work will accelerate the research of budget-aware variational quantum algorithms in a realistic setting.

## 5.7 Related works

**Hybrid optimization**: The generalized QAOA ansatz introduces a discrete and continuous control problem: the discrete degrees of freedom are the gates/unitaries that define the control protocol, while the continuous degrees of freedom are the gate duration. Most reinforcement learning algorithms [195, 30, 310] typically deal with the control of either discrete or continuous degree of freedom, and hardly consider the discrete and continuous control simultaneously in the policy. Even though the continuous control can always be discretized, it is always beneficial and desirable to consider discrete and continuous variables together, without loss of the flexibility of continuous control. Furthermore, the idea of continuous and discrete optimization can be quite general, and shows up in real world application like robotics [233, 84] and strategic games [317]. Combining the discrete and continuous control together, the control capability of the algorithm can be quickly enhanced. In general, the discrete variables are usually chosen as the categories of actions, and the continuous variables

will are naturally given by the strength for each specific action. Our work aims to shed light on the hybrid control in the field of quantum control, and we also hope it will accelerate the research of hybrid discrete-continuous optimization algorithms in the wider community.

**Counter diabatic driving**: Counter diabatic driving [280, 139], an example of a shortcut to adiabaticity (STA), introduces an extra auxiliary counter-diabatic (CD) Hamiltonian to suppress transitions (or excitations) between instantaneous eigenvalues.

For a given quantum state $|\psi\rangle$ evolving under a time dependent Hamiltonian $H_0(\lambda(t))$, the Schrödinger equation reads as

$$\begin{aligned} i\hbar\partial_t |\psi\rangle &= H_0(\lambda(t)) |\psi\rangle , \\ |\psi_i\rangle = |\psi_{\mathrm{GS}}(\lambda=0)\rangle , |\psi_*\rangle &= |\psi_{\mathrm{GS}}(\lambda=1)\rangle . \end{aligned} \tag{5.7.1}$$

In the rotating frame, Hamiltonian remains stationary under the unitary transformation $U(\lambda(t))$, i.e. in the instantaneous eigenbasis of Hamiltonian $H_0(\lambda)$. The wave function $|\tilde{\psi}\rangle = U(\lambda)|\psi\rangle$ in the rotating frame satisfies the following Schrödinger equation:

$$i\hbar\partial_t|\tilde{\psi}\rangle = \left(\tilde{H}_0(\lambda(t)) - \dot{\lambda}\tilde{\mathcal{A}}_\lambda\right)|\tilde{\psi}\rangle, \tag{5.7.2}$$

where $\tilde{H}_0(\lambda(t)) = U^\dagger H_0(\lambda(t))U, \tilde{\mathcal{A}}_\lambda = iU^\dagger\partial_\lambda U$. Specifically, instead of being diagonalized, the original Hamiltonian picks up an extra contribution due to the change in the parameter $\lambda(t)$, and the effective Hamiltonian becomes

$$H_0^{\mathrm{eff}} = \tilde{H}_0 - \dot{\lambda}\tilde{\mathcal{A}}_\lambda. \tag{5.7.3}$$

The idea of the CD driving is to evolve the system with the counterdiabatic Hamiltonian

$$H_{\mathrm{CD}}(t) = H_0 + \dot{\lambda}\mathcal{A}_\lambda. \tag{5.7.4}$$

Importantly, in the moving frame $H_{\mathrm{CD}}^{\mathrm{eff}}(t) = \tilde{H}_0$ is *stationary and no transitions occur.*

However, in practice, the precise counter-diabatic Hamiltonian is intractable and usually approximated by different methods. A good number of prior works [242, 135, 139, 138, 361, 341, 140] are based on the concept of a variational approximation to the CD Hamiltonian [280]. Most of these works typically make use of an analytically computed expression available for few-qubit systems; they first derive the continuous form of the variational gauge potential, and then discretize the underlying dynamics using the Trotter-Suzuki formula In this work, we aim to bypass these constraints by applying the variational generalized QAOA ansatz using additional gates, generated

by terms that occur in the approximation to the variational adiabatic gauge potential. These extra gates can provide a shortcut to the preparation of the ground state, compared to the original alternating QAOA ansatz. Physically, this shortcut results in shorter circuit simulation times, which provices a significant advantage on noisy NISQ devices.

**AutoML and neural architecture search**: Automatic machine learning or AutoML has recently attracted lots of attentions as it reduces human efforts in designing the neural architecture from experience and instead leverage the computational power to search the best configuration. One of the most pronounced examples are neural architecture search (NAS) and their variants [367, 99, 198, 51, 259], where reinforcement learning or evolutionary strategies are used to find a better network architecture. Inspired by the success of AutoML, the architecture of quantum circuits can also be improved by machine learning algorithms, such as the quantum version of Neural Architecture Search [319, 320, 357, 356, 182]. These prior works interpret the problem as quantum compiling problems, which assembles quantum gates in the low level. Instead of exposing a huge number of choice alternatives for the search algorithms, our work specially uses the variational gauge potentials as the Hamiltonian pool for the search algorithm in a computation-efficient way. Compared with QAOA, MCTS-QAOA has more degree of freedom to approximate the unitary operator; compared with the quantum compiling, it does not search gates in the low level due to the constraint of computations. From this perspective, our method hits the sweet spot between the expressivity and efficiency.

## 5.8 Setup of physical models

We first give a brief review on the physical models used in the numerical experiments. In all experiments, we choose the target state as the ground state of the Hamiltonian $H$, denoted $|\psi_{\mathrm{GS}}(H)\rangle$. The spin-1/2 matrices describing spin $i$ are denoted by $X_i, Y_i, Z_i$. In contrast to the models considered in [350, 347], the Hamiltonians used in this work is normalized by its operator norm $\|H\|$, i.e., we use $H/\|H\|$ instead of the original Hamiltonian $H$. The reason for introducing the normalized Hamiltonian is as follows. For generic Hamiltonians $H$ (e.g., sparse matrices), the cost of performing a Hamiltonian evolution $e^{-iH\alpha}$ on a quantum device is $\Omega(\|H\|\alpha)$ [28, 201]. Due to the potential differences between the Hamiltonian norms in the Hamiltonian pool $\mathcal{A}$, using a normalized Hamiltonian $H/\|H\|$ (the corresponding duration parameter $\alpha$ is thus multiplied by $\|H\|$) can lead to a more realistic estimate of the cost of the quantum simulation. Due to this multiplication factor, the duration shown in the results below is larger than that presented in [350, 347].

**One-dimensional (1D) Ising model**

The spin-1/2 Ising Hamiltonian reads as:

$$H \;=\; H_1+H_2, \quad H_1=\sum_{i=1}^{N} JZ_{i+1}Z_i+h_zZ_i, \quad H_2=\sum_{i=1}^{N} h_xX_i,$$

where $N$ is the number of qubits and the parameters are set as $h_z/J=0.4523$ and $h_x/J=0.4045$ [162]. These parameters are close to the critical line of the model in the thermodynamic limit, where the quantum phase transition occurs. They are also reported in Ref. [209] to be in the most challenging parameter region using QAOA. We use periodic boundary conditions here. The initial state for this experiment is given by $z$-polarized product state, i.e. $|\psi_{\text{init}}\rangle=|\uparrow\cdots\uparrow\rangle$.

For the Hamiltonian pool, we use $\mathcal{A}=\left\{J\frac{H_1}{||H_1||}, J\frac{H_2}{||H_2||}, J\frac{A_1}{||A_1||}, J\frac{A_2}{||A_2||}, J\frac{A_3}{||A_3||}\right\}$, where $A_1=\sum_{i=1}^{N} Y_i,\ A_2=\sum_{i=1}^{N} X_iY_i+Y_iX_i,\ A_3=\sum_{i=1}^{N} Z_iY_i+Y_iZ_i$. The operators $A_j$ are precisely the first three terms in the expansion for the adiabatic gauge potential of the translation-invariant 1D Ising model [347].

**Two-dimensional (2D) Ising model**

The 2D spin-1/2 transverse-field Ising model reads:

$$H=H_1+H_2, \quad H_1=J\sum_{\langle i,j\rangle} Z_iZ_j+h_z\sum_j Z_j,\ H_2=\sum_j h_xX_j,$$

where $\langle i,j\rangle$ denotes nearest neighbors on the square lattice. The model parameters are set as $h_z/J=2$ and $h_x/J=3$. The initial state is $|\psi_{\text{init}}\rangle=|\uparrow\rangle$, i.e. $z$-polarized product state on 2D lattice.

For the Hamiltonian pool, we use $\mathcal{A}=\left\{J\frac{H_1}{||H_1||}, J\frac{H_2}{||H_2||}, J\frac{A_1}{||A_1||}, J\frac{A_2}{||A_2||}, J\frac{A_3}{||A_3||}\right\}$, where $A_1=\sum_j Y_j,\ A_2=\sum_{\langle i,j\rangle} X_iY_j+Y_iX_j,\ A_3=\sum_{\langle i,j\rangle} Z_iY_j+Y_iZ_j$.

**Lipkin-Meshkov-Glick (LMG) model**

The Lipkin-Meshkov-Glick (LMG) model [196] reads:

$$H=H_1+H_2, \quad H_1=-\frac{J}{N}\sum_{i,j=1}^{N} X_iX_j,\ H_2=h\sum_{j=1}^{N}\left(Z_j+\frac{1}{2}\right),$$

where $J$ is the interactions trength, and $h$ stands for the magnetic field strength. The LMG model preserves the total spin, and the ground state is contained in an $N+1$

dimensional subspace due to this symmetry. This makes the LMG model particularly interesting because it allows us to simulate its dynamics for a large number of spins, where many-body effects, such as collective phenomena, dominate the physics of the system.

For instance, in the thermodynamic limit $N \to \infty$, the LMG model exhibits a quantum phase transition at $h_c/J=1$ [38]. The transition is between a ferromagnetic (FM) order in the ground state in the $x$-direction ($h/J \ll 1$), and the paramagnetic order ($h/J \gg 1$).

For the Hamiltonian pool, we use $\mathcal{A}=\left\{J\frac{H_1}{||H_1||}, J\frac{H_2}{||H_2||}, J\frac{A_1}{||A_1||}, J\frac{A_2}{||A_2||}, J\frac{A_3}{||A_3||}\right\}$, where

$$\begin{aligned} A_1 &= \sum_{j=1}^{N} Y_j, \\ A_2 &= \frac{1}{N}\left(\sum_{j=1}^{N} Y_j\right)\left(\sum_{j=1}^{N} X_j\right) + \frac{1}{N}\left(\sum_{j=1}^{N} X_j\right)\left(\sum_{j=1}^{N} Y_j\right), \\ A_3 &= \frac{1}{N}\left(\sum_{j=1}^{N} Y_j\right)\left(\sum_{j=1}^{N}\left(Z_j+\frac{1}{2}\right)\right) + \frac{1}{N}\left(\sum_{j=1}^{N}\left(Z_j+\frac{1}{2}\right)\right)\left(\sum_{j=1}^{N} Y_j\right). \end{aligned} \quad (5.8.1)$$

## 5.9 Noise models

An essential part of our study is the performance of the algorithms in the presence of noise. As mentioned in the main text, noise sets the current bottle neck for reliable quantum computation. Therefore, it is of primary importance for the near-term utility of quantum computers to develop stable and noise-robust manipulation algorithms.

We use the following three noise models in our numerical experiments: (i) classical measurement noise, (ii) quantum measurement error which micmic the situation on present-day NISQ devices, and (iii) gate rotation error noise.

***Classical measurement Gaussian noise*** is added to the cost function according to

$$\mathcal{L}_\gamma(\boldsymbol{\theta}) = \mathcal{L}(\boldsymbol{\theta}) + \epsilon_\gamma,$$

where $\epsilon_\gamma \sim \mathcal{N}(0, \gamma^2)$ and $\gamma$ denotes the noise strength, and $\mathcal{N}$ is the normal distribution. Gaussian noise models various kinds of uncertainty present in experiments using an additive Gaussian random variable, which follows from the Central Limit theorem.

***Quantum measurement noise***:

$$\mathcal{L}_Q(\boldsymbol{\theta}) = \mathcal{L}(\boldsymbol{\theta}) + \epsilon_Q,$$

where the noise strength depends on the strength of the energy quantum fluctuations

$$\Delta\mathcal{E} = N^{-1}\sqrt{\langle\psi(T)|H^2|\psi(T)\rangle - \langle\psi(T)|H|\psi(T)\rangle^2},$$

and $\epsilon_Q$ is randomly sampled from $\mathcal{N}(0, \Delta\mathcal{E}^2)$. Quantum noise models the uncertainty arising from quantum measurements. For instance, quantum fluctuations are large when the evolved quantum state is far away from the target, while they decrease when the final state approaches the target ground state.

***Gate rotation error noise***:

$$\mathcal{L}_\delta(\boldsymbol{\theta}) = \mathcal{L}(\boldsymbol{\theta}'), \;\; \boldsymbol{\theta}' = (\{\alpha_i + \alpha_i\epsilon_i\}_{i=1}^q, \boldsymbol{\tau})$$

where gate error strengths are multiplicative and the corresponding ratios are $\epsilon_i \sim \mathcal{N}(0, \delta^2)$ for some simulation parameter $\delta$ which controls the noise strength. Gate rotation errors [293] present yet another common noise source, which arises due to imperfections or lack of calibration in the quantum computer hardware.

## 5.10 Details for the natural policy gradient with entropy regularization

For a general $d$-dimensional Gaussian distribution $\mathcal{N}(\mu, \Sigma)$, the Shannon entropy is defined as $\mathbb{E}(-\log(p(x)))$, where $p(x) = (2\pi)^{-\frac{d}{2}}|\Sigma|^{-\frac{1}{2}}\exp\big(-\frac{1}{2}(x-\mu)^\top\Sigma^{-1}(x-\mu)\big)$. Hence

$$\begin{aligned}
\mathbb{E}(-\log(p(x))) &= -\mathbb{E}\log\left[(2\pi)^{-\frac{d}{2}}|\Sigma|^{-\frac{1}{2}}\exp\left(-\frac{1}{2}(x-\mu)^\top\Sigma^{-1}(x-\mu)\right)\right] \\
&= \mathbb{E}\left[\frac{d}{2}\log 2\pi + \frac{1}{2}\log|\Sigma| + \frac{1}{2}(x-\mu)^\top\Sigma^{-1}(x-\mu)\right] \\
&= \frac{d}{2}\log 2\pi + \frac{1}{2}\log|\Sigma| + \frac{1}{2}\mathbb{E}(x-\mu)^\top\Sigma^{-1}(x-\mu) \\
&= \frac{d}{2}\log 2\pi + \frac{1}{2}\log|\Sigma| + \frac{1}{2}\mathbb{E}\,\mathrm{Tr}\big((x-\mu)^\top\Sigma^{-1}(x-\mu)\big) \\
&= \frac{d}{2}\log 2\pi + \frac{1}{2}\log|\Sigma| + \frac{1}{2}\mathbb{E}\,\mathrm{Tr}\big(\Sigma^{-1}(x-\mu)(x-\mu)^\top\big) \\
&= \frac{d}{2}\log 2\pi + \frac{1}{2}\log|\Sigma| + \frac{d}{2}.
\end{aligned}$$

Omitting the constants, it is equivalent to take the entropy as $\frac{1}{2}\log|\Sigma|$. For the model used, the probability distribution is a product of normal distribution, i.e., $\Sigma$ is a diagonal matrix with length $q$ and diagonal elements $\sigma_i$, so the corresponding entropy function is $\mathbb{E}(-\log(p(x))) = \sum_{i=1}^{q}\log\sigma_i$.

In the implementation, we adopt the parameterization $\sigma_i = \exp(t_i)$ to assure that $\sigma_i$ is positive. Then for the distribution $\mathcal{N}(\mu_i, \sigma_i)$, we have

$$\log p_i(x) = -\frac{(x-\mu_i)^2}{2\sigma_i^2} - \log\sigma_i - \frac{1}{2}\log(2\pi) = -\frac{1}{2}(x-\mu_i)^2 e^{-2t_i} - t_i - \frac{1}{2}\log(2\pi),$$

and

$$\nabla \log p_i(x) = ((x-\mu_i)e^{-2t_i}, (x-\mu_i)^2 e^{-2t_i} - 1)^\top,$$

where the gradient is taken with respect to the parameters. Since $\{\delta_i\}_{i=1}^q$ are independent, the Fisher information matrix is a block diagonal matrix with the $i$-th block equal to

$$F_i = \mathbb{E}\nabla\log p_i(x)\nabla\log p_i(x)^\top = \mathbb{E}\begin{bmatrix} \frac{(x-\mu_i)^2}{\sigma_i^4} & \frac{(x-\mu_i)^3}{\sigma_i^3} - \frac{(x-\mu_i)}{\sigma_i} \\ \frac{(x-\mu_i)^3}{\sigma_i^3} - \frac{(x-\mu_i)}{\sigma_i} & \frac{(x-\mu_i)^4}{\sigma_i^4} - \frac{2(x-\mu_i)^2}{\sigma_i^2} + 1 \end{bmatrix} = \begin{bmatrix} \frac{1}{\sigma_i^2} & 0 \\ 0 & 2 \end{bmatrix}.$$

Recall that for a fixed gate sequence $\boldsymbol{\tau}$, we set $R(\boldsymbol{\delta}) = -E\left(\left\{\frac{T\mathfrak{g}(\delta_j)}{\sum_k \mathfrak{g}(\delta_k)}\right\}_{j=1}^q, \boldsymbol{\tau}\right)/N$, where $\mathfrak{g}$ denotes the sigmoid function, and

$$\mathcal{J}(\{\mu_j,\sigma_j\}_{j=1}^q) = \mathbb{E}_{\delta_j\sim\mathcal{N}(\mu_j,\sigma_j)} R(\boldsymbol{\delta}) + \beta_S^{-1}\sum_{j=1}^{q}\log\sigma_j.$$

Hence the gradient of $\mathcal{J}$ is

$$\mathbb{E}R(\boldsymbol{\delta})\nabla\log p(\boldsymbol{\delta}) + \beta_S^{-1}\nabla\sum_{j=1}^{q}\log\sigma_j,$$

where the gradient is taken with respect to the parameters, and $p(\boldsymbol{\delta}) = \prod_{i=1}^q p_i(\delta_i)$. Therefore, the unbiased estimators for the variables are

$$\begin{bmatrix} \frac{\partial\mathcal{J}}{\partial\mu_j} \\ \frac{\partial\mathcal{J}}{\partial t_j} \end{bmatrix} \leftarrow \begin{bmatrix} R(\boldsymbol{\delta})\xi_j/\sigma_j \\ R(\boldsymbol{\delta})(\xi_j^2-1) + \beta_S^{-1} \end{bmatrix},$$

where $\xi_j$ are independent standard normal variables and $\delta_j = \sigma_j\xi_j + \mu_j$. As a result, the unbiased estimators for the natural gradient direction become

$$F_j^{-1}\begin{bmatrix} \frac{\partial\mathcal{J}}{\partial\mu_j} \\ \frac{\partial\mathcal{J}}{\partial t_j} \end{bmatrix} \leftarrow \begin{bmatrix} \sigma_j R(\boldsymbol{\delta})\xi_j \\ \frac{1}{2}R(\boldsymbol{\delta})(\xi_j^2-1) + \frac{1}{2}\beta_S^{-1} \end{bmatrix},$$

since $F$ is a block diagonal matrix with the $j$-th block given by $F_j$.

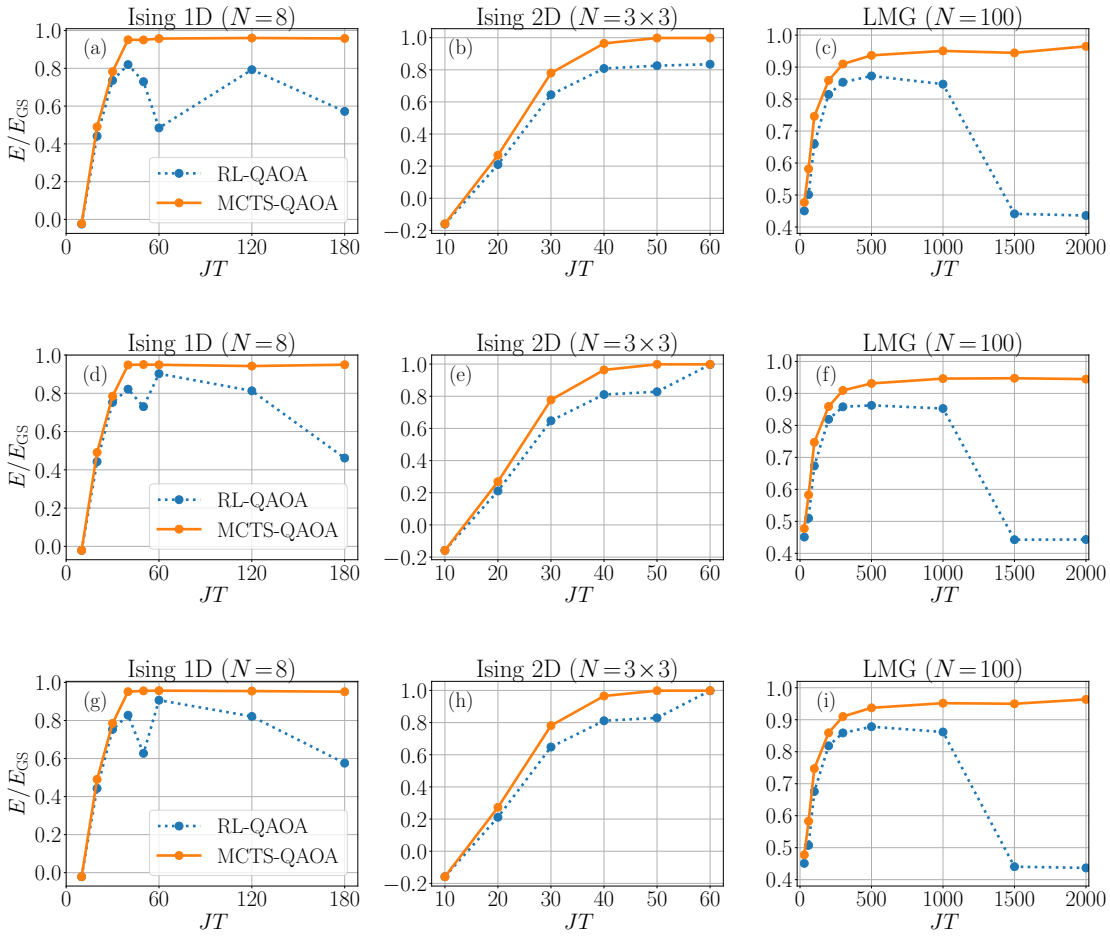



Figure 5.6: (**experiment with other types of noise models or without noise**) comparison between MCTS-QAOA and RL-QAOA. The physics setup is the same as that in Figure 5.2. (a-c): Gaussian noise with $\gamma = 0.1$; (d-f): gate rotation noise with $\delta = 0.1$; (g-i): experiments without noise (cf. Section 6.11).

## 5.11 Additional experiment results

In Section 5.5, we have presented a comparison between the RL-QAOA method and MCTS-QAOA for three different physics models with the quantum noise. In this section, we report the test results with the other types of noise, namely the results with the Gaussian noise, the results with the gate rotation error, and the results when no noise is considered (cf. Section 6.11). We can observe from the comparison that MCTS-QAOA's performance is much more stable and accurate.

From Figure 5.6, one can observe similar behavior the two methods as in Section 5.5, i.e., MCTS-QAOA outperforms RL-QAOA in all settings and the gaps grow larger in the regime of large total gate durations. The raw data for the energy ratio obtained by MCTS-QAOA is summarized in Table 5.2 (highlighted in bold), which offers a more visually and quantitatively convenient comparison across different models.

| $JT$ | Gate rotation noise | Quantum noise | Gaussian noise | No noise |
|---|---|---|---|---|
| | | $(E/E_{\mathrm{GS}})$ | | |
| (*Model*) | | (a) **Ising 1D** | | |
| 10.0 | −0.0208 (−0.0208) | −0.0219 (−0.0238) | −0.0225 (−0.0209) | −0.0210 (−0.0207) |
| 20.0 | 0.4907 (0.4884) | 0.4862 (0.4863) | 0.4903 (0.4905) | 0.4907 (0.4908) |
| 30.0 | 0.7849 (0.7844) | 0.7830 (0.7796) | 0.7825 (0.7833) | 0.7850 (0.7850) |
| 40.0 | 0.9481 (0.9486) | 0.9521 (0.9477) | 0.9512 (0.9513) | 0.9516 (0.9527) |
| 50.0 | 0.9503 (0.9499) | 0.9499 (0.9564) | 0.9505 (0.9581) | 0.9559 (0.9574) |
| 60.0 | 0.9489 (0.9614) | 0.9526 (0.9540) | 0.9576 (0.9560) | 0.9570 (0.9621) |
| 120.0 | 0.9424 (0.9495) | 0.9543 (0.9548) | 0.9606 (0.9524) | 0.9548 (0.9602) |
| 180.0 | 0.9495 (0.9415) | 0.9486 (0.9502) | 0.9582 (0.9556) | 0.9514 (0.9543) |
| (*Model*) | | (b) **Ising 2D** | | |
| 10.0 | −0.1586 (−0.1586) | −0.1645 (−0.1610) | −0.1589 (−0.1614) | −0.1587 (−0.1587) |
| 20.0 | 0.2688 (0.2692) | 0.2663 (0.2672) | 0.2680 (0.2677) | 0.2730 (0.2688) |
| 30.0 | 0.7771 (0.7777) | 0.7786 (0.7800) | 0.7799 (0.7797) | 0.7812 (0.7812) |
| 40.0 | 0.9635 (0.9635) | 0.9641 (0.9651) | 0.9647 (0.9633) | 0.9654 (0.9635) |
| 50.0 | 0.9984 (0.9984) | 0.9979 (0.9982) | 0.9982 (0.9983) | 0.9985 (0.9985) |
| 60.0 | 0.9980 (0.9979) | 0.9978 (0.9981) | 0.9981 (0.9965) | 0.9986 (0.9984) |
| (*Model*) | | (c) **LMG** | | |
| 30.0 | **0.4775** (0.4774) | **0.4762** (0.4729) | **0.4766** (0.4770) | **0.4776** (0.4774) |
| 60.0 | **0.5828** (0.5828) | **0.5792** (0.5803) | **0.5818** (0.5815) | **0.5828** (0.5828) |
| 100.0 | **0.7471** (0.7467) | **0.7447** (0.7468) | **0.7459** (0.7460) | **0.7472** (0.7471) |
| 200.0 | **0.8591** (0.8592) | **0.8561** (0.8568) | **0.8583** (0.8587) | **0.8591** (0.8591) |
| 300.0 | **0.9093** (0.9098) | **0.9091** (0.9077) | **0.9095** (0.9093) | **0.9101** (0.9104) |
| 500.0 | **0.9312** (0.9368) | **0.9349** (0.9363) | **0.9367** (0.9367) | **0.9372** (0.9371) |
| 1000.0 | **0.9463** (0.9484) | **0.9522** (0.9475) | **0.9505** (0.9510) | **0.9518** (0.9523) |
| 1500.0 | **0.9474** (0.9436) | **0.9385** (0.9336) | **0.9444** (0.9524) | **0.9499** (0.9453) |
| 2000.0 | **0.9447** (0.9448) | **0.9646** (0.9496) | **0.9648** (0.9521) | **0.9636** (0.9562) |

Table 5.2: **Energy ratio obtained by MCTS-QAOA**: MCTS-QAOAusing the Hamiltonian pool without identity (**bold**, see Section 5.8) and with identity operator (gray in the parenthesis, see Section 5.13). Sector (a): 1D spin-1/2 Ising chain ($N=8$); Sector (b): 2D spin-1/2 Ising chain ($N=3\times 3$); Sector (c): LMG model ($N=100$) at $h/J=0.9$. We use $\gamma=0.1$ for Gaussian noise and $\delta=0.1$ for the gate rotation noise (see Section. 6.11).

## 5.12 Additional numerical results on the energy landscape

In Section 5.5 we reported the discrete landscape of the generalized QAOA ansatz under the condition that the continuous variables are solved with high quality with the improved NPG solver. Here we include the landscape under another physical model, i.e. 2D Ising model. We consider the case where $(|\mathcal{A}|, q) = (5, 8)$, and the total number of gate sequences is thus 81920. Similar to the plots displayed in Section 5.5, the landscape with a longer total duration ($JT = 50$) features a dominant cluster at the rightmost part of the histogram. When the total duration is smaller, the number of clusters increases, and is shifted to the left.

Figure 5.8 shows the influence of the parameter $h/J$ in the discrete landscape for the LMG model with gate duration $JT = 1500$ and $N = 100$. When $h/J = 0.8$ and $h/J = 0.99$, the rightmost peak in the energy ratio histogram gets close to 1, which means that reaching the ground state would be a easy task in these two cases. The more difficult cases lies in between, for example when $h/J = 0.95$. For the parameter $h/J = 0.9$ we choose in the main text, there is a bigger gap (cf. Fig. 5.4(c)) between the rightmost peak of the energy ratio and 1, which means the problem we choose to solve is relatively challenging.

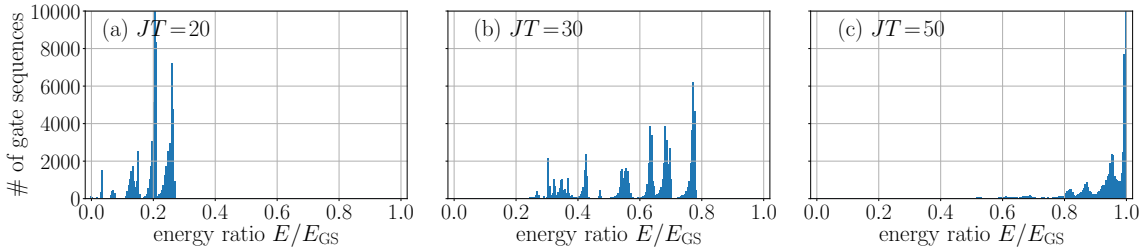



Figure 5.7: **Discrete landscape of 2D Ising model**: (a-c): Histograms of the energy ratio optimized by the improved natural gradient solver for $JT = 20, 30, 50$, respectively. $N_{\text{hist}} = 81920$ samples are chosen from the discrete gate sequences of generalized QAOA with parameters $q = 8$ and $|\mathcal{A}| = 5$.

## 5.13 Physical models with the identity action

The generalized QAOA ansatz provides us the freedom of adding different Hamiltonians to the Hamiltonian pool. One meaningful addition is the identity operator.

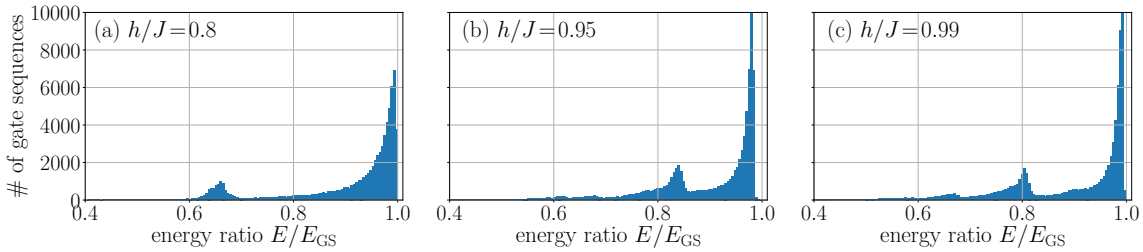



Figure 5.8: **Discrete landscape of LMG model with respect to different parameter** $h/J$: (a-c): Histograms of the energy ratio optimized by the improved natural gradient solver for $h/J = 0.8, 0.95$, and $0.99$, respectively with gate duration $JT = 1500$. $N_{\mathrm{hist}} = 81920$ samples are chosen from the discrete gate sequences of generalized QAOA with parameters $q = 8$, $|\mathcal{A}| = 5$ and $N = 100$. For the LMG model, the gap between the right-most peak and 1 is larger when $h/J$ is between 0.8 and 0.99.

Here, the identity operator corresponds to the identity gate that does not move the quantum state. If we take $\tilde{H} = \mathbf{0}$, then its corresponding unitary gate will be identity, i.e. $\exp\left(-i\tilde{H}\tilde{\alpha}\right) = \mathbf{I}$. This approach adds an extra amount of freedom to the optimization since the quantum control no longer needs to figure out how to *exactly* distribute the gate duration budgets to different gates so as to reach the ground state. In other words, the original optimization problem (see Eqn. 5.4.2) becomes the relaxed form:

$$\min_{\{\alpha_j\}_{j=1}^q} \left\{ E(\{\alpha_j\}_{j=1}^q, \boldsymbol{\tau}) : \sum_{j=1}^{q} \alpha_j \leq T;\ 0 \leq \alpha_j \leq T \right\}. \tag{5.13.1}$$

With the identity action, the extended action space becomes

$$\mathcal{A} = \left\{ \mathbf{0}, \frac{H_1}{||H_1||}, \frac{H_2}{||H_2||}, \frac{A_1}{||A_1||}, \frac{A_2}{||A_2||}, \frac{A_3}{||A_3||} \right\},$$

with the definitions shown in Sec. 5.8 for three different physics models.

In this setting, a similar behavior is observed as in Figure 5.2 and Figure 5.6, which is shown in Figure 5.9. We conclude that MCTS-QAOA outperforms RL-QAOA in all settings and MCTS-QAOAstill maintains a robust performance when that of RL-QAOA begins to deteriorate in the regime of large total gate durations. The raw data of the energy ratio obtained by MCTS-QAOA is reported in Table 5.2 (highlighted in gray), which also gives a direct comparison with the energy ratios obtained without the identity action. It can be seen that the performance of MCTS-QAOA in this setting is on par with the setting presented in the main text.

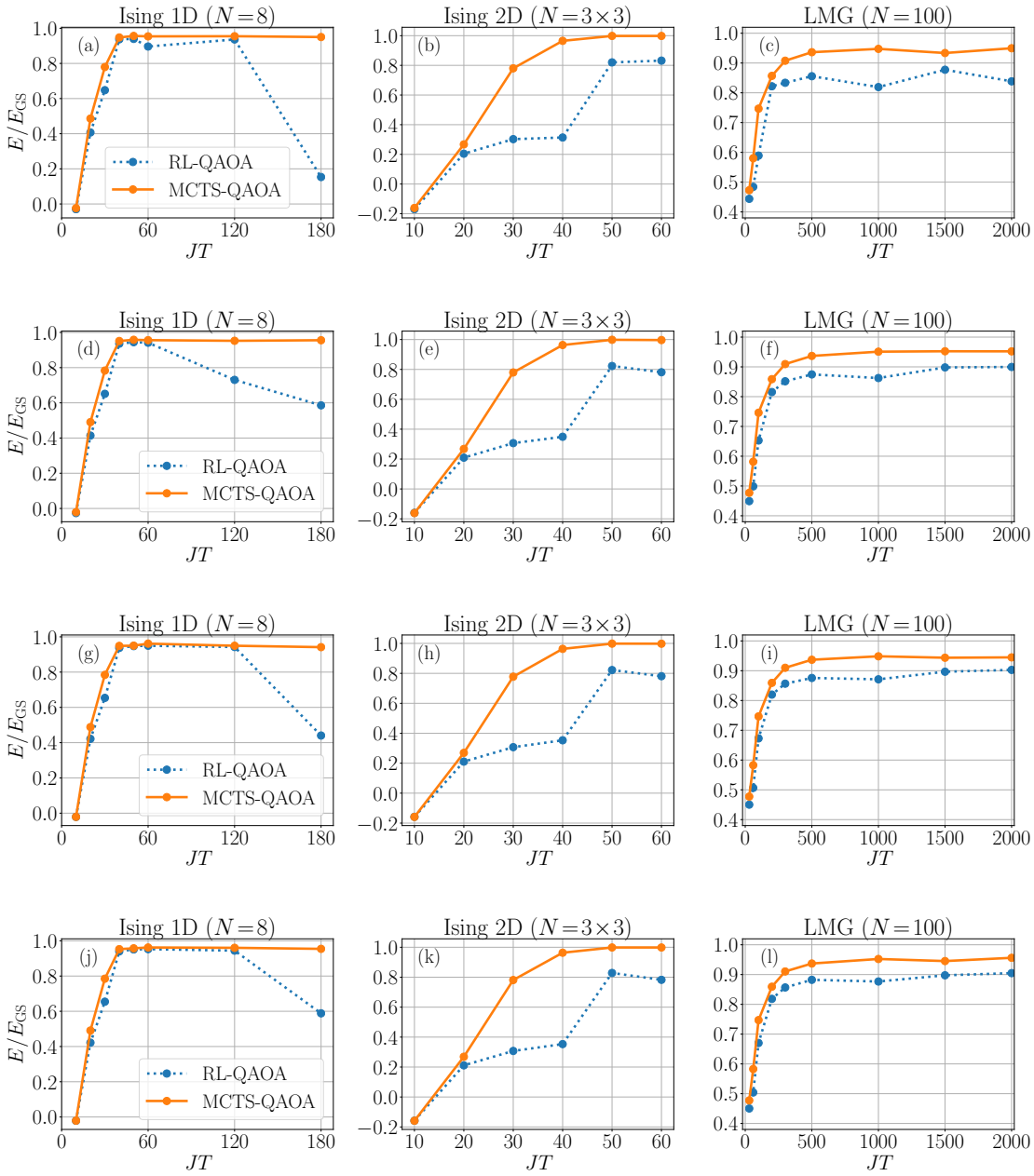



Figure 5.9: Comparison between MCTS-QAOA and RL-QAOA using the Hamiltonian pool with the identity operation. The physics setup is the same as that in Figure 5.2. (a-c): quantum measurement noise; (d-f): Gaussian noise with $\gamma = 0.1$; (g-i): gate rotation noise with $\delta = 0.1$; (k-l): experiments without noise (cf. Section 6.11).

# Chapter 6

# Random Coordinate Descent for optimizing parameterized quantum circuit

Variational quantum algorithms rely on the optimization of parameterized quantum circuits in noisy settings. The commonly used back-propagation procedure in classical machine learning is not directly applicable in this setting due to the collapse of quantum states after measurements. Thus, gradient estimations constitute a significant overhead in a gradient-based optimization of such quantum circuits. This paper introduces a random coordinate descent algorithm as a practical and easy-to-implement alternative to the full gradient descent algorithm. This algorithm only requires one partial derivative at each iteration. Motivated by the behavior of measurement noise in the practical optimization of parameterized quantum circuits, this paper presents an optimization problem setting that is amenable to analysis. Under this setting, the random coordinate descent algorithm exhibits the same level of stochastic stability as the full gradient approach, making it as resilient to noise. The complexity of the random coordinate descent method is generally no worse than that of the gradient descent and can be much better for various quantum optimization problems with anisotropic Lipschitz constants. Theoretical analysis and extensive numerical experiments validate our findings.

## 6.1 Introduction

Variational quantum algorithms have emerged as a promising application for near-term quantum devices, addressing various computational challenges with enhanced

efficiency [222, 63]. These algorithms encompass several notable approaches, such as the variational quantum eigensolver [244], the variational quantum simulation [26], the quantum approximate optimization algorithm [102, 63, 190], and quantum machine learning [186, 32, 277, 59]. They are designed to operate in a hybrid quantum-classical fashion [214, 100]. In these algorithms, the quantum component involves the implementation of parameterized quantum gate operations. By performing measurements, a cost function (and optionally, its gradient) is obtained as the output. The classical computational procedure then utilizes an iterative method to produce updates for the parameters, which are subsequently leveraged to refine and reprogram the quantum circuits. This iterative process continues until convergence is achieved, forming a feedback loop that continues to improve the algorithm's performance.

In variational quantum algorithms, the optimizable parameters are defined within parameterized quantum circuits (PQCs) [27, 240, 286, 5]. A PQC is a sequence of unitary operators represented by parameterized quantum gates that can be readily implemented on a quantum computer. Assuming we are working in an $n$-qubit Hilbert space, a parameterized quantum circuit can be expressed as follows:

$$U(\boldsymbol{\theta}) = \prod_{j=1}^{J} U_j(\boldsymbol{\theta}_j) W_j. \tag{6.1.1}$$

Here, $\boldsymbol{\theta} = \{\boldsymbol{\theta}_j\}_{j=1}^{J}$ are the parameters that we need to optimize, $U_j(\boldsymbol{\theta}_j) \in \mathbb{C}^{2^n \times 2^n}$ are the parameterized unitary operators, and $W_j \in \mathbb{C}^{2^n \times 2^n}$ are fixed unitary operators. For instance, a simple example of a PQC consisting only of one-qubit Pauli rotation operators takes the form

$$U_j(\boldsymbol{\theta}_j) = \bigotimes_{m=1}^{M} e^{-i\theta_{j,k_{j,m}} \sigma_{j,k_{j,m}}},$$

where $\sigma_{j,k_{j,m}} \in \mathbb{C}^{2\times 2}$ is a single-qubit Pauli matrix that acts on $k_{j,m}$-th qubit, $\theta_{j,k_{j,m}}$ represents one of the parameters in $\boldsymbol{\theta}$, and $W_j$'s can be used to represent quantum gates that do not require parameterization, such as the controlled-NOT (CNOT) gate.

Let $d$ be the dimension of the parameters, and we write $\boldsymbol{\theta} = (\theta_1, \theta_2, \cdots, \theta_d)$. We then optimize the parameter $\boldsymbol{\theta}$ by minimizing a properly chosen cost function $f(\boldsymbol{\theta})$. As an example, the variation quantum eigensolvers (VQE) finds the smallest eigenvalue (ground-state energy) and its corresponding eigenvector (ground state) of a given Hamiltonian matrix $H$ by minimizing the energy of the state:

$$\theta^* = \operatorname{argmin}_{\boldsymbol{\theta}} f(\boldsymbol{\theta}) = \operatorname{argmin}_{\boldsymbol{\theta}} \langle U(\boldsymbol{\theta})\psi_0 | H | U(\boldsymbol{\theta})\psi_0 \rangle . \tag{6.1.2}$$

Here, $|\psi_0\rangle \in \mathbb{C}^{2^n}$ is a predetermined initial state that can be easily prepared on a quantum computer. For each given $\boldsymbol{\theta}$, $U(\boldsymbol{\theta})$ is implemented on a quantum computer to evolve $|\psi_0\rangle$, and the corresponding energy $f(\boldsymbol{\theta})$ and its gradient $\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta})$ can be consequently obtained with measurements. By solving the optimization problem equation 6.1.2, the minimum value gives an approximation to the smallest eigenvalue of $H$, while $U(\boldsymbol{\theta}^*)|\psi_0\rangle$ approximates the corresponding eigenvector.

## Problem setup

Although the problem of optimizing parameters in VQAs resembles classical optimization problems in machine learning, there exist key differences, particularly in how the cost function is evaluated and the level of accuracy that can be obtained for function and gradient evaluations. Firstly, quantum circuits used for estimating partial derivatives in various directions are typically different. This is predominantly because there is no straightforward method (in parallel to backpropagation) to estimate the entire gradient at once, given the inherent nature of quantum states. The predominant method for computing partial derivatives in a PQC is called the parameter-shift rule [75, 330, 20], which can only be applied to evaluate one component of the partial derivatives at a time. As a result, the estimation of the gradient, $\nabla f(\boldsymbol{\theta})$, typically incurs a cost that is $d$ times greater than the cost associated with merely estimating a single partial derivative, $\partial_i f(\boldsymbol{\theta})$.

Secondly, the evaluation of any given quantity, a function value or a partial derivative, requires measurement from quantum computers and is subject to measurement noise. We note that this noise is associated with a finite sampling space. For example, a measurement of the Hamiltonian in equation 6.1.2, which is defined in a finite-dimensional Hilbert space, yields one of its eigenvalues corresponding to the ansatz. Thus, with an increased number of samples or measurements, the central limit theorem suggests that the distribution of the sample average of the function value or the partial derivative can be approximated by a Gaussian distribution, and as a result, the accuracy of function and gradient evaluations can be relatively low. Therefore, the optimization algorithm must be designed to be resilient to measure noise.

In an idealized scenario, we may assume that both the function value and the partial derivatives incorporated into the optimization routine are subject to some Gaussian noise. But the magnitude of corresponding noises can differ up to a constant, especially in situations where the parameter shift rule is applicable (see [298]). With this consideration, the problem of optimizing PQCs can be stated as follows:

**Problem 1** (Optimizing parameterized quantum circuits)**.** *Finding an efficient*

*algorithm to solve the optimization problem,*

$$\boldsymbol{\theta}^* = \text{argmin}_{\boldsymbol{\theta}\in\mathbb{R}^d} f(\boldsymbol{\theta}), \tag{6.1.3}$$

*under the following assumptions:*

1. *The cost of evaluating a partial derivative scales linearly with that of a function value.*
2. *Every evaluation of the function and partial derivative is susceptible to Gaussian noise:*

$$f(\boldsymbol{\theta}) \to f(\boldsymbol{\theta}) + N(0, \sigma_1^2(\boldsymbol{\theta})), \quad \partial_i f(\boldsymbol{\theta}) \to \partial_i f(\boldsymbol{\theta}) + N(0, \sigma_2^2(\boldsymbol{\theta})). \tag{6.1.4}$$

Here, $\sigma_1(\boldsymbol{\theta})$ and $\sigma_2(\boldsymbol{\theta})$ depend on the real implementation and are not necessarily the same (see [298] for example). For simplicity, in our later analysis, we assume that $\sigma_2(\boldsymbol{\theta})$ has a uniform upper bound $\sigma_\infty$ (see assumption 2).

## Optimization methods

One widely used approach for optimizing VQA is through the application of gradient descent (GD) [299, 354]. The classical gradient descent method involves iteratively updating the parameters $\boldsymbol{\theta}$ by utilizing the gradient of the cost function.

$$\boldsymbol{\theta}_{n+1} = \boldsymbol{\theta}_n - a_n \nabla f(\boldsymbol{\theta}_n), \tag{6.1.5}$$

where $a_n$ denotes the learning rate. In light of the measurement process in quantum computing, we consider the noisy gradient descent: Rather than implementing eq. (6.1.5) with exact $\nabla f(\boldsymbol{\theta}_n)$, we apply an unbiased estimator $g(\boldsymbol{\theta})$ [1] (for example, equation 6.1.4). Consequently, the parameter update involves the following iteration,

$$\boldsymbol{\theta}_{n+1} = \boldsymbol{\theta}_n - a_n g(\boldsymbol{\theta}_n). \tag{6.1.6}$$

Since $g(\boldsymbol{\theta}_n)$ is an unbiased estimation, Eq. equation 6.1.6 is equivalent to Eq. equation 6.1.5 in the expectation sense. Specifically, by taking the conditional expectation on both sides, we have

$$\mathbb{E}(\boldsymbol{\theta}_{n+1}|\boldsymbol{\theta_n}) = \boldsymbol{\theta}_n - a_n \nabla f(\boldsymbol{\theta}_n) \tag{6.1.7}$$

where $\mathbb{E}(\cdot|\boldsymbol{\theta_n})$ denotes the conditional expectation given $\boldsymbol{\theta}_n$.

[1] $g(\boldsymbol{\theta})$ satisfies $\mathbb{E}[g(\boldsymbol{\theta})] = \nabla f(\boldsymbol{\theta})$

While noisy gradient descent avoids the need for precise gradient information, it still requires the approximated full gradient information at each iteration. As argued before, in the context of VQA, it is often necessary to compute $d$ partial derivatives separately for each direction, which makes the cost of each updating step at least $d$. In this paper, we introduce an alternative optimization method called random coordinate descent (RCD) [335, 230, 263] for addressing problem 1, with the goal of eliminating the cost dependency on $d$ in each step. RCD can be viewed as a variant of gradient descent (GD) where the full gradient in GD is approximated by a randomly selected component of $\nabla f(\boldsymbol{\theta}_n)$ in each iteration. Specifically, one RCD iteration can be expressed as:

$$\boldsymbol{\theta}_{n+1} = \boldsymbol{\theta}_n - a_n \boldsymbol{e}_{i_n} \partial_{i_n} f(\boldsymbol{\theta}_n)\,. \tag{6.1.8}$$

Here $\boldsymbol{e}_{i_n}$ is the $i_n$-th unit direction, $f'_{i_n}(\boldsymbol{\theta}_n)$ is the corresponding partial derivative of the cost function, and $i_n$ is a random index uniformly drawn from $\{1, 2, \cdots, d\}$. Similar to Eq. equation 6.1.6, we can write the noisy RCD as:

$$\boldsymbol{\theta}_{n+1} = \boldsymbol{\theta}_n - a_n \boldsymbol{e}_{i_n} g_{i_n}(\boldsymbol{\theta}_n)\,. \tag{6.1.9}$$

It is important to emphasize that in each iteration of RCD equation 6.1.9, only one partial derivative information is needed. Consequently, within the scope of VQA (as stated in the first assumption of problem 1), the cost per step of RCD is $d$ times smaller than that of GD.

## Contribution

This paper primarily focuses on the investigation of RCD in the context of noisy gradient evaluation. Our analysis is conducted in a specific comparison with GD, and we illustrate that, under specific conditions, RCD can serve as a favorable alternative for optimizing parameterized quantum circuits. The main contributions of this study can be summarized as follows:

- We show that RCD is theoretically no worse than GD when measuring the complexity by the number of partial derivative calculations (Theorems 3 and 4), assuming the presence of noise and the local PL condition. A summary of the complexities of the two methods is presented in Table 6.1 for comparison. It is important to highlight that the inequality $L_{\text{avg}} \leq L \leq dL_{\text{avg}}$ always holds. Consequently, when the optimization problem is highly anisotropic, i.e., $L \gg L_{\text{avg}}$, RCD is more cost-effective than GD. In the most extreme case when $L$ is nearly equal to $dL_{\text{avg}}$, RCD can reduce the complexity by a factor of $d$ compared to GD.

| Algorithm | Iteration cost | Iterations to reach $\epsilon$ tolerance | Total cost |
|---|---|---|---|
| GD | $\Omega(d)$ | $\tilde{\mathcal{O}}\left(\max\left\{\frac{L\sigma_\infty^2 d}{\mu^2\epsilon}, \frac{L}{\mu}\log\left(\frac{1}{\epsilon}\right)\right\}\right)$ | $\Omega(\frac{L\sigma_\infty^2 d^2}{\epsilon})$ |
| RCD | $\Omega(1)$ | $\tilde{\mathcal{O}}\left(\max\left\{\frac{L_{\text{avg}}\sigma_\infty^2 d^2}{\mu^2\epsilon}, \frac{dL_{\max}}{\mu}\log\left(\frac{1}{\epsilon}\right)\right\}\right)$ | $\Omega(\frac{L_{\text{avg}}\sigma_\infty^2 d^2}{\epsilon})$ |

Table 6.1: Comparison of the gradient descent and the randomized coordinate descent methods with an unbiased noisy gradient estimation. $d$ is the dimension of the parameter, and the smoothness constants $L$ and $L_{\text{avg}}$ are defined in equation 6.2.1 and equation 6.2.3, respectively. $\sigma_\infty^2$ is a bound for the measurement noise defined in equation 6.2.5a. In the table, we limit our attention to the situation where the learning rate is fixed.

- We demonstrate that (noisy) GD and RCD converge with high probability under the local PL condition (Assumption 3) and are stable under noisy gradient information. Specifically, if the initial parameter $\boldsymbol{\theta}_0$ resides within the basin $\mathcal{N}(\mathcal{X})$ surrounding the global minimum, both noisy methods ensure that the subsequent parameters $\boldsymbol{\theta}_n$ will remain consistently within this basin until they converge with the same high probability (Lemmas 5 and 6). To the best of the our knowledge, such stochastic stability has not been established for the optimization methods in variational quantum algorithms.
- We provide extensive empirical evidence demonstrating that RCD consistently delivers superior performance compared to GD (Sections 6.1 and 6.4). Our numerical findings support the theoretical observation that RCD can take a larger learning rate than GD, leading to faster convergence.

## Related works

**Gradient descent with noise** The noisy gradient descent equation 6.1.6 is a popular optimization method in the classical machine learning community. Notable examples are the stochastic gradient descent (SGD) [39] or the perturbed gradient descent (PGD) [148]. The convergence properties of the noisy gradient descent method in (6.1.6) have been extensively studied [225, 234, 255, 235, 298, 197]. For classical machine learning, these previous works except [298] established that when the cost function is $L$ smooth, $\mu$ strong convex (or Polyak-ojasiewicz condition (PL) [248]) and satisfies an additional condition, $f(\boldsymbol{\theta}_n)$ converges linearly to an approximation of $f_{\min}$. In the recent work [298], a similar theoretical result was shown for the noisy GD method applied to quantum optimization problems.

**Randomized coordinate descent** The randomized coordinate descent method (RCD) has proven its efficiency over GD in many large-scale optimization problems. The convergence properties of RCD have been extensively explored in the fields of machine learning and optimization [335, 230, 263, 232, 189, 69]. For example, it was shown in [230] that when $f$ is strongly convex, the convergence complexity of RCD can be consistently lower than or equal to that of GD. Here, complexity refers to the total number of partial derivative calculations required for convergence. Later, for strongly convex functions, RCD accelerations were achieved with adaptive momentum-based strategies in various regimes [189, 232]. For the non-convex optimization, recent work [69] shows the global convergence behavior of RCD with a focus on saddle point avoidance. Nevertheless, convergence rates of RCD have been scarcely studied for nonconvex optimization problems. More importantly, most related works focused on the case where partial derivatives are computed exactly, while in this work, we deal with the case where partial derivatives are estimated, which is subject to noise, and we will refer to it as *noisy* RCD equation 6.1.9.

**Locally-defined convex conditions for convergence analysis** One limitation of the conventional convergence analysis is its reliance on assumptions of global convex [39] or global PL [298] conditions for the cost function $f(\boldsymbol{\theta})$. However, we show that such global assumptions are not satisfied in quantum problem applications with PQCs, as elaborated in remark 2. Thus, one must weaken such a global assumption to a local one in the analysis. Convergence analysis under local assumptions requires more sophisticated techniques (see [104, 170, 243, 218] and therein), but it provides important insights that help to interpret empirical results. In our work, we make a local non-convex condition based on the local PL condition [197]. Under this condition and suitable assumptions for the cost function, we

By employing a stochastic stability argument, we demonstrate that the noisy GD and RCD methods maintain a comparable convergence rate under our local PL condition with high probability (refer to theorem 3 and theorem 4). To the best of the authors' knowledge, this paper is the first to provide a rigorous result for the complexity of noisy GD and RCD under a local PL condition designed for variational quantum algorithms built from PQCs.

**Other quantum optimization methods** Another promising direction of research in variational quantum algorithms is zero-order optimization, more commonly known as gradient-free methods. Notably, policy gradient-based techniques have shown their effectiveness in noise robust optimization in the NISQ [346]. Sung et al.[294] construct models based on the previous method and further improve the sample efficiency of the methods. Furthermore, these zero-order optimization methods leverage the

strengths of reinforcement learning [351, 106, 45, 49], Monte Carlo tree search [349, 217, 266], and natural evolutionary strategies [9, 360, 114], Bayesian [306, 301], as well as Gaussian processes [364].

In addition to these zero-order methods, several other optimization methods have been proposed recently [151, 260, 289, 113, 109]. One interesting example is the quantum natural gradient [289] (QNG), an approximate second-order method, that incorporates the quantum geometric tensor, which is similar to the natural gradient in classical machine learning. While an outcome of measurement is used as an estimate of the gradient in the QNG or the noisy gradient equation 6.1.6 from equation 6.1.1, the Jordan algorithm [151] encodes the partial derivatives as binary numbers in the computational basis. This algorithm was later improved by Gilyen et al. [113] using high-order finite difference approximations, and applications to VQAs for a certain class of smooth functions were considered. However, the methods [151, 113] require a significant number of ancilla qubits and complex control logics, due to the binary encoding of partial derivatives. Alternatively, [2] proposed a quantum backpropagation algorithm, which uses $\log d$ copies of the quantum state to compute $d$ derivatives. The overhead for computing $d$ derivatives is polylog($d$) times that of function evaluation (therefore mimicking backpropagation). One of the main drawbacks of their algorithm is that there is an exponential classical cost associated with the process. For a more restrictive class of cost functions (polynomial functions), [260] proposed a framework to implement the gradient descent and Newton's methods. This method also requires the coherent implementation of the cost function on a quantum computer using e.g., sparse input oracle, and thus can be challenging to implement in near-term devices.

## A numerical illustration: Variational quantum eigenvalue solver

As a brief illustration of the performance of noisy GD versus RCD methods, we consider the transverse-field Ising model,

$$H = J \sum_{j=1}^{N-1} Z_j Z_{j+1} + \Delta \sum_{j=1}^{N} X_j, \tag{6.1.10}$$

with the coefficient $J = 1$ and $\Delta = 1.5$. Here, $N$ denotes the number of qubits, and $X_j, Z_j$ are Pauli operators acting on the $j$-th qubit. In fig. 6.1, we set $N = 10$. To implement the quantum circuits, we use Qiskit Aer-simulator [252] with the command "result.get_counts" that outputs measurement outcomes as classical bitstrings. We

utilize the resulting classical bitstrings to compute partial derivatives by applying the parameter shift rule [298]. Thus, the result in fig. 6.1 takes into account the measurement noise.

In each experiment, 10 independent simulations are used with a fixed initialization. The parameterized quantum circuit used for estimating the ground state energy of the Hamiltonian equation 6.1.10 is given in Figure 6.10 (Section 6.9).

We compare the optimization performance of the two methods in terms of the number of partial derivative evaluations. The optimization results in fig. 6.1 suggest that RCD requires nearly 4 times fewer partial derivative evaluations than GD to converge to an energy ratio of 0.96 and a fidelity of 0.9, both of which are higher than the energy ratio and the fidelity obtained from GD. This observation can be explained by the analysis in section 6.2, i.e., RCD can be more efficient than GD when the ratio of Lipschitz constants ($L/L_{\text{avg}}$ or $L/L_{\max}$) is significantly larger than 1. Specifically, the ratio of the total computational cost of GD to RCD can be linked to the Lipschitz ratios, as summarized in table 6.1. For instance, in the lower panels of fig. 6.1, we observe that the ratio $L/L_{\text{avg}}$ and $L/L_{\max}$ remains above 30 and 8 throughout the iterations. The faster convergence of RCD can be attributed to these large Lipschitz ratios.

## 6.2 Preliminaries and main results

Before we establish results pertinent to the performance of RCD, we first establish consistent notations and assumptions, which are presented in Section 6.2. Following that, we outline our key theoretical findings in Section 6.2.

### Notations and assumptions

Given a vector $\boldsymbol{v} \in \mathbb{R}^d$, we use standard norms for $\boldsymbol{v}$, including the 2-norm $\|\boldsymbol{v}\|_2 := \sqrt{\sum_i v_i^2}$ and the $\infty$-norm $\|\boldsymbol{v}\|_\infty := \max_i |v_i|$. In order to ensure the convergence of gradient-based methods, we list several technical assumptions.

We assume the cost function $f$ satisfies the $L$-smoothness. Specifically, it satisfies the following assumption:

**Assumption 1.** *The cost function $f$ is $L$-smooth, in that,*

$$\|\nabla f(\boldsymbol{\theta}) - \nabla f(\boldsymbol{\theta}')\|_2 \leq L\|\boldsymbol{\theta} - \boldsymbol{\theta}'\|_2, \quad \textit{for all } \boldsymbol{\theta}, \boldsymbol{\theta}' \in \mathbb{R}^d. \tag{6.2.1}$$

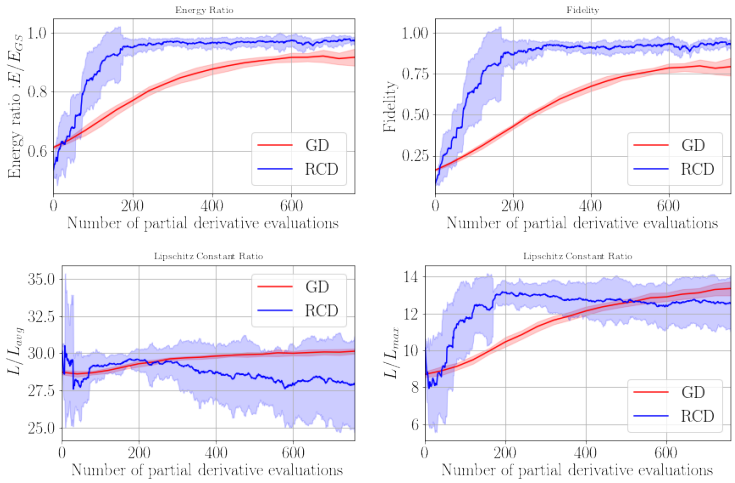



Figure 6.1: The comparison of the performance of GD (red) and RCD (blue) for optimizing the Hamiltonian equation 6.1.10. The unit of the $x$-axis labels the number of partial derivative evaluations as an indication of the computational complexity. The top panels show the approximation of the ground state, including the energy ratio (left) and fidelity (right). In the bottom panels, we show the ratios of Lipschitz constants obtained from the two methods are compared: $\frac{L}{L_{avg}}$ (left) and $\frac{L}{L_{\max}}$ (right).

Since the gradient is Lipschitz continuous, the partial derivatives are Lipschitz continuous as well. We define the componentwise Lipschitz constants,

**Definition 1.** *We say that a function $f$ is $L_i$-smooth with respect to the $i$-th component if*

$$|\partial_i f(\boldsymbol{\theta} + \boldsymbol{e}_i h) - \partial_i f(\boldsymbol{\theta})| \leq L_i |h| \quad \forall h \in \mathbb{R}, \tag{6.2.2}$$

*where $\partial_i f(\boldsymbol{\theta})$ denotes the partial derivative in the $i$-th direction.*

From these componentwise Lipschitz constants, we denote the maximum and average of those constants as

$$L_{\max} := \max_{1 \leq i \leq d} L_i, \quad L_{\text{avg}} = \frac{1}{d} \sum_{i=1}^{d} L_i \,. \tag{6.2.3}$$

As shown in [335], in general we have,

$$L_i \leq L_{avg} \leq L_{\max} \leq L \leq d L_{\max} \,. \tag{6.2.4}$$

Another interpretation is through the hessian: When $f$ is twice continuously differentiable, the condition equation 6.2.1 is equivalent to $\nabla^2 f(x) \preceq L I_d$, and similiarly, the condition equation 6.2.2 is equivalent to $\sup_{\boldsymbol{\theta}} |\partial_i^2 f(\boldsymbol{\theta})| \leq L_i$. We note that both the upper and lower bounds of $L$ in terms of $L_{\max}$ in equation 6.2.4 are tight. If $\nabla^2 f$ is a diagonal matrix, then $L_{\max} = L$, both being the largest diagonal element of $\nabla^2 f$. (This is the case in which all coordinates are independent of each other, for example, $f = \sum_i \lambda_i x_i^2$.) On the other hand, if $\nabla^2 f = \mathsf{e} \cdot \mathsf{e}^\top$ where $\mathsf{e} \in \mathbb{R}^d$ satisfies $\mathsf{e}_i = 1$ for all $i$, then $L = dL_{\max}$. This is a situation where $f$ is highly anisotropic, e.g., $f = (\sum_i x_i)^2/2$, where $L = d$ and $L_{\max} = 1$. In addition, when $L_{avg} = L$, we see that $L_{avg} = L_{\max} = L_i$ for all $i$.

Next, it is important to note that the estimation of the gradients in quantum computing can be susceptible to noise, which stems from the inherent nature of quantum measurements. Consequently, in our analysis and comparative studies of different optimization methods, we will take into account the presence of noise. To facilitate such analysis, we make the following assumption:

**Assumption 2** (Bounds of the noise with respect to the 2-norm)**.** *Given any $\boldsymbol{\theta} \in \mathbb{R}^d$, we assume that we can find an unbiased random estimate $\boldsymbol{g}(\boldsymbol{\theta})$ for the gradient $\nabla f(\boldsymbol{\theta})$, meaning that*

$$\mathbb{E}\left[\boldsymbol{g}(\boldsymbol{\theta})\right] = \nabla f(\boldsymbol{\theta})\,.$$

*Furthermore, we assume that there exists a constant $\sigma_\infty^2 > 0$ such that*

$$\sigma_\infty^2 > \sup_{\boldsymbol{\theta} \in \mathbb{R}^d} \max_{1 \leq i \leq d} \mathbb{E}\left[|\partial_i f(\boldsymbol{\theta}) - g_i(\boldsymbol{\theta})|^2\right]\,. \tag{6.2.5a}$$

*Here, we also assume $g(\boldsymbol{\theta})$ is independent for different $\boldsymbol{\theta}$.*

Additionally, we assume the existence of a basin encompassing the global minimum, within which $f$ satisfies the Polyak-Łojasiewicz condition (PL) [248], equivalently, the local PŁ condition [197].

**Assumption 3** (Local PL condition)**.** *Define $\mathcal{X}$ as the set of global minima and $f_{\min}$ as the global minimum value evaluated over $\mathcal{X}$. Then there exists a $\delta_f, \mu > 0$ such that for any $\boldsymbol{\theta} \in \mathcal{N}(\mathcal{X}) := f^{-1}([f_{\min}, \delta_f))$,*

$$\|\nabla f(\boldsymbol{\theta})\|^2 \geq 2\mu \left(f(\boldsymbol{\theta}) - f_{\min}\right).$$

It is worthwhile to highlight that the PL condition is defined not on the entire space $\mathbb{R}^d$ but $\mathcal{N}(\mathcal{X})$, which is reasonable in the context of the variational quantum algorithm. We support this argument with the following remark.

**Remark 2.** *Let $f(\boldsymbol{\theta})$ be a cost function defined by some parameterized quantum circuit equation 6.1.2. Note that $f$ is periodic and smooth, due to its specialized form. By the extreme value theorem, we see that there exist global maximum and minimum of $f$, denoted by $\boldsymbol{\theta}_{\max}$ and $\boldsymbol{\theta}_{\min}$. In general, $f$ is not constant, which means that $f_{\max} > f_{\min}$. Had $f$ satisfied the global PL condition, it would have followed that at the global maximum $\boldsymbol{\theta}_{\max}$,*

$$0 = \|\nabla f(\boldsymbol{\theta}_{\max})\|^2 \geq 2\mu \left(f_{\max} - f_{\min}\right) \geq 0, \tag{6.2.6}$$

*which gives a contradiction to the general case that $f_{\max} > f_{\min}$. As another case study, if $f$ is assumed to be convex, namely,*

$$f(\boldsymbol{\theta}') \geq f(\boldsymbol{\theta}) + (\nabla f(\boldsymbol{\theta}), \boldsymbol{\theta}' - \boldsymbol{\theta}) \text{ for all } \boldsymbol{\theta}, \boldsymbol{\theta}' \in \mathbb{R}^d, \tag{6.2.7}$$

*then setting $\boldsymbol{\theta} = \boldsymbol{\theta}_{\max}$ and $\boldsymbol{\theta}' = \boldsymbol{\theta}_{\min}$ results in a contradiction. Therefore, the cost function $f$ that is constructed from an ansatz similar to equation 6.1.2, will not satisfy global PL or convex conditions in general.*

## Main result: complexity comparison of GD and RCD

In this study, our main focus is to compare the complexity of noisy gradient descent (GD) and randomized coordinate descent (RCD) under the assumptions of a local Polyak-Łojasiewicz (PL) condition 3. For the sake of simplicity, in the remaining part of this paper, we will refer to "noisy gradient descent" and "noisy randomized coordinate descent" as "GD" and "RCD", respectively, without explicitly mentioning the term "noisy".

The main theoretical results are summarized in the following two theorems:

**Theorem 3** (Complexity of GD equation 7)**.** *Assume $f$ is a $L$-smooth function that satisfies assumption 3 and $g$ satisfies assumption 2. Given $\epsilon > 0$ small enough, if $f(\boldsymbol{\theta}_1) \leq \delta_f$ and $a_n = \Theta(\min\{\mu\epsilon/(L\sigma_\infty^2 d), 1/L\})$ in GD equation 7, then with probability $1 - f(\boldsymbol{\theta}_1)/\delta_f - o(1)$, there exists at least one*

$$n < N = \tilde{\Theta}(\max\left\{L\sigma_\infty^2 d/(\mu^2\epsilon), L/\mu\right\}) \tag{6.2.8}$$

*such that $f(\boldsymbol{\theta}_n) \leq f_{\min} + \epsilon$.*

**Theorem 4** (Complexity of RCD equation 8)**.** *Assume $f$ is a $L$-smooth function that satisfies assumption 3 and $g$ satisfies assumption 2. Given $\epsilon > 0$ small enough, if*

*$f(\boldsymbol{\theta}_1) \leq \delta_f$ and $a_n = \Theta(\max\{\mu\epsilon/(L_{\text{avg}}\sigma_\infty^2 d), 1/L_{\max}\})$ in RCD equation 8, then with probability $1 - f(\boldsymbol{\theta}_1)/\delta_f - o(1)$, there exists at least one*

$$n < N = \tilde{\Theta}(\max\left\{L_{\text{avg}}\sigma_\infty^2 d^2/(\mu^2\epsilon), L_{\max}d/\mu\right\}) \tag{6.2.9}$$

*such that $f(\boldsymbol{\theta}_n) \leq f_{\min} + \epsilon$.*

Based on the theorem mentioned above, to achieve $f(\boldsymbol{\theta}_n) - f_{\min} \leq \epsilon$, we can select the learning rate $a_n = \frac{\mu\epsilon}{L\sigma_\infty^2 d}$ for GD and $a_n = \frac{\mu\epsilon}{L_{\text{avg}}\sigma_\infty^2 d}$ for RCD. Recalling equation equation 6.2.4, we observe that $L_{\text{avg}} \leq L$, which means that we could use a larger learning rate for RCD. This choice aligns with the learning rates utilized in the numerical experiments presented in Section 6.1 as well as those in Section 6.4.

We compare the complexity of the noisy GD and RCD methods with the estimates of the number of iterations. First, according to the above result, we conclude that the number of iterations required for GD is $N = \tilde{\Theta}\left(\frac{L\sigma_\infty^2 d}{\mu^2\epsilon}\right)^2$, while for RCD, we have $N = \tilde{\mathcal{O}}\left(\frac{L_{\text{avg}}\sigma_\infty^2 d^2}{\mu^2\epsilon}\right)$. Notably, in RCD, there is an additional factor of $d$, which can be understood in the expectation sense: During each iteration of the noisy RCD, the randomness arises from two sources: the random direction $i_n$ and the noisy partial derivative $g_{i_n}(\boldsymbol{\theta}_n)$. By taking the conditional expectation with respect to $\boldsymbol{\theta}_n$, we obtain:

$$\mathbb{E}(\boldsymbol{\theta}_{n+1}|\boldsymbol{\theta_n}) = \boldsymbol{\theta}_n - \frac{a_n}{d}\nabla f(\boldsymbol{\theta}_n)\,. \tag{6.2.10}$$

Compared with equation 6.1.7, there is an extra $1/d$ factor in the expectation of RCD. Consequently, in each iteration, the rate of decay of the cost function is smaller in RCD compared to GD. Consequently, we anticipate that RCD would necessitate more iteration steps to achieve convergence. On the other hand, it is also important to note that in certain scenarios where $L_{\text{avg}}d$ is comparable to $L$, the number of iterations required for RCD is comparable to that of GD.

Meanwhile, it is important to point out that a more practical criterion for comparing the two methods is the cumulative cost of each method, which is represented by the number of partial derivative calculations from the quantum circuits. This is because quantum algorithms for estimating the gradient have a cost proportional to $d$. Since each iteration of GD needs to calculate the full gradient ($d$ partial derivatives), the total number of partial derivative estimations in GD is

$$N_{\text{partial,GD}} = \tilde{\Theta}\left(\frac{L\sigma_\infty^2 d^2}{\mu^2\epsilon}\right)\,.$$

[2]This complexity aligns with the classical theoretical results for gradient descent (GD), which typically assume the presence of strong convexity or a local PL condition for the function $f$.

In contrast, the number of partial derivative estimations in RCD is:

$$N_{\text{partial,RCD}} = \tilde{\mathcal{O}}\left(\frac{L_{\text{avg}}\sigma_\infty^2 d^2}{\mu^2\epsilon}\right).$$

From equation equation 6.2.4, we can deduce that:

$$\tilde{\Omega}(N_{\text{partial,RCD}}) = N_{\text{partial,GD}} = d\tilde{\mathcal{O}}(N_{\text{partial,RCD}}).$$

This suggests that the computational cost of RCD is $L/L_{\text{avg}}$ times cheaper than that of GD. In an extreme case where $f$ is highly skewed, i.e., $L/L_{\text{avg}} \approx d$, RCD can reduce the computational cost by a factor of the dimension $d$, which will be a significant reduction for large quantum circuits.

In addition to the complexity result, it is worth noting that the two methods exhibit similar success probability, which is approximately $1 - f(\boldsymbol{\theta}_1)/\delta_f$, as indicated by the two aforementioned theorems. This observation is quite surprising, as each iteration of RCD appears noisier due to the random selection of the updating direction $i_n$. Intuitively, this suggests that we might need to choose a smaller learning rate $a_n$ to ensure stability in RCD, which would consequently increase its complexity. However, our theory unveils that choosing a similar learning rate $a_n$ is adequate to stabilize RCD. To elucidate this point, it's important to recognize that, on average, RCD behaves equivalently to GD. By conducting more iterations, RCD can approximate its average behavior (expectation), effectively mitigating the extra randomness introduced by $i_n$. This compensation mechanism ensures that the success probabilities remain consistent between the two methods.

## 6.3 Proof of main results

In this section, we provide the proofs for Theorems 3 and 4. We will start by showing the stochastic stability of the two methods in Section 6.3. This will guarantee that the parameter $\boldsymbol{\theta}$ is likely to stay close to the global minimum until attaining a small loss. Following that, in Section 6.3, we utilize the local Polyak-Łojasiewicz (PL) condition around the global minimum to establish the convergence of $f(\boldsymbol{\theta}_n)$. In all of the following theoretical results and the corresponding proofs in the appendices, we assume $f_{\min} = 0$ without loss of generality by modifying the original function as

$$f(\boldsymbol{\theta}) \leftarrow f(\boldsymbol{\theta}) - f_{\min}. \tag{6.3.1}$$

Thus, all results in this section can be reformulated for the original cost function by the substitution equation 6.3.1, which will yield theorems 3 and 4.

## Stochastic stability

In the context of optimization, stability and convergence are not separate properties. In a deterministic algorithm, convergence immediately guarantees stability. However, this connection does not hold for stochastic processes in general. For instance, when optimization methods such as noisy GD, SGD, or noisy RCD are applied, discrete-time stochastic processes are generated. In such cases, a convergence theory must be developed for a collection of random paths, which can exhibit different convergence behaviors among themselves.

In our specific case, we anticipate that when $\boldsymbol{\theta}_n$ remains within the basin $\mathcal{N}(\mathcal{X})$ and the learning rate is correctly chosen, both the GD and the RCD methods, when the gradient is exactly calculated, converge to a global minimum due to the local PL condition stated in assumption 3. However, in the presence of noise in the gradient and the use of a constant learning rate, it is generally impossible to ensure that $\boldsymbol{\theta}_n \in \mathcal{N}(\mathcal{X})$ almost surely, unless a different strategy is adopted such as the decreasing learning rates [243, 104, 170]. On the other hand, the purpose of the optimization algorithm is to minimize the loss function, which means that it suffices to ensure stability until small loss is achieved. To quantify such a likelihood, in this section, we demonstrate that when $\boldsymbol{\theta}_0 \in \mathcal{N}(\mathcal{X})$, there exists a finite probability that $\boldsymbol{\theta}_n$ obtained from GD and RCD remain within $\mathcal{N}(\mathcal{X})$ until achieving a small loss. This provides a high probability of convergence for the two methods.

We summarize the result for noisy GD in the following lemma.

**Lemma 5.** *Assume that $f$ is a $L$ smooth function that satisfies the assumption 3 and $g$ satisfies the assumption 2. If $f(\boldsymbol{\theta}_1) \leq \delta_f$ and the learning rate is chosen as follows,*

$$a_n = a < \min\left\{\frac{1}{L}\,, \frac{2\mu\delta_f}{L\sigma_\infty^2 d}\right\},$$

*then, with high probability, iterations of noisy GD equation 7 remain in $f^{-1}([0, \delta_f))$ until a small loss is achieved. Specifically,*

$$\mathbb{P}\left\{\exists N > 0 \text{ such that } f(\boldsymbol{\theta}_N) \notin \mathcal{N} \text{ and } f(\boldsymbol{\theta}_n) > \frac{La\sigma_\infty^2 d}{\mu},\ \forall n < N\right\} \leq \frac{f(\boldsymbol{\theta}_1)}{\delta_f}. \quad (6.3.2)$$

In light of equation equation 6.3.2, if we select the learning rate $a_n$ to be sufficiently small, then with a probability of $1 - \frac{f(\boldsymbol{\theta}_1)}{\delta_f}$, the parameters are guaranteed to achieve a small loss before escaping the basin.

Despite infrequent updates of the gradient components, RCD still demonstrates a similar level of stochastic stability. This key observation is summarized in the following lemma:

**Lemma 6.** *Assume that $f$ is a $L$-smooth function that satisfies assumption 3 and $g$ satisfies assumption 2. Given any $f(\boldsymbol{\theta}_1) < \delta_f$, if one chooses the learning rate*

$$a_n = a < \min\left\{\frac{1}{L_{\max}},\ \frac{d}{\mu},\ \frac{2\mu\delta_f}{L_{\text{avg}}\sigma_\infty^2 d}\right\},$$

*then, with high probability, iterations from the noisy RCD equation 8 stay at $f^{-1}([0,\delta_f))$ until achieving a small loss. Specifically,*

$$\mathbb{P}\left\{\exists N > 0 \text{ such that } f(\boldsymbol{\theta}_N) \notin \mathcal{N} \text{ and } f(\boldsymbol{\theta}_n) > \frac{L_{\text{avg}} a\sigma_\infty^2 d}{\mu},\ \forall n < N\right\} \leq \frac{f(\boldsymbol{\theta}_1)}{\delta_f}.$$

The proofs of Lemma 5 and 6 are provided in Sections 6.6 and 6.7, respectively. The core concept of these proofs is based on the construction of a specialized supermartingale and the utilization of Markov's inequality. For example, to prove Lemma 5, we define a stochastic process

$$V_n = \begin{cases} f(\boldsymbol{\theta}_n)\mathbb{I}_n, & n < \tau \\ f(\boldsymbol{\theta}_\tau)\mathbb{I}_\tau, & n \geq \tau \end{cases}.$$

where the indicator random variable is given by,

$$I_n = \begin{cases} 1, & \text{if} \quad \{\boldsymbol{\theta}_k\}_{k=1}^{n-1} \subset f^{-1}([0,\delta_f)) \\ 0, & \text{otherwise.} \end{cases},$$

and the stopping time

$$\tau = \inf\left\{k : f(\boldsymbol{\theta}_k) \leq \frac{La\sigma_\infty^2 d}{\mu}\right\}.$$

We observe that $V_n$ is a meticulously crafted supermartingale, allowing us to distinguish between stable and unstable events. In particular, we demonstrate that if $\boldsymbol{\theta}_n$ exits the basin before it reaches $f(\boldsymbol{\theta}_n) = \frac{La\sigma_\infty^2 d}{\mu}$ (an unstable event), then $\sup_n V_n \geq \delta_f$. Therefore, we can employ $V_n$ as a categorizer and the probability of failure of GD can be characterized by the value of $V_n$. More specifically,

$$\mathbb{P}\left\{\exists N > 0 \text{ such that } f(\boldsymbol{\theta}_N) \notin \mathcal{N} \text{ and } f(\boldsymbol{\theta}_n) > \frac{La\sigma_\infty^2 d}{\mu},\ \forall n < N\right\} \leq \mathbb{P}\left\{\sup_n V_n \geq \delta_f\right\}.$$

Except for its use as a categorizer, we have designed $V_n$ in such a way that it is a supermartingale, meaning $\mathbb{E}(V_{n+1}|\boldsymbol{\theta}_{k\leq n}) \leq V_n$. Therefore, we can use Markov's inequality for supermartingales to bound the supremum of $V_n$ and achieve the desired result.

## Convergence analysis

In this section, we present the convergence properties of noisy GD and RCD methods. It is important to note that Theorems 3 and 4 directly follow from Theorems 7 and 8, respectively.

Our first theorem shows the convergence performance of the noisy GD method,

**Theorem 7.** *Assume $f$ is a $L$-smooth function that satisfies Assumption 3 and $g$ satisfies assumption 2. Given any precision $0 < \epsilon < \delta_f$, the initial guess $f(\boldsymbol{\theta}_1) < \delta_f$, and the probability of failure $\eta \in \left(\frac{f(\boldsymbol{\theta}_1)}{\delta_f^{-1}}, 1\right)$, we choose the learning rate in equation 6.1.5*

$$a_n = a = \mathcal{O}\left(\min\left\{\frac{1}{L}, \frac{\mu\epsilon}{L\sigma_\infty^2 d}\right\}\right),$$

*and the total number of iterations*

$$N = \Omega\left(\frac{1}{\mu a \eta}\log\left(\frac{f(\boldsymbol{\theta_1})}{\left(\eta - \frac{f(\boldsymbol{\theta}_1)}{\delta_f}\right)\epsilon}\right)\right).$$

*Then, with probability $1-\eta$, we can find at least one $\boldsymbol{\theta}_m$ with $1 \leq m \leq N$ such that $f(\boldsymbol{\theta}_m) \leq \epsilon$. In particular,*

$$\mathbb{P}\left\{\exists m \leq N,\ f(\boldsymbol{\theta}_m) \leq \epsilon\right\} \geq 1 - \eta,$$

Next, we state the convergence property of the noisy RCD method in the following theorem,

**Theorem 8.** *Assume $f$ is a $L$-smooth function that satisfies Assumption 3 and $g$ satisfies Assumption 2. Given any precision $0 < \epsilon < \delta_f$, the initial guess $f(\boldsymbol{\theta}_1) < \delta_f$, and the probability of failure $\eta \in \left(\frac{f(\boldsymbol{\theta}_1)}{\delta_f^{-1}}, 1\right)$, we choose the learning rate in equation 6.1.9*

$$a_n = a = \mathcal{O}\left(\min\left\{\frac{1}{L_{\max}}, \frac{d}{\mu}, \frac{\mu\epsilon}{L_{\text{avg}}\sigma_\infty^2 d}\right\}\right),$$

*and the total number of iterations*

$$N = \Omega\left(\frac{d}{\mu a \eta}\log\left(\frac{f(\boldsymbol{\theta_1})}{\left(\eta - \frac{f(\boldsymbol{\theta}_1)}{\delta_f}\right)\epsilon}\right)\right).$$

*Then, with probability* $1-\eta$*, we can find at least one* $\boldsymbol{\theta}_m$ *with* $1 \leq m \leq N$ *such that* $f(\boldsymbol{\theta}_m) \leq \epsilon$*. In particular,*

$$\mathbb{P}\left\{\exists m \leq N,\ f(\boldsymbol{\theta}_m) \leq \epsilon\right\} \geq 1-\eta,$$

The proofs of these theorems can be found in the section 6.8.

**Remark 9.** *We emphasize that theorem 7 and theorem 8 are general convergence results that require only mild conditions. Specifically, theorem 7 can be used to demonstrate the stability and convergence of the traditional SGD algorithm when the right assumptions are in place. A convergence result analogous to the one previously discussed has been investigated in [197, Theorem 7], where the authors impose a more stringent requirement on the cost function [23]. In our work, we demonstrate the convergence of noisy GD using more sophisticated techniques in probability theory and adopt a weak version of probabilistic convergence [169]. In addition, our approach can be directly extended to show the convergence of noisy RCD as in theorem 8, which to the best of our knowledge, has not been established before. These two theorems suggest that with a high probability, the loss function can achieve small loss during the training process. In other words, it is likely that the parameter* $\boldsymbol{\theta}$ *remains in the basin* $\mathcal{N}$ *until the precision* $\epsilon$ *is attained at some point. After that, the optimization algorithm could diverge unless a certain strategy is applied, for example, a schedule of decreasing learning rates or an early stopping criterion.*

**Remark 10.** *Our theoretical result clarifies a relation between the learning rate and the desired precision in optimization. For example, the precision* $\epsilon$ *is manifested in the upper bounds of the learning rates in theorem 7 and theorem 8. Thus, to reach precision* $\epsilon$*, it is suggested to use an* $\mathcal{O}(\epsilon)$ *learning rate. Otherwise, due to the stability issue, the trajectory is no longer guaranteed to converge to the precision with positive probability.*

We present the roadmap for proving Theorem 7 as follows: Define the stopping time

$$\tau = \inf\left\{k : f(\boldsymbol{\theta}_k) \leq \epsilon\right\}.$$

To prove Theorem 7, it suffices to demonstrate that the probability of failure $\mathbb{P}(\tau > N)$ is small. Since the learning rate $a_n$ is selected to be sufficiently small and, according

to the lemma 5, it is likely that $\boldsymbol{\theta}_n$ will remain within the basin until the $\epsilon$ loss is achieved[3]. Thus, informally, it suffices for us to assume $\boldsymbol{\theta}_n \in \mathcal{N}$. The next step is to find an upper bound for the probability of failure $p_{\text{fail}} = \mathbb{P}(\tau > N)$. Using the local PL condition, we can show that when $\epsilon < f(\boldsymbol{\theta}_n) < \delta_f$,

$$\mathbb{E}(f(\boldsymbol{\theta}_{n+1})|\boldsymbol{\theta}_n) \leq \left(1 - \frac{\mu a}{2}\right) f(\boldsymbol{\theta}_n)\,,$$

meaning that the conditional expectation of $f(\boldsymbol{\theta}_{n+1})$ decays to zero with rate $\left(1 - \frac{\mu a}{2}\right)$. Inspired by this observation, we can construct a supermartingale to show that, if $\tau > N$, with high probability, we have $\inf_{1 \leq n \leq N} f(\boldsymbol{\theta}_n) \leq \epsilon$. We note that this event is complementary to the failure event $\{\tau > N\}$. Consequently, we obtain an upper bound for $p_{\text{fail}}$.

## 6.4 Numerical results

In section 6.1, depicted in fig. 6.1, we have demonstrated that the noisy RCD leads to faster convergence than the noisy GD for VQE problems. In this section, we extend our investigation to gauge the efficiency of noisy RCD applied to various other variational quantum algorithms, especially those involving non-convex optimization problems. The implementation of these algorithms is executed on classical computers. To emulate quantum measurement noise, the partial derivatives undergo perturbation through additive Gaussian noise as outlined in section 6.1 [4]. Subsequently, we substantiate this approximation through a numerical experiment on a quantum simulator. This experiment further also proposes suitable values for the strength of the Gaussian noise that we will introduce in the upcoming numerical tests to appropriately mimic the measurement noise.

In the experiment presented in section 6.4, we utilize Qiskit-0.44.1 [252].

The algorithms for subsequent examples are implemented using Numpy [134] and Jax [40]. We conducted each experiment ten times, employing different random initializations for each run. All tests are executed on an Intel(R) Xeon(R) CPU @ 2.20GHz, complemented by a T4 GPU.

[3]Rigorously, we must also take into account the possibility that the optimization algorithm does not reach $\epsilon$ loss in a finite number of iterations.

[4]The derivative with noise is computed by adding Gaussian noise to the original derivative: $\partial_i f(x) \leftarrow \partial_i f(x) + \epsilon$, where $\epsilon$ follows a Gaussian distribution, denoted as $\mathcal{N}(0, \sigma)$. In this notation, $\sigma$ signifies the standard deviation, defining the intensity of the Gaussian noise.

## Analyzing the noise distribution

Building on the numerical experiment detailed in section 6.1 and executed in Qiskit, we analyze the statistics of the partial derivatives derived from the quantum circuit. fig. 6.2 showcases the histograms representing 10,000 estimates of partial derivatives with respect to the initial 12 directions, while the histograms for the remaining directions are presented in section 6.11. Each estimate of the partial derivatives is averaged over 1000 shots. From all histograms, we can clearly see that the distribution is closely approximated by a Gaussian distribution. In addition, the magnitude of the standard deviation of partial derivative estimates across all directions is comparable. These observations support assumptions of the noise model in problem 1. For simplicity, we will employ the Gaussian noise model in our subsequent investigations to compare the performance of the noisy GD and RCD methods.

In the next two sections, we conduct a comprehensive comparison between noisy RCD and GD across a broad spectrum of variational quantum algorithms and applications.

## VQE with a varied circuit structure

In section 6.1, we utilize the VQE for the TFIM equation 6.1.10 employing both the noisy GD and the noisy RCD. In this section, we tackle the same optimization task but with a modified setup. Specifically, fig. 6.3 depicts the PQC [156] utilized in the experiments showcased in fig. 6.4, distinct from that presented in fig. 6.10.

In the experiments illustrated in fig. 6.4, each optimization outcome derives from 10 identical simulations with the same initial condition. We set the learning rates for the RCD and GD at 0.3 and 0.05, respectively. Each experiment utilizes 10,000 shots, with 18 trainable parameters. Results shown in fig. 6.4 demonstrate that, compared to GD, RCD requires nearly three times fewer partial derivative evaluations to converge.

## Quantum Approximate Optimization Algorithm (QAOA) for quantum Hamiltonians

The Quantum Approximate Optimization Algorithm (QAOA) [102], originally devised for solving combinatorial problems, is a leading example for demonstrating quantum advantage on near-term quantum computers. As introduced in [102], the QAOA utilizes a parametrized quantum circuit (PQC), which naturally enables optimization through the variational quantum algorithm.

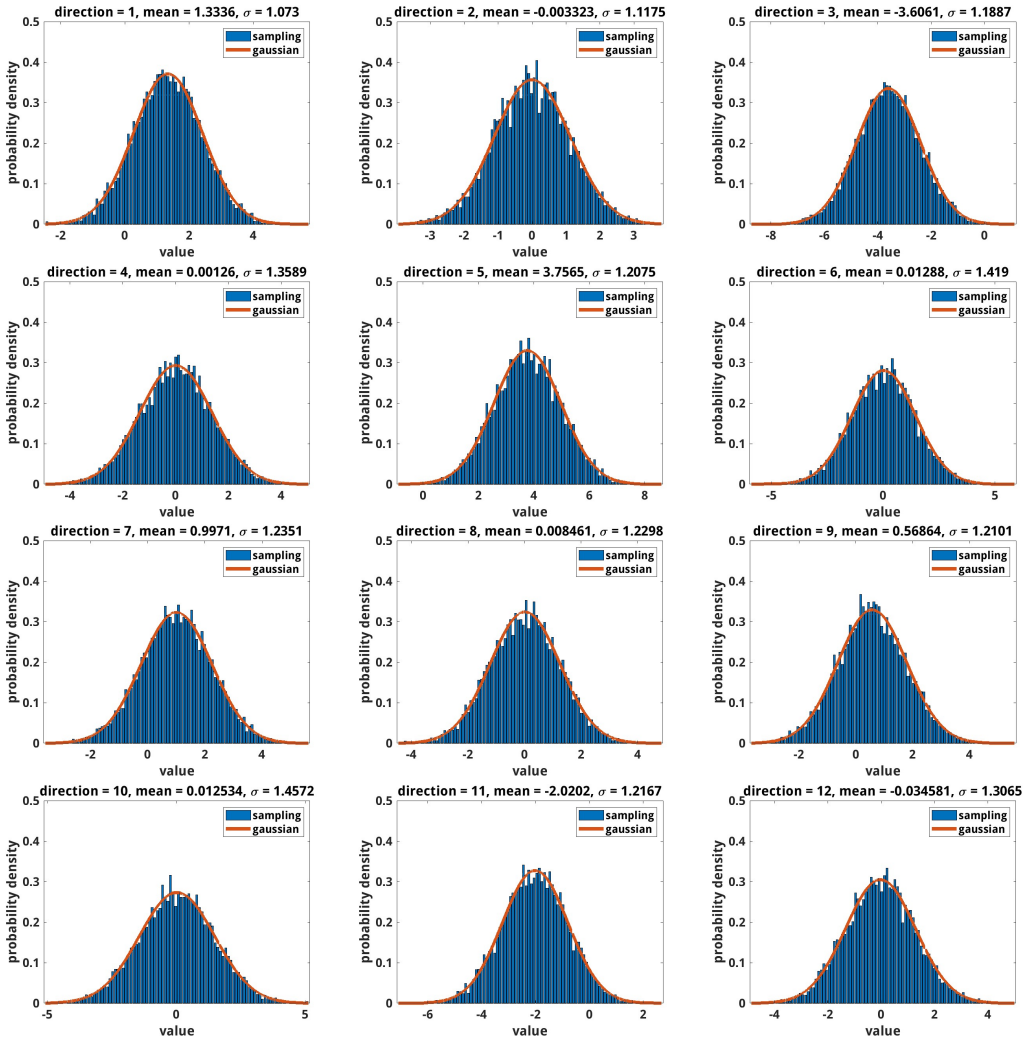



Figure 6.2: Histograms of the estimated partial derivatives: Each panel displays the histogram of 10000 partial derivative estimates in one of the first 12 directions, which are obtained by applying the parameter-shift rule for the ansatz in fig. 6.10. The sampling of the partial derivatives is carried out at a suboptimal point chosen from one simulation used in fig. 6.1, where the fidelity is about 0.889.

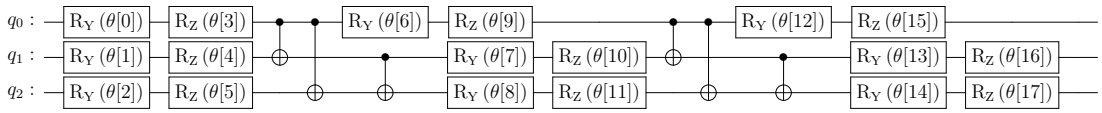



Figure 6.3: A variational circuit ansatz is employed for the Transverse-Field Ising Model expressed in Equation equation 6.1.10, utilizing 3 qubits. This circuit is a parameterized construct comprised of alternating rotation and entanglement layers. Each rotation layer involves the application of single qubit gates, specifically Rotation-y and Rotation-z gates, to all qubits. In contrast, the entanglement layer employs two-qubit gates, namely the controlled-X gate, to facilitate entanglement among the qubits. The ansatz is designated with 18 parameters.

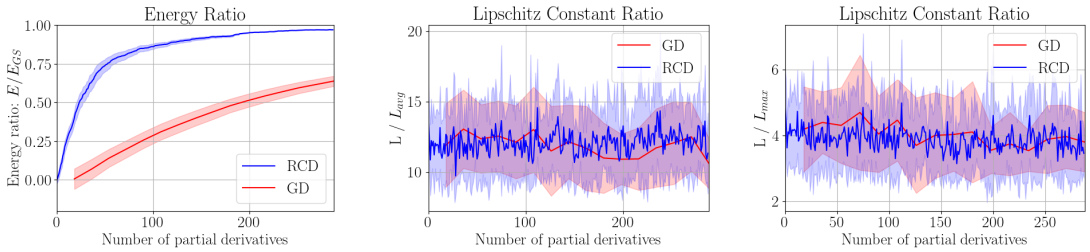



Figure 6.4: Performance comparison between GD (red) and RCD (blue) in terms of energy ratio and Lipschitz constant ratios for optimizing the Hamiltonian equation 6.1.10. The energy ratio $E/E_{GS}$ is presented in the left panel, while the Lipschitz constant ratios, denoted as $\frac{L}{L_{\text{avg}}}$ and $\frac{L}{L_{\text{max}}}$, are shown in the middle and right panels respectively. The shaded areas in each panel represent variations observed across multiple trials.

In a generalized QAOA model, we begin with an initial quantum state $|\psi_i\rangle$, which can be easily prepared in experiments, and let it evolve by a parameterized unitary transformation,

$$|\psi(\boldsymbol{\alpha},\boldsymbol{\beta})\rangle = U(\{\alpha_j,\beta_j\}_{j=1}^{p})\,|\psi_i\rangle = e^{-iH_2\beta_p}e^{-iH_1\alpha_p}\cdots e^{-iH_2\beta_1}e^{-iH_1\alpha_1}\,|\psi_i\rangle\,, \qquad (6.4.1)$$

where the vector $\boldsymbol{\alpha}$ (or $\boldsymbol{\beta}$) enumerates the parameters $\alpha_j$ (or $\beta_j$), and thus the total number of parameters is $2p$ and the unitary transformation alternates between two kinds of parameterized unitary transformations. With this ansatz, the optimization is performed with the parameters $\{\alpha_j,\beta_j\}$ associated with the application-dependent Hamiltonian matrices $H_1$ and $H_2$, respectively.

In the subsequent sections, we will consider optimization problems based on the

QAOA equation 6.4.1. We will conduct a comparative analysis of the noisy GD and RCD for various QAOA models that will span a range of systems, including the Ising model (refer to Section 6.4), the Heisenberg model (refer to Section 6.4), and Variational Quantum Factoring (refer to Section 6.4).

### QAOA – Ising Model

In this section, we parameterize the transverse-field Ising model by a Hamiltonian

$$H[h] = \sum_{j=1}^{N-1} Z_{j+1}Z_j + \sum_{j=1}^{N}(Z_j + hX_j), \tag{6.4.2}$$

where $N$ denotes the total number of qubits. The global control field $h \in \{\pm 4\}$ takes two discrete values, corresponding to the two alternating QAOA generators $H_1 = H[-4]$ and $H_2 = H[+4]$ [48, 346]. The initial state $|\psi_i\rangle$ corresponds to the ground state of $H[-2]$, while the desired target state $|\psi_*\rangle$ is selected as the ground state of $H[+2]$. The variational problem aims to optimize the fidelity [5],

$$\max_{\{\alpha_i,\beta_i\}_{i=1}^p} \mathcal{F}(\{\alpha_i,\beta_i\}_{i=1}^p) = \max_{\{\alpha_i,\beta_i\}_{i=1}^p} |\langle\psi_*|U(\{\alpha_i,\beta_i\}_{i=1}^p)|\psi_i\rangle|^2, \tag{6.4.3}$$

where,

$$U(\{\alpha_i,\beta_i\}_{i=1}^p)\,|\psi_i\rangle = e^{-iH_2\beta_p}e^{-iH_1\alpha_p}\cdots e^{-iH_2\beta_1}e^{-iH_1\alpha_1}\,|\psi_i\rangle\,. \tag{6.4.4}$$

We note that the fidelity optimization equation 6.4.3 is equivalent to the optimization of the form equation 6.1.2 by letting the Hamiltonian be $|\psi_*\rangle\langle\psi_*|$.

In the numerical test, we choose a system from equation 6.4.2 with three qubits ($N = 3$), and then apply both GD and RCD methods in the optimization. Figure 6.5 shows the optimization results obtained from the noisy GD and RCD with the respective learning rates of 0.0045 and 0.015 by using an ansatz defined with 20 parameters. By adjusting the learning rate and tracking the stability, We observe that RCD permits a larger learning rate in comparison to GD, while maintaining the stability. Similar to the results presented in fig. 6.1, we compare the performance of the two methods in terms of the number of partial derivative evaluations. From fig. 6.5, We observe that noisy RCD converges much faster than noisy GD. While RCD achieves a fidelity near 1 with 500 partial derivative evaluations, GD only attains

[5] Fidelity serves as a metric for optimization. However, one caveat of utilizing fidelity is its reliance on the ground state. In this context, we assume the presence of an oracle capable of producing the fidelity value. Subsequently, we also employ energy as an observable metric for optimization purposes.

a fidelity below 0.25 with an equivalent number of evaluations. This computational effectiveness of RCD can be attributed to the large ratios of Lipschitz constants shown in fig. 6.5, which are obtained along the optimization trajectories.

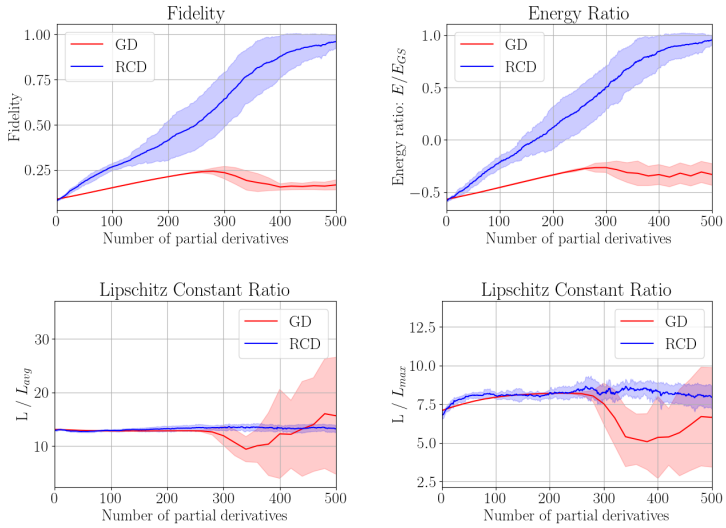



Figure 6.5: Performance comparison between the noisy GD and RCD for the Ising model equation 6.4.2. The corresponding Lipschitz constant ratios, denoted as $\frac{L}{L_{\text{avg}}}$ and $\frac{L}{L_{\max}}$, are presented in the bottom figures. The shaded areas within the figures represent variations that have been observed across ten random realizations. The optimization is performed for parameters with dimension equal to 20.

### QAOA – Heisenberg Model

Our second test problem with QAOA is the (anisotropic) spin-1 Heisenberg model, $H = H_1 + H_2$, with the alternating Hamiltonians given by,

$$H_1 = J \sum_{j=1}^{N} (X_{j+1}X_j + Y_{j+1}Y_j), \quad H_2 = \Delta \sum_{j=1}^{N} Z_{j+1}Z_j,$$

with anisotropic parameter $\Delta/J = 0.5$ (topological/Haldane [68, 247, 185, 348]). For the Heisenberg model, we consider a system consisting of eight qubits ($N = 8$) and choose the fidelity as a measure for optimization, similar to the setup for the results in fig. 6.5. We set the antiferromagnetic initial state to $|\psi_i\rangle = |10101010\rangle$. The target state is the ground state of the Hamiltonian $H = H_1 + H_2$. We employ the QAOA

ansatz represented by Eqn. equation 6.4.4 and carry out the fidelity optimization detailed in Eqn. equation 6.4.3.

Figure 6.6 showcases the performance outcomes from noisy GD and RCD simulations with learning rates set to 0.01 and 0.1, respectively. This QAOA model involves 28 parameters. The fidelity result shows that RCD converges to the target state much faster than GD. This phenomenon can be elucidated by noting that the ratios of Lipschitz constants derived from both noisy methods, $\frac{L}{L_{\text{avg}}}$ and $\frac{L}{L_{\text{max}}}$, average around 10 and 6 along the trajectories, respectively. Especially, the magnitude of the ratio $\frac{L}{L_{\text{max}}}$ is similar to that of the ratio of the numbers of partial derivative evaluations to reach a high fidelity $> 0.8$ from both noisy methods, as shown in fig. 6.6. Based on the observed numerical results, a high ratio of $\frac{L}{L_{\text{max}}}$ is responsible for the efficiency of RCD in this optimization problem.

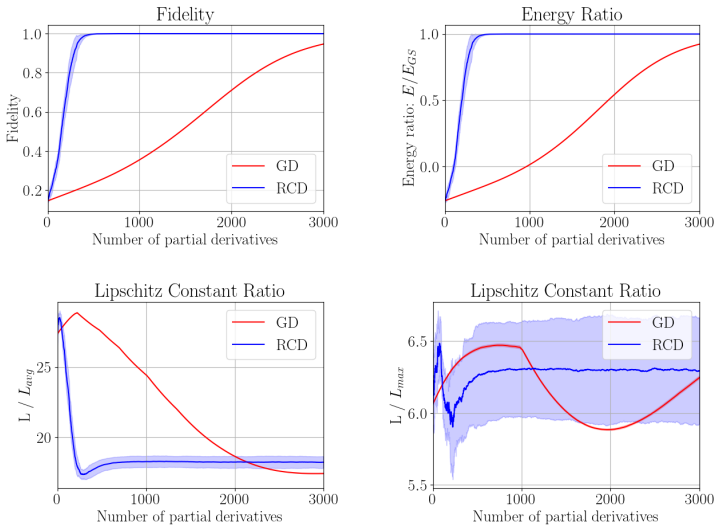



Figure 6.6: Performance comparison between noisy GD and RCD for the Heisenberg model. The corresponding Lipschitz constant ratios, denoted as $\frac{L}{L_{\text{avg}}}$ and $\frac{L}{L_{\text{max}}}$, are presented in the middle and right. The shaded areas within the figure represent variations that have been observed across ten random realizations. The optimization is performed in dimensions of 28.

## QAOA for classical combinatorial optimization problems

Quadratic Unconstrained Binary Optimization (QUBO) problems have significant applications in fields such as finance, logistics, and machine learning, etc. Recognized

as one prominent optimization model in quantum computing, QUBO consolidates a wide range of combinatorial optimization problems [94, 171, 14, 115] and translates them into identifying the ground state of classical Ising models [203].

The goal of QUBO is to identify a sequence of binary variables (0 or 1) that minimize a quadratic function. Specifically, a cost function $f_Q$ is constructed over the set of binary vectors, $\mathbb{B}^n$:

$$f_Q(x) = x^\top Q x = \sum_{i,j=1}^{n} Q_{ij} x_i x_j. \tag{6.4.5}$$

In this context, $\mathbb{B} = \{0, 1\}$ signifies the set of binary values (or bits), and $\mathbb{B}^n$ represents the collection of binary vectors with length $n > 0$. A symmetric, real-valued matrix $Q \in \mathbb{R}^{n\times n}$ is introduced, with each element $Q_{ij}$ determining the weight for the corresponding pair of indices $i, j \in 1, \ldots, n$. For example, if $i = j$, the term $Q_{ii}x_i^2$ contributes $Q_{ii}$ to the function value when $x_i = 1$. On the other hand, if $i \neq j$, the term $Q_{ij}x_i x_j$ contributes $Q_{ij}$ to the function value when both $x_i = 1$ and $x_j = 1$.

Overall, QUBO seeks to minimize the function $f_Q$ over the set of binary vectors by determining an optimal minimizer $x^*$,

$$x^* = \arg\min_{x\in\mathbb{B}^n} f_Q(x). \tag{6.4.6}$$

Incorporating the variational quantum algorithm into QUBO, we reformulate the cost function using the substitution:

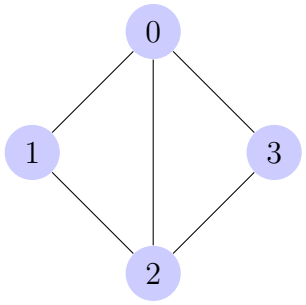
$$x_i = \frac{1 - Z_i}{2} \text{ or } \frac{1 + Z_i}{2}, \tag{6.4.7}$$

where the variable $x_i$ is supplanted by the Pauli Z matrix operating on the $i$-th qubit. This replacement facilitates the formulation of a model Hamiltonian. Its ground state can be approximated by minimizing the expected energy via the variational quantum algorithm, as elaborated in section 6.4.

In the following sections, we evaluate the performance of the noisy GD and RCD across various QUBO applications, focusing on the ground state energy estimation. These applications encompass Max-Cut in section 6.4, the Traveling Salesman Problem in section 6.4, and Variational Quantum Factoring in section 6.4.

### Max-Cut

For the Max-Cut problem, the graph employed in our numerical experiments is presented as follows:




The global cost function is designed to maximize $C = \sum_{(i,j)\in E} x_i(1 - x_j)$, where $E$ represents the edges in the graph. For the given graph, the QUBO problem can be formulated as:

$$\min_{x_i\in\{0,1\}} -3x_0^2 + 2x_0x_1 + 2x_0x_2 + 2x_0x_3 - 2x_1^2 + 2x_1x_2 - 3x_2^2 + 2x_2x_3 - 2x_3^2 \,.$$

In order to construct the corresponding Hamiltonian, we associate the binary variables $x_i$ with the Pauli $Z$ matrices, denoted as $Z_i$, which act on individual qubits. Taking into account the relationship between the binary variables $x_i$ and the Pauli matrices $Z_i$, defined by the equation $x_i = \frac{1-Z_i}{2}$, the cost function is articulated by the Hamiltonian:

$$H = \frac{1}{2}I - 3Z_0 + \frac{1}{2}Z_0Z_1 + \frac{1}{2}Z_0Z_2 + \frac{1}{2}Z_0Z_3 + \frac{1}{2}Z_1Z_2 + \frac{1}{2}Z_2Z_3. \tag{6.4.8}$$

Using this Hamiltonian, we construct a parameterized quantum circuit with four qubits ($N = 4$) and 20 parameters. The circuit consists of alternating single-gate rotations, denoted as $U_{\text{single}}(\boldsymbol{\theta}) = \prod_{i=1}^{n} \text{RY}(\theta_i)$ [6] and entangler gate $U_{\text{entangler}}$ [7]. The configuration of the parametrized quantum circuit is illustrated in Figure 6.11 in Section 6.10. This structure resembles the variational quantum circuit of the QAOA, with the ansatz given by $|\psi(\boldsymbol{\theta})\rangle = [U_{\text{single}}(\boldsymbol{\theta})U_{\text{entangler}}]^m |+\rangle$. For the optimization process, we assign a learning rate of 0.1 for GD and 3.0 for RCD and select energy as the optimization metric.

As illustrated in fig. 6.7, the RCD also outperforms GD in this case, as it converges to an energy ratio of 1 with roughly 200 partial derivative evaluations. In contrast, the GD achieves only an average of 0.75 with 1000 derivative evaluations. The superior performance of RCD in fig. 6.7 can again be attributed to the significant values of

[6] Each layer of rotation gates includes a rotation-Y gate applied to every qubit.

[7] The entanglement layer incorporates two-qubit gates for qubit entanglement without tunable parameters. In this experiment, the entangler gate employs controlled-Z gates. For a comprehensive explanation, refer to the circuit architecture in Section 6.10

$\frac{L}{L_{\text{avg}}}$ and $\frac{L}{L_{\max}}$, both exceeding an order of magnitude of 3. As observed from the optimization result, a high ratio of $\frac{L}{L_{\text{avg}}}$ is indicative of the rapid convergence of RCD in this application.

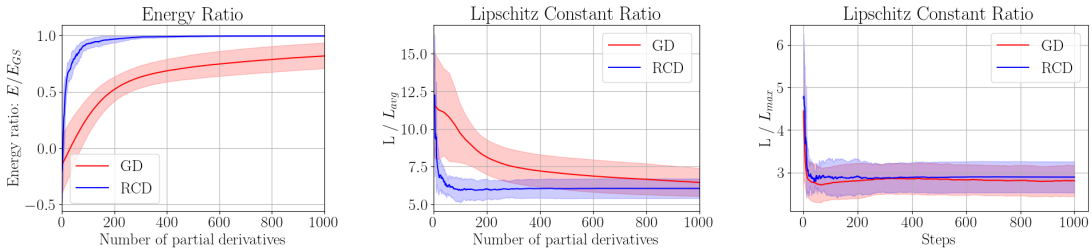



Figure 6.7: Performance comparison between noisy GD and RCD for the Max-cut problem. The corresponding Lipschitz constant ratios, denoted as $\frac{L}{L_{\text{avg}}}$ and $\frac{L}{L_{\max}}$, are presented in the middle and right panels. The shaded areas within the figure represent variations that have been observed across ten random realizations. The optimization process has been performed in 20 dimensions.

### Traveling Salesman Problem (TSP)

We have designed a numerical test for the Traveling Salesman Problem (TSP) using three cities as an example. The intercity costs for these cities are 48, 63, and 91 respectively. The cost of TSP is defined as

$$C(\mathbf{x}) = \sum_{i,j} w_{ij} \sum_{p} x_{i,p} x_{j,p+1} + A \sum_{p} \left(1 - \sum_{i} x_{i,p}\right)^2 + A \sum_{i} \left(1 - \sum_{p} x_{i,p}\right)^2,$$

where $i$ labels the node, $p$ indicates its order, and $x_{i,p}$ is in the set $\{0, 1\}$ and the penalty parameter $A$ is set sufficiently large to effectively enforce constraints. More details regarding the expansion of $C(\mathbf{x})$ can be found in section 6.12.

Utilizing the defined cost function, we establish a model Hamiltonian in the same manner as presented in section 6.4. We aim to prepare its ground state to address the QUBO problem. A detailed representation of the Hamiltonian is available in section 6.12. We construct a parameterized quantum circuit comprising alternating single-gate rotations, represented by $U_{\text{single}}(\boldsymbol{\theta}) = \prod_{i=1}^{n} \mathrm{R_Y}(\theta_i)$ and entangler gate $U_{\text{entangler}}$. This circuit resembles the one depicted in Figure 6.11 in Section 6.10, albeit with a greater number of qubits. The total number of trainable parameters is 90,

which requires nine qubits ($N = 9$) and ten alternating layers. We employ energy as the measure for the optimization cost function.

In the left panel in fig. 6.8, the optimization results obtained from the noisy RCD and GD are plotted. Notably, GD exhibits slower convergence compared to RCD in achieving an energy ratio of 1. The employment of 90 parameters in the optimization, a number markedly greater than those in prior applications, might account for this disparity. This increased parameter count likely requires additional iterations and partial derivative evaluations when applying GD. Similar to previous results, the two types of Lipschitz constant ratios are obtained and shown along with the iterations in fig. 6.8. Again, we can see that the values of the ratios are considerably large, especially during the initial stage of the optimization, underlining the efficiency of RCD in the optimization process.

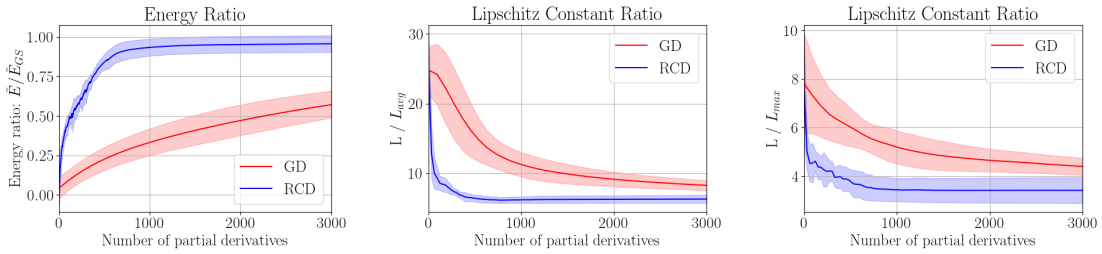



Figure 6.8: Performance comparison between noisy GD and RCD for the TSP problem. The corresponding Lipschitz constant ratios, denoted as $\frac{L}{L_{\text{avg}}}$ and $\frac{L}{L_{\text{max}}}$, are presented in the middle and right panels. The shaded areas within the figure represent variations that have been observed across ten random realizations. The optimization process has been performed in 90 dimensions. In the first panel, $\tilde{E}/\tilde{E}_{GS}$ is defined as $(E - c)/(E_{GS} - c)$, where $c/E_{GS} = 3000$. For clarity in the presentation, we adjust the energy by a constant.

### Variational Quantum Factoring

Our next QUBO problem is designed as a variational quantum factoring task. For this task, we formulated the optimization problem within the framework of quantum adiabatic computation [80, 13]. For example, to factorize 143 into the product of two prime numbers, let $143 = pq$, where

$$p = 8 + 4p_2 + 2p_1 + 1,$$
$$q = 8 + 4q_2 + 2q_1 + 1.$$

Upon direct computation, the relations are simplified to

$$p_1 + q_1 - 1 = 0, \tag{6.4.9}$$
$$p_2 + q_2 - 1 = 0, \tag{6.4.10}$$
$$p_2 q_1 + p_1 q_2 - 1 = 0. \tag{6.4.11}$$

To solve this system of equations, we introduce a cost function

$$c(p_1, q_1, p_2, q_2) = (p_1 + q_1 - 1)^2 + (p_2 + q_2 - 1)^2 + (p_2 q_1 + p_1 q_2 - 1)^2. \tag{6.4.12}$$

By borrowing techniques (see Section. 6.13 for more details) from [344, 274], the cost function can be reduced to

$$c(p_1, q_1, p_2, q_2) = 5 - 3p_1 - p_2 - q_1 + 2p_1 q_1 - 3p_2 q_1 + 2p_1 p_2 q_1 - 3q_2 + p_1 q_2 + 2p_2 q_2 + 2p_2 q_1 q_2. \tag{6.4.13}$$

Following the methods detailed in the QUBO, we treat $(p_1, q_1, p_2, q_2)$ as boolean functions and substitute each boolean with $\frac{1}{2}(1 - Z_i)$ as we did in previous sections. Then, the problem can be reformulated into the Ising Hamiltonian,

$$\begin{aligned} H = &- 3\mathbb{I} + \frac{1}{2}Z_0 + \frac{1}{4}Z_1 + \frac{3}{4}Z_0Z_2 + \frac{1}{4}Z_2 - \frac{1}{4}Z_1Z_2 + \frac{1}{4}Z_0Z_1 \\ &- \frac{1}{4}Z_0Z_1Z_2 + \frac{1}{2}Z_3 + \frac{1}{4}Z_0Z_3 + \frac{3}{4}Z_1Z_3 + \frac{1}{4}Z_2Z_3 - \frac{1}{4}Z_1Z_2Z_3. \end{aligned} \tag{6.4.14}$$

The ground states of this Hamiltonian are $|0110\rangle$ and $|1001\rangle$, which respectively correspond to the solutions for the factorization of the number 143. We summarize it as follows,

$$(p_1, p_2, q_1, q_2) = (0, 1, 1, 0) \longleftrightarrow (p, q) = (13, 11) \tag{6.4.15}$$
$$(p_1, p_2, q_1, q_2) = (1, 0, 0, 1) \longleftrightarrow (p, q) = (11, 13) \tag{6.4.16}$$
$$p = 8 + 4p_2 + 2p_1 + 1 \text{ and } q = 8 + 4q_2 + 2q_1 + 1 \quad \text{Boolean functions.} \tag{6.4.17}$$

In our numerical experiment, we select the mixer Hamiltonian $H_2 = \sum X_i$ and set up a 20-layer QAOA, which corresponds to 40 parameters [8]. We set the learning rates to 0.0001 for GD and 0.005 for RCD and choose the energy as a measure for optimization. Even with a small step size, the variance of GD is notably large. Employing a larger step size for GD further exacerbates the results.

[8]The QAOA ansatz builds the variational circuit by alternating between the parametrized unitary evolution associated with the problem Hamiltonian $H$ and the mixer Hamiltonian $H_2$.

In fig. 6.9, the optimization results of the Hamiltonian equation 6.4.14 are depicted, showing that the number of partial derivative evaluations for the RCD to reach an energy ratio of 1 is about 400 whereas the GD seems to require more than 1000 to the same tolerance. As discussed previously, this observation aligns with prior discussions, particularly given the pronounced magnitude of the Lipschitz constant ratios evident in fig. 6.9.

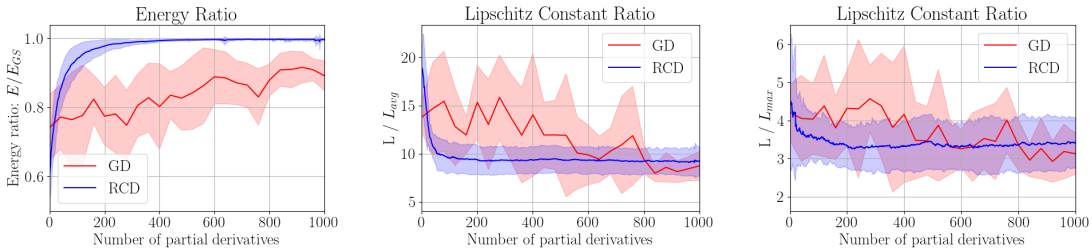



Figure 6.9: Performance comparison between noisy GD and RCD for the quantum factoring problem. The corresponding Lipschitz constant ratios, denoted as $\frac{L}{L_{\text{avg}}}$ and $\frac{L}{L_{\max}}$, are presented in the middle and right panels. The shaded areas within the figure represent variations that have been observed across ten random realizations. The optimization process has been performed in 40 dimensions.

## 6.5 Conclusion

We considered the use of a noisy random coordinate descent method to analyze its potential advantage over the noisy gradient descent, which evaluates all partial derivatives at each step, in the context of variational quantum optimization. Most previous works on randomized coordinate descent algorithms studied the case of convex cost functions, which do not fit into most variational quantum applications that involve non-convex cost functions. In this work, we generalized the conventional convergence analysis of randomized coordinate descent to local convergence analysis under a local-PL condition that can capture a large class of non-convex optimization. In particular, we proved that noisy randomized coordinate descent can converge faster than noisy gradient descent in terms of the total cost, measured in terms of the total number of partial derivative estimations. In addition, we conducted extensive numerical experiments implementing both methods for many interesting quantum optimization problems. We observed that noisy randomized coordinate descent typically demands less measurement cost than noisy gradient descent, thereby

demonstrating its efficiency in many non-convex quantum applications. From an optimization standpoint, variational quantum optimization as outlined in problem 1 raises many interesting questions. For example, can second order, or zeroth order optimization methods (i.e., methods using only function evaluation) be more efficient compared to the current gradient-based algorithms? In a technical viewpoint, another question is whether the stability result lemma 5 can be generalized so that the event covers the case that the iteration diverges at some timepoint, but it remains in the entire basin until then, $f^{-1}[0, \delta_f)$, not necessarily in the region above the set of global minima, $f^{-1}\left(\frac{La\sigma_\infty^2 d}{\mu}, \delta_f\right)$. If this is possible to show, then it will provide stronger result such as the stability of the noisy GD and RCD within the entire basin as the stability of Markov Chain in [183].

## 6.6 Stochastic stability of noisy GD

In this section, we prove Lemma 5.

*Proof of Lemma 5.* Define the probability filtration: $\mathcal{F}_n = \sigma(\boldsymbol{\theta}_k | k \leq n)$ and the stopping time[9]

$$\tau = \inf\left\{k : f(\boldsymbol{\theta}_k) \leq \frac{La\sigma_\infty^2 d}{\mu}\right\},$$

which is the smallest timepoint where the noisy GD achieves $f(\boldsymbol{\theta}_k) \leq \frac{La\sigma_\infty^2 d}{\mu}$.

Define the indicator function $\mathbb{I}_n$:

$$\mathbb{I}_n = \begin{cases} 1, & \text{if} \quad \{\boldsymbol{\theta}_k\}_{k=1}^{n-1} \subset f^{-1}([0, \delta_f)) \\ 0, & \text{otherwise} \end{cases}, \tag{6.6.1}$$

and the stochastic process

$$V_n = \begin{cases} f(\boldsymbol{\theta}_n)\mathbb{I}_n, & n < \tau \\ f(\boldsymbol{\theta}_\tau)\mathbb{I}_\tau, & n \geq \tau \end{cases}.$$

According to the definition of $V_n$, there are complementary and exclusive events (cases):

[9] It is straightforward to see

$$\{\tau \leq n\} \in \mathcal{F}_n, \quad \{\tau > n\} \in \mathcal{F}_n.$$

- Case 1: If there exists $0 < n < \infty$ such that: 1. $\boldsymbol{\theta}_n \notin \mathcal{N}$; 2. For any $m < n$, $\boldsymbol{\theta}_m \in \mathcal{N}$ and $f(\boldsymbol{\theta}_m) > \frac{La\sigma_\infty^2 d}{\mu}$. Then

$$V_n \geq \delta_f \Rightarrow \sup_n V_n \geq \delta_f \,.$$

- Case 2: For any $n < \tau$, $f(\boldsymbol{\theta}_n) \in \mathcal{N}$.

We observe that Case 2 is the stable situation, indicating that $f(\boldsymbol{\theta}_n)$ remains in the basin of the global minimum until it achieves a small loss[10]. To prove equation 6.3.2, it suffices to show that

$$\mathbb{P}(\Omega_1) \leq \frac{f(\boldsymbol{\theta}_1)}{\delta_f}, \tag{6.6.2}$$

where $\Omega_1$ denotes the event associated with Case 1.

Now, we show that $V_n$ is a supermartingale to bound $\sup_n V_n$. Taking the conditional expectation, we obtain

$$\mathbb{E}(V_{n+1}|\mathcal{F}_n) = \mathbb{E}(V_{n+1}|\mathcal{F}_n, \mathbb{I}_n = 1, \tau \leq n)\mathbb{P}(\tau \leq n) + \mathbb{E}(V_{n+1}|\mathcal{F}_n, \mathbb{I}_n = 1, \tau > n)\mathbb{P}(\tau > n)\,,$$

where we use $\mathbb{I}_{n+1} \leq \mathbb{I}_n$. There are two terms in the above equation:

- For the first term, when $\tau \leq n$, we obtain $V_{n+1} = V_\tau = V_n$. This implies

$$\mathbb{E}(V_{n+1}|\mathcal{F}_n, \mathbb{I}_n = 1, \tau \leq n) = V_n\,. \tag{6.6.3}$$

- For the second term, when $\tau > n$, we have $f(\boldsymbol{\theta}_n) > \frac{La\sigma_\infty^2 d}{\mu}$. Then, taking the conditional expectation yields

$$\begin{aligned}
&\mathbb{E}[V_{n+1}|\mathbb{I}_n = 1, \boldsymbol{\theta}_1, \tau > n] \\
=&\mathbb{E}[f(\boldsymbol{\theta}_{n+1})\mathbb{I}_{n+1}|\mathbb{I}_n = 1, \boldsymbol{\theta}_1, \tau > n] \\
\leq& f(\boldsymbol{\theta}_n) - a\|\nabla f(\boldsymbol{\theta}_n)\|^2 + \frac{La^2}{2}(\|\nabla f(\boldsymbol{\theta}_n)\|^2 + \sigma_\infty^2 d) \\
\leq& (1-\mu a) f(\boldsymbol{\theta}_n) + \frac{La^2\sigma_\infty^2 d}{2} \\
<& (1-\mu a) f(\boldsymbol{\theta}_n) + \frac{\mu a}{2} f(\boldsymbol{\theta}_n) \\
\leq& \left(1 - \frac{\mu a}{2}\right) f(\boldsymbol{\theta}_n)\mathbb{I}_n = \left(1 - \frac{\mu a}{2}\right) V_n,
\end{aligned} \tag{6.6.4}$$

[10]We emphasize that Case 2 also includes the situation where $f(\boldsymbol{\theta}_n)$ remains in the basin and never achieves the small loss.

where we use assumption 3 and $a < \frac{1}{L}$ in the second inequality, $\tau > n$ in the third inequality.

Combining equation 6.6.3 and equation 6.6.4, we obtain

$$\mathbb{E}(V_{n+1}|\mathcal{F}_n) = V_n\mathbb{P}(\tau \leq n) + \left(1 - \frac{\mu a}{2}\right) V_n\mathbb{P}(\tau > n) \leq V_n\,. \tag{6.6.5}$$

Thus, $V_n$ is a supermartingale.

Now, we consider the Case 1 event:

$$\Omega_1 = \{\exists n > 1, \boldsymbol{\theta}_n \notin \mathcal{N} \text{ and } f(\boldsymbol{\theta}_m) > \epsilon \text{ with } \boldsymbol{\theta}_m \in \mathcal{N},\ \forall 1 \leq m < n\} \subset \left\{\sup_n V_n \geq \delta_f\right\}\,.$$

Because $V_n$ is a supermartingale, we obtain Case 1 happens with small probability:

$$\mathbb{P}(\Omega_1) \leq \frac{V_1}{\delta_f} = \frac{f(\boldsymbol{\theta}_1)}{\delta_f}\,.$$

This concludes the proof. □

## 6.7 Stochastic stability of noisy RCD

In this section, we prove lemma 6 with a slight modification of the proof in section 6.6. From a theoretical viewpoint, the difference between the noisy GD and RCD methods is made by the construction of gradient estimate (e.g. see equation 6.1.5 and equation 6.1.8). Compared to GD, the additional randomness of RCD comes with the random selection of a component as in equation 6.1.8. This difference affects the recursive inequality equation 6.6.4 mainly in the previous proof, where we considered the properties of the gradient estimator. From this observation, it suffices to derive a recursive inequality similar to equation 6.6.4 to prove lemma 6.

Note that the sampling of a component within RCD is performed before estimating a partial derivative. Thus, the first step is to take expectation on the partial derivative estimate,

$$\mathbb{E}_{\xi_{i_n}}[f(\boldsymbol{\theta}_{n+1})] \leq f(\boldsymbol{\theta}_n) - a\mathbb{E}_{\xi_{i_n}}[\partial_{i_n} f(\boldsymbol{\theta}_n) g_{i_n}(\boldsymbol{\theta}_n)] + \frac{L_{i_n} a^2}{2}\mathbb{E}_{\xi_{i_n}}[|g_{i_n}(\boldsymbol{\theta}_n)|^2]\,, \tag{6.7.1}$$

where $i_n$ is uniformly sampled index and $g_{i_n}$ is the corresponding unbiased estimate for the partial derivative. Let $\mathcal{F}_n$, $\tau$, $\mathbb{I}_n$, and $V_n$ be as defined in the previous section.

By considering the inequality equation 6.7.1 and the conditional expectation in equation 6.6.4, we achieve the following result by taking expectations with respect to the random index $i_n$,

$$\begin{aligned}
&\mathbb{E}[V_{n+1}|\mathbb{I}_n = 1, \boldsymbol{\theta}_1, \tau > n] \\
=&\mathbb{E}[f(\boldsymbol{\theta}_{n+1})\mathbb{I}_{n+1}|\mathbb{I}_n = 1, \boldsymbol{\theta}_1, \tau > n] \\
\leq& \left( f(\boldsymbol{\theta}_n) - \frac{a}{d}\|\nabla f(\boldsymbol{\theta}_n)\|^2 + \frac{L_{\max}a^2}{2d}\|\nabla f(\boldsymbol{\theta}_n)\|^2 + \frac{L_{\text{avg}}\sigma_\infty^2 d a^2}{2d} \right) \mathbb{I}_{n+1} \\
\leq& \left( \left(1 - \frac{\mu a}{d}\right) f(\boldsymbol{\theta}_n) + \frac{L_{\text{avg}}a^2\sigma_\infty^2}{2} \right) \mathbb{I}_{n+1} \\
<& \left( \left(1 - \frac{\mu a}{d}\right) f(\boldsymbol{\theta}_n) + \frac{\mu a}{2d} f(\boldsymbol{\theta}_n) \right) \mathbb{I}_{n+1} \\
=& \left(1 - \frac{\mu a}{2d}\right) f(\boldsymbol{\theta}_n)\mathbb{I}_n = \left(1 - \frac{\mu a}{2d}\right) V_n,
\end{aligned} \tag{6.7.2}$$

provided that $f(\boldsymbol{\theta}_n) > \frac{L_{\text{avg}}a\sigma_\infty^2 d}{\mu}$ and $a_n = a < \min\left\{\frac{1}{L_{\max}}, \frac{d}{\mu}, \frac{2\mu\delta_f}{L_{\text{avg}}\sigma_\infty^2 d}\right\}$.

Similar to equation 6.6.5, in the case of RCD, equation 6.7.2 implies

$$\mathbb{E}(V_{n+1}|\mathcal{F}_n) = V_n\mathbb{P}(\tau \leq n) + \left(1 - \frac{\mu a}{2d}\right) V_n\mathbb{P}(\tau > n) \leq V_n\,, \tag{6.7.3}$$

which implies $V_n$ forms a supermartingale. The remaining proof of lemma 6 follows the same steps as the proof of lemma 5, so we will not include them here.

## 6.8 The proofs of theorem 7 and theorem 8

We first show the convergence rate of the noisy GD method, followed by a similar analysis for the noisy RCD method. The following proofs are similar to those in section 6.6 and section 6.7 with minor differences.

*Proof of Theorem 7.* Define the probability filtration: $\mathcal{F}_n = \sigma\left(\boldsymbol{\theta}_k | k \leq n\right)$ and the stopping time

$$\tau = \inf\left\{k : f(\boldsymbol{\theta}_k) \leq \epsilon\right\}\,,$$

which is the smallest timepoint where the noisy GD achieves $f(\boldsymbol{\theta}_k) \leq \epsilon$. Then, our ultimate goal is to show that the event that $\inf_{1\leq n\leq N} f(\boldsymbol{\theta}_n) \leq \epsilon$ occurs with high probability, say, at least $1 - \eta$. Then, our goal is to show that for any $\eta \in \left(\frac{f(\boldsymbol{\theta}_1)}{\delta_f}, 1\right)$, there exists a sufficiently large $N$ such that

$$p_{\text{fail}} := \mathbb{P}(\tau > N) \leq \eta. \tag{6.8.1}$$

Define the indicator function $\mathbb{I}_n$:

$$\mathbb{I}_n = \begin{cases} 1, & \text{if} \quad \{\boldsymbol{\theta}_k\}_{k=1}^{n-1} \subset f^{-1}\left([0, \delta_f)\right) \\ 0, & \text{otherwise} \end{cases},$$

and the stochastic process

$$V_n = \begin{cases} f(\boldsymbol{\theta}_n)\mathbb{I}_n, & n < \tau \\ f(\boldsymbol{\theta}_\tau)\mathbb{I}_\tau, & n \geq \tau \end{cases}.$$

Define the unstable event:

$$\Omega = \{\exists n > 1, \boldsymbol{\theta}_n \notin \mathcal{N} \text{ and } f(\boldsymbol{\theta}_m) > \epsilon,\ \forall 1 \leq m < n\} \subset \left\{\sup_n V_n \geq \delta_f\right\}.$$

According to Lemma 5 and the proof in section 6.6, for learning rate $a$ with $\frac{La\sigma_\infty^2 d}{\mu} < \epsilon$, we obtain $\Omega$ happens with small probability:

$$\mathbb{P}(\Omega) \leq \frac{V_1}{\delta_f} = \frac{f(\boldsymbol{\theta}_1)}{\delta_f}. \tag{6.8.2}$$

Recalling equation 6.8.1, we note that, for any $n \leq N$,

$$\mathbb{P}(\tau > n) \geq p_{\text{fail}}.$$

Plugging this into equation 6.6.5, we obtain that

$$\begin{aligned} \mathbb{E}(V_{n+1}|\mathcal{F}_n) &= \left(1 - \mathbb{P}(\tau > n) + \left(1 - \frac{\mu a}{2}\right)\mathbb{P}(\tau > n)\right) V_n \\ &= \left(1 - \frac{\mu a \mathbb{P}(\tau > n)}{2}\right) V_n \\ &\leq \left(1 - \frac{\mu a p_{\text{fail}}}{2}\right) V_n. \end{aligned} \tag{6.8.3}$$

By taking the total expectation on both sides and using a telescoping trick, we achieve that

$$\mathbb{E}(V_{n+1}) \leq \left(1 - \frac{\mu a p_{\text{fail}}}{2}\right)^n V_1 = \left(1 - \frac{\mu a p_{\text{fail}}}{2}\right)^n f(\boldsymbol{\theta}_1). \tag{6.8.4}$$

This means that if the probability of failure, $p_{\text{fail}}$, is large, the expectation of $V_{n+1}$ decreases quickly. By Markov's inequality, we have

$$\mathbb{P}\left(V_N > \epsilon\right) \leq \frac{\left(1 - \frac{\mu a p_{\text{fail}}}{2}\right)^{N-1} f(\boldsymbol{\theta}_1)}{\epsilon},$$

equivalently,

$$\mathbb{P}\left(V_N \leq \epsilon\right) \geq 1 - \frac{\left(1 - \frac{\mu a p_{\text{fail}}}{2}\right)^{N-1} f(\boldsymbol{\theta}_1)}{\epsilon}\,.$$

Now, if we consider the event $\{V_N \leq \epsilon\}$, it is the union of the following two events (not necessarily exclusive and complementary), which are slightly different from the ones in section 6.6:

- $\Omega_1$: There exists $n \leq N$ such that $f(\boldsymbol{\theta}_n) \leq \epsilon$ and $\boldsymbol{\theta}_n \in \mathcal{N}$. This means

  $$\inf_{1 \leq n \leq N} f(\boldsymbol{\theta}_n) \leq \epsilon\,.$$

  We want to show that $\Omega_1$ happens with high probability.

- $\Omega_2$: There exists $n < N$ such that $f(\boldsymbol{\theta}_n) > \delta_f$ and $f(\boldsymbol{\theta}_m) > \epsilon$ for any $m < n$.

  We note that, when $\Omega_2$ happens, we have $V_{n+1} = 0$ with $f(\boldsymbol{\theta}_n) > \delta_f$, which implies $\Omega_2 \subset \Omega$. According to equation 6.8.2, we obtain

  $$\mathbb{P}\left(\Omega_2\right) \leq \mathbb{P}\left(\Omega\right) \leq \frac{f(\boldsymbol{\theta}_1)}{\delta_f}\,.$$

Now, we give a lower bound for the event $\Omega_1$:

$$\mathbb{P}\left(\inf_{1 \leq n \leq N} f(\boldsymbol{\theta}_n) \leq \epsilon\right) = \mathbb{P}\left(\Omega_1\right) \geq \mathbb{P}\left(V_N \leq \epsilon\right) - \mathbb{P}(\Omega_2) \geq 1 - \frac{\left(1 - \frac{\mu a p_{\text{fail}}}{2}\right)^{N} f(\boldsymbol{\theta}_1)}{\epsilon} - \frac{f(\boldsymbol{\theta}_1)}{\delta_f}. \tag{6.8.5}$$

Notice

$$\mathbb{P}\left(\inf_{1 \leq n \leq N} f(\boldsymbol{\theta}_n) \leq \epsilon\right) \leq \mathbb{P}(\tau \leq N) = 1 - p_{\text{fail}}\,.$$

Combining the above two inequalities, we have

$$p_{\text{fail}} \leq \frac{\left(1 - \frac{\mu a p_{\text{fail}}}{2}\right)^{N} f(\boldsymbol{\theta}_1)}{\epsilon} + \frac{f(\boldsymbol{\theta}_1)}{\delta_f}\,. \tag{6.8.6}$$

Next, we show equation 6.8.1 using the proof by contradiction. Assume that the conclusion of the theorem is not true, meaning that for some $\eta \in \left(\frac{f(\boldsymbol{\theta}_1)}{\delta_f}, 1\right)$ and every $N$,

$$p_{\text{fail}} > \eta\,.$$

When $p_{\text{fail}} > \eta$ and $N = \frac{2}{\mu a \eta} \log\left(\frac{f(\boldsymbol{\theta_1})}{\left(\eta - \frac{f(\boldsymbol{\theta}_1)}{\delta_f}\right)\epsilon}\right)$, then

$$\begin{aligned}
&\frac{\left(1 - \frac{\mu a p_{\text{fail}}}{2}\right)^N f(\boldsymbol{\theta}_1)}{\epsilon} + \frac{f(\boldsymbol{\theta}_1)}{\delta_f} \\
<&\frac{\left(1 - \frac{\mu a \eta}{2}\right)^N f(\boldsymbol{\theta}_1)}{\epsilon} + \frac{f(\boldsymbol{\theta}_1)}{\delta_f} \\
\leq&\frac{\exp\left(-\frac{\mu a \eta N}{2}\right) f(\boldsymbol{\theta}_1)}{\epsilon} + \frac{f(\boldsymbol{\theta}_1)}{\delta_f} = \eta < p_{\text{fail}}
\end{aligned}$$

where we use $p_{\text{fail}} > \eta$ in the first inequality and $(1-x)^N \leq \exp(-xN)$ in the second inequality. This contradicts to equation 6.8.6. Thus, equation 6.8.1 must be true and we conclude the proof.

□

*Proof of Theorem 8.* Denote the probability of failure

$$p_{\text{fail}} = \mathbb{P}(\tau > N).$$

Similar to the calculation in the previous proof, from equation 6.7.3, we have

$$\mathbb{P}\left(\inf_{1 \leq n \leq N} f(\boldsymbol{\theta}_n) \leq \epsilon\right) \geq 1 - \frac{\left(1 - \frac{\mu a p_{\text{fail}}}{2d}\right)^N f(\boldsymbol{\theta}_1)}{\epsilon} - \frac{f(\boldsymbol{\theta}_1)}{\delta_f}. \tag{6.8.7}$$

With the same logic below equation 6.8.5, we conclude the proof of theorem 8. □

## 6.9 Parameterized Circuit for the VQE

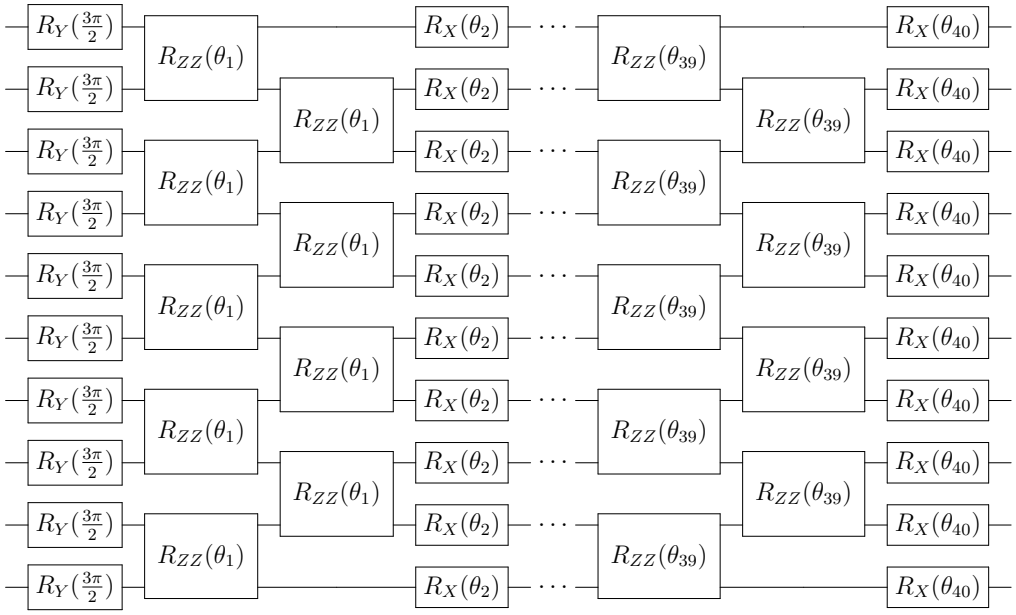



Figure 6.10: A QAOA-like ansatz motivated by [331] is used for the TFIM model equation 6.1.10 with 10 qubits. For the result in fig. 6.1, 40 parameters are assigned with 20 layers of alternating rotation ZZ gates and rotation X gates. The circuit depth is 61.

## 6.10 Parameterized Circuit for the VQE in QUBO experiments

The quantum circuit described below is utilized in the QUBO experiments.

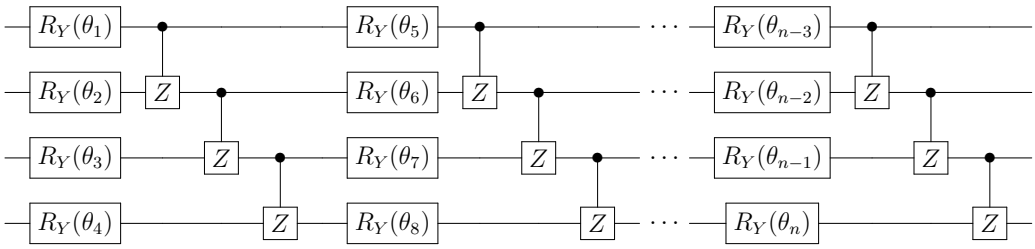



Figure 6.11: A parametrized quantum circuit is employed in the QUBO experiments. This circuit features alternating layers of single rotation gates and entangling controlled-z gates. The adjustable parameters are exclusively found in the single rotation gates, and these parameters vary across different layers and qubits.

## 6.11 Additional histograms of partial derivative estimates

fig. 6.2 plots the histograms with respect to the first 12 parameters among 40. The rest of 28 histograms are shown in the following figures. It is observed in all figures that the variances of partial derivative estimates in all directions are a similar magnitude of value.

## 6.12 Cost function for the TSP

First, the cost function is defined as

$$C(\mathbf{x}) = \sum_{i,j} w_{ij} \sum_{p} x_{i,p} x_{j,p+1} + A \sum_{p} \left(1 - \sum_{i} x_{i,p}\right)^2 + A \sum_{i} \left(1 - \sum_{p} x_{i,p}\right)^2 ,$$

where $A = 10000$ , $w_{12} = w_{21} = 48$, $w_{13} = w_{31} = 91$, and $w_{23} = w_{32} = 63$, and $w_{ii} = 0, i = 1, 2, 3$.

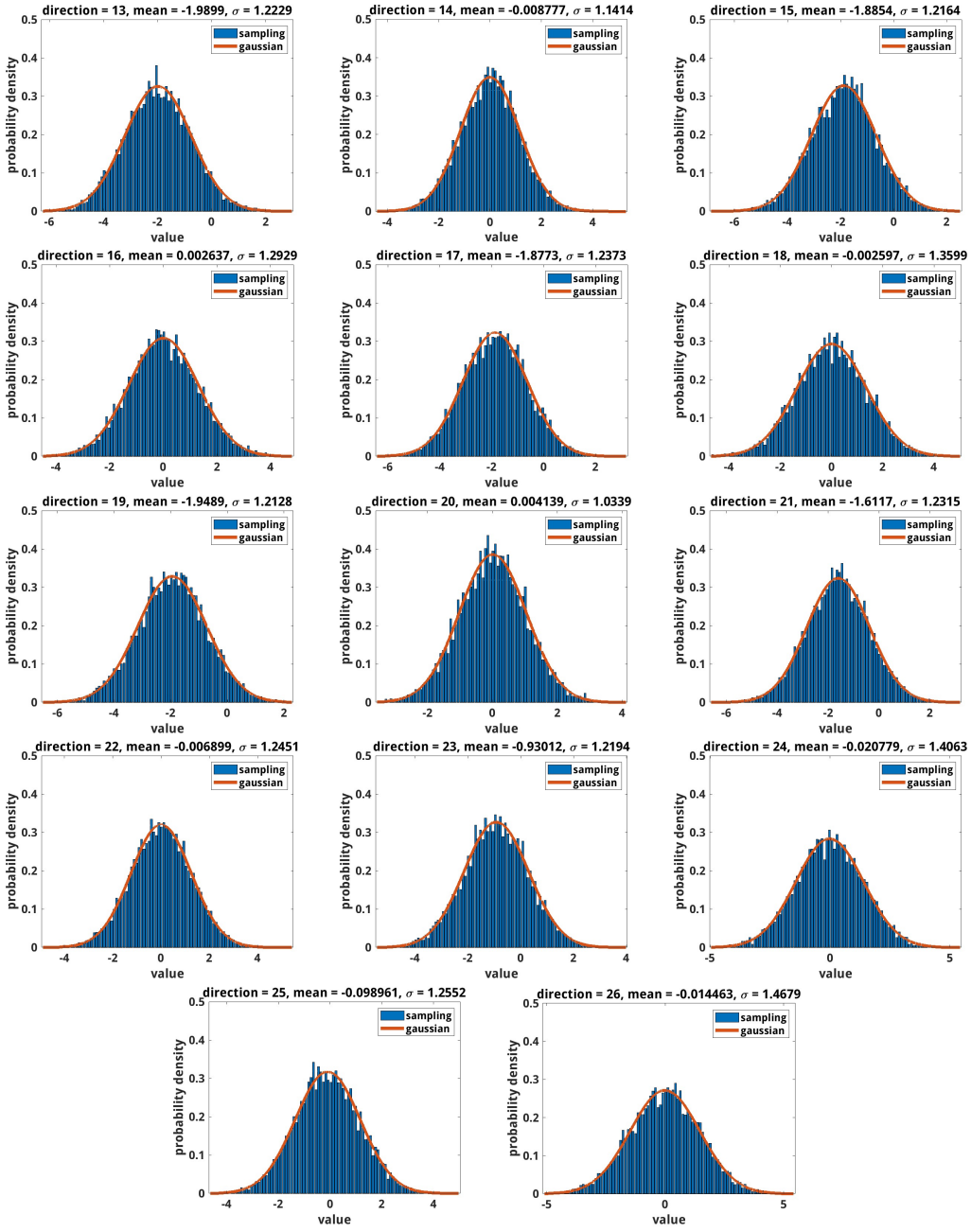



Figure 6.12: The histogram of partial derivative estimates with respect to the 13-th to the 26-th parameters are plotted with the same setup as in fig. 6.2.

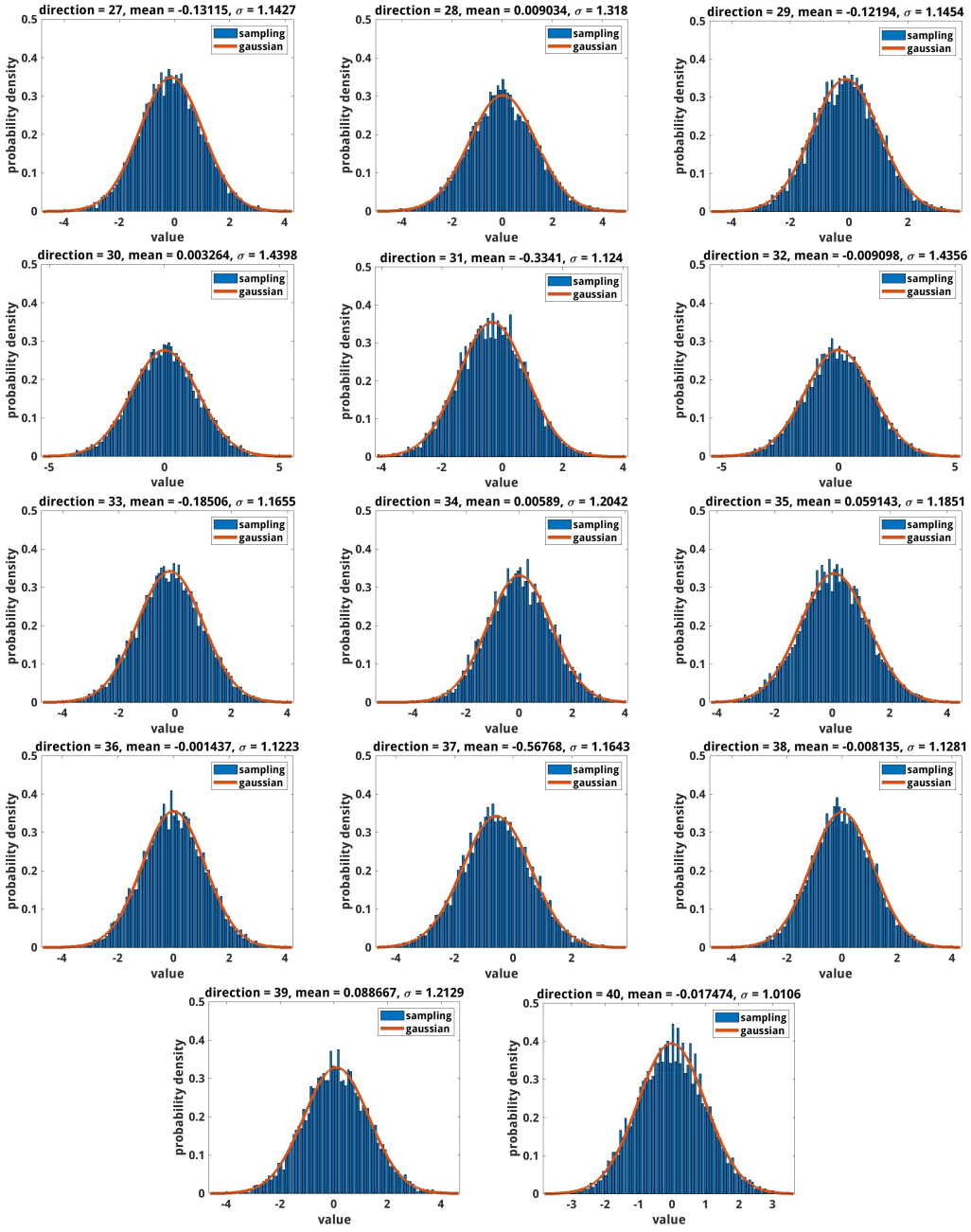



Figure 6.13: The histogram of partial derivative estimates with respect to the 27-th to the 40-th parameters are plotted with the same setup as in fig. 6.2.

We can introduce a new Boolean variable, denoted by $\tilde{x}_{3i+j-4} = x_{i,j}$, where $i, j = 1, 2, 3$. For simplicity, in the following formula, we will use $x_0, \ldots, x_8$ to represent $\tilde{x}_0, \ldots, \tilde{x}_8$. With this notation, the expanded form of the cost function can be expressed as:

$$\begin{aligned}
C(\boldsymbol{x}) = &-200000x_0 - 200000x_1 - 200000x_2 - 200000x_3 - 200000x_4 - 200000x_5 \\
&- 200000x_6 - 200000x_7 - 200000x_8 \\
&+ [200000x_0x_1 + 200000x_0x_2 + 200000x_0x_3 + 48x_0x_4 + 48x_0x_5 \\
&+ 200000x_0x_6 + 91x_0x_7 + 91x_0x_8 + 200000x_1x_2 + 48x_1x_3 \\
&+ 200000x_1x_4 + 48x_1x_5 + 91x_1x_6 + 200000x_1x_7 + 91x_1x_8 \\
&+ 48x_2x_3 + 48x_2x_4 + 200000x_2x_5 + 91x_2x_6 + 91x_2x_7 \\
&+ 200000x_2x_8 + 200000x_3x_4 + 200000x_3x_5 + 200000x_3x_6 \\
&+ 63x_3x_7 + 63x_3x_8 + 200000x_4x_5 + 63x_4x_6 + 200000x_4x_7 \\
&+ 63x_4x_8 + 63x_5x_6 + 63x_5x_7 + 200000x_5x_8 + 200000x_6x_7 \\
&+200000x_6x_8 + 200000x_7x_8] + 600000
\end{aligned}$$

In order to build the corresponding Hamiltonian, we align the binary variables $x_i$ with the Pauli $Z$ matrices, which operate on individual qubits, and are represented by $Z_i$. Taking into account the relationship between the binary variables $x_i$ and the Pauli $Z$ matrices, defined by the equation $x_i = \frac{1-Z_i}{2}$, we can express the Hamiltonian for QUBO as follows,

$$\begin{aligned}
H_{\text{TSP}} = 600303.0 &- 100069.5Z_0 - 100055.5Z_4 + 12.0Z_4Z_0 - 100069.5Z_1 \\
&- 100055.5Z_5 + 12.0Z_5Z_1 - 100069.5Z_2 - 100055.5Z_3 + 12.0Z_3Z_2 \\
&- 100077.0Z_7 + 22.75Z_7Z_0 - 100077.0Z_8 + 22.75Z_8Z_1 \\
&- 100077.0Z_6 + 22.75Z_6Z_2 + 12.0Z_3Z_1 + 12.0Z_4Z_2 \\
&+ 12.0Z_5Z_0 + 15.75Z_7Z_3 + 15.75Z_8Z_4 + 15.75Z_6Z_5 \\
&+ 22.75Z_6Z_1 + 22.75Z_7Z_2 + 22.75Z_8Z_0 \\
&+ 15.75Z_6Z_4 + 15.75Z_7Z_5 + 15.75Z_8Z_3 \\
&+ 50000.0Z_3Z_0 + 50000.0Z_6Z_0 + 50000.0Z_6Z_3 \\
&+ 50000.0Z_4Z_1 + 50000.0Z_7Z_1 + 50000.0Z_7Z_4 \\
&+ 50000.0Z_5Z_2 + 50000.0Z_8Z_2 + 50000.0Z_8Z_5 \\
&+ 50000.0Z_1Z_0 + 50000.0Z_2Z_0 + 50000.0Z_2Z_1 \\
&+ 50000.0Z_4Z_3 + 50000.0Z_5Z_3 + 50000.0Z_5Z_4 \\
&+ 50000.0Z_7Z_6 + 50000.0Z_8Z_6 + 50000.0Z_8Z_7
\end{aligned}$$

## 6.13 Technique used in quantum factoring

The introduced technique proposes an alternative formulation for equations of the type $AB + S = 0$. Here, $A$ and $B$ represent Boolean variables, while $S$ denotes integers with $S \in \mathbb{Z}$. The optimization algorithm targets the minimization of the quadratic version of this equation.

Given the problem Hamiltonian, defined as $H = (AB + S)^2$, it can be restructured as:

$$H = 2\left[\frac{1}{2}\left(A + B - \frac{1}{2}\right) + S\right]^2 - \frac{1}{8}. \tag{6.13.1}$$

While the two Hamiltonians are not generally equivalent, they do share the same minimizer due to their underlying Boolean function properties. For instance:

- When $AB = 1$: The minimizer for the first Hamiltonian dictates $S = -1$. In the reformulated version, we get

$$\begin{aligned} H &= 2\left[\frac{1}{2}\left(1 + 1 - \frac{1}{2}\right) - 1\right]^2 - \frac{1}{8} \\ &= 0. \end{aligned}$$

- When $AB = 0$: At least one of $A$ or $B$ is zero. Assuming $A = 0$ (without loss of generality) and due to the minimizer, we get $S = 0$. This also minimizes the reformulated Hamiltonian since, regardless of whether $B$ is 0 or 1, the result remains 0.

Thus, the reformulated version can be employed interchangeably in certain scenarios. However, this updated representation leads to a significant reduction in the many-body interactions observed experimentally. Specifically, the quartic terms in the Ising Hamiltonian are eliminated, simplifying experimental realizations. As a result, the third Hamiltonian term $(p_2q_1 + p_1q_2 - 1)^2$ in Eqn.(6.4.12) is reformulated as:

$$H' = 2\left[\frac{1}{2}\left(p_1 + q_2 - \frac{1}{2}\right) + p_2q_1 - 1\right]^2 - \frac{1}{8}.$$

# Bibliography


[1] Martın Abadi et al. "TensorFlow: Large-scale machine learning on heterogeneous systems, 2015". In: *Software available from tensorflow. org* 1.2 (2015).

[2] Amira Abbas et al. "On quantum backpropagation, information reuse, and cheating measurement collapse". In: *arXiv preprint arXiv:2305.13362* (2023).

[3] Bruce Abramson. *The expected-outcome model of two-player games*. Morgan Kaufmann, 2014.

[4] Francisco Albarrán-Arriagada et al. "Measurement-based adaptation protocol with quantum reinforcement learning". In: *Phys. Rev. A* 98 (4 2018), p. 042315. DOI: 10.1103/PhysRevA.98.042315. URL: https://link.aps.org/doi/10.1103/PhysRevA.98.042315.

[5] Sergio Altares-López, Angela Ribeiro, and Juan José García-Ripoll. "Automatic design of quantum feature maps". In: *Quantum Science and Technology* 6.4 (Aug. 2021), p. 045015.

[6] S. Amari and S. C. Douglas. "Why natural gradient?" In: *Proceedings of the 1998 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP'98*. Vol. 2. IEEE. 1998, pp. 1213–1216.

[7] Shun-ichi Amari. "Natural gradient works efficiently in learning". In: *Neural computation* 10.2 (1998), pp. 251–276.

[8] Zheng An and DL Zhou. "Deep Reinforcement Learning for Quantum Gate Control". In: *arXiv preprint arXiv:1902.08418* (2019).

[9] Abhinav Anand, Matthias Degroote, and Alán Aspuru-Guzik. "Natural evolutionary strategies for variational quantum computation". In: *Machine Learning: Science and Technology* 2.4 (July 2021), p. 045012. DOI: 10.1088/2632-2153/abf3ac. URL: https://dx.doi.org/10.1088/2632-2153/abf3ac.

[10] Abhinav Anand, Matthias Degroote, and Alan Aspuru-Guzik. "Natural Evolutionary Strategies for Variational Quantum Computation". In: *arXiv preprint arXiv:2012.00101* (2020).

[11] Panagiotis G. Anastasiou et al. "Qubit-excitation-based adaptive variational quantum eigensolver". In: *Nature* 606.7893 (2022), pp. 788–793.

[12] Panagiotis G. Anastasiou et al. "TETRIS-ADAPT-VQE: An adaptive algorithm that yields shallower, denser circuit ansätze". In: *arXiv preprint arXiv:2209.03585* (2022).

[13] Eric Anschuetz et al. "Variational quantum factoring". In: *Quantum Technology and Optimization Problems: First International Workshop, QTOP 2019, Munich, Germany, March 18, 2019, Proceedings 1*. Springer. 2019, pp. 74–85.

[14] Martin Anthony et al. "Quadratic reformulations of nonlinear binary optimization problems". In: *Mathematical Programming* 162 (2017), pp. 115–144.

[15] Martin Arjovsky, Soumith Chintala, and Léon Bottou. "Wasserstein GAN". In: *arXiv preprint arXiv:1701.07875* (2017).

[16] Frank Arute et al. "Hartree-Fock on a superconducting qubit quantum computer". In: *Science* 369.6507 (2020), pp. 1084–1089.

[17] Frank Arute et al. "Quantum supremacy using a programmable superconducting processor". In: *Nature* 574.7779 (2019), pp. 505–510. URL: `https://www.nature.com/articles/s41586-019-1666-5`.

[18] Peter Auer, Nicolo Cesa-Bianchi, and Paul Fischer. "Finite-time analysis of the multiarmed bandit problem". In: *Machine learning* 47.2-3 (2002), pp. 235–256.

[19] Moritz August and José Miguel Hernández-Lobato. "Taking gradients through experiments: LSTMs and memory proximal policy optimization for black-box quantum control". In: *International Conference on High Performance Computing*. Springer. 2018, pp. 591–613.

[20] Leonardo Banchi and Gavin E Crooks. "Measuring analytic gradients of general quantum evolution with the stochastic parameter shift rule". In: *Quantum* 5 (2021), p. 386.

[21] Dor Bank, Noam Koenigstein, and Raja Giryes. "Autoencoders. arXiv". In: *arXiv preprint arXiv:2003.05991* (2020).

[22] Edwin Barnes et al. "Qubit-ADAPT-VQE: An Adaptive Algorithm". In: *PRX Quantum* 2.2 (2021), p. 020310.

[23] Raef Bassily, Mikhail Belkin, and Siyuan Ma. "On exponential convergence of sgd in non-convex over-parametrized learning". In: *arXiv:1811.02564* (2018).

[24] Matthew J. S. Beach et al. "Making trotters sprint: A variational imaginary time ansatz for quantum many-body systems". In: *Phys. Rev. B* 100 (9 2019), p. 094434. DOI: 10.1103/PhysRevB.100.094434. URL: https://link.aps.org/doi/10.1103/PhysRevB.100.094434.

[25] Marc G. Bellemare et al. "Autonomous navigation of stratospheric balloons using reinforcement learning". In: *Nature* 588.7836 (2020), pp. 77–82. DOI: 10.1038/s41586-020-2939-8. URL: https://doi.org/10.1038/s41586-020-2939-8.

[26] Marcello Benedetti, Mattia Fiorentini, and Michael Lubasch. "Hardware-efficient variational quantum algorithms for time evolution". In: *Phys. Rev. Res.* 3 (3 July 2021), p. 033083.

[27] Marcello Benedetti et al. "Parameterized quantum circuits as machine learning models". In: *Quantum Science and Technology* 4.4 (2019), p. 043001.

[28] Dominic W Berry et al. "Efficient quantum algorithms for simulating sparse Hamiltonians". In: *Commun. Math. Phys.* 270.2 (2007), pp. 359–371.

[29] MV Berry. "Transitionless quantum driving". In: *Journal of Physics A: Mathematical and Theoretical* 42.36 (2009), p. 365303. URL: http://iopscience.iop.org/article/10.1088/1751-8113/42/36/365303/meta.

[30] Dimitri Bertsekas. *Reinforcement learning and optimal control*. Athena Scientific, 2019.

[31] Craig J Bester, Steven D James, and George D Konidaris. "Multi-pass q-networks for deep reinforcement learning with parameterised action spaces". In: *arXiv preprint arXiv:1905.04388* (2019).

[32] Jacob Biamonte et al. "Quantum machine learning". In: *Nature* 549.7671 (2017), pp. 195–202.

[33] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

[34] Rainer Blatt and Christian F Roos. "Quantum simulations with trapped ions". In: *Nature Physics* 8.4 (2012), p. 277. URL: https://www.nature.com/articles/nphys2252.

[35] Immanuel Bloch, Jean Dalibard, and Wilhelm Zwerger. "Many-body physics with ultracold gases". In: *Rev. Mod. Phys.* 80 (3 2008), pp. 885–964. DOI: 10.1103/RevModPhys.80.885. URL: https://link.aps.org/doi/10.1103/RevModPhys.80.885.

[36] Annabelle Bohrdt et al. "Classifying snapshots of the doped Hubbard model with machine learning". In: *Nature Physics* 15.9 (2019), pp. 921–924.

[37] Adrien Bolens and Markus Heyl. "Reinforcement Learning for Digital Quantum Simulation". In: *arXiv preprint arXiv:2006.16269* (2020). URL: https://arxiv.org/abs/2006.16269.

[38] R Botet and R Jullien. "Large-size critical behavior of infinitely coordinated systems". In: *Physical Review B* 28.7 (1983), p. 3955.

[39] Léon Bottou, Frank E Curtis, and Jorge Nocedal. "Optimization methods for large-scale machine learning". In: *Siam Review* 60.2 (2018), pp. 223–311.

[40] James Bradbury et al. *JAX: composable transformations of Python+NumPy programs*. Version 0.2.5. 2018. URL: http://github.com/google/jax.

[41] Sergey Bravyi et al. "Hybrid quantum-classical algorithms for approximate graph coloring". In: *arXiv preprint arXiv:2011.13420* (2020).

[42] David H Brookes et al. "A view of Estimation of Distribution Algorithms through the lens of Expectation-Maximization". In: *arXiv:1905.10474* (2019).

[43] Cameron B Browne et al. "A survey of monte carlo tree search methods". In: *IEEE Transactions on Computational Intelligence and AI in games* 4.1 (2012), pp. 1–43.

[44] Marin Bukov. "Reinforcement learning for autonomous preparation of Floquet-engineered states: Inverting the quantum Kapitza oscillator". In: *Physical Review B* 98.22 (2018), p. 224305.

[45] Marin Bukov. "Reinforcement learning for autonomous preparation of Floquet-engineered states: Inverting the quantum Kapitza oscillator". In: *Phys. Rev. B* 98 (22 Dec. 2018), p. 224305. DOI: 10.1103/PhysRevB.98.224305. URL: https://link.aps.org/doi/10.1103/PhysRevB.98.224305.

[46] Marin Bukov, Dries Sels, and Anatoli Polkovnikov. "Geometric Speed Limit of Accessible Many-Body State Preparation". In: *Phys. Rev. X* 9 (1 2019), p. 011034. DOI: 10.1103/PhysRevX.9.011034. URL: https://link.aps.org/doi/10.1103/PhysRevX.9.011034.

[47] Marin Bukov et al. "Broken symmetry in a correlated quantum control landscape". In: *Physical Review A* (2018).

[48] Marin Bukov et al. "Reinforcement Learning in Different Phases of Quantum Control". In: *Physical Review X* 8.3 (2018), p. 031086.

[49] Marin Bukov et al. "Reinforcement Learning in Different Phases of Quantum Control". In: *Phys. Rev. X* 8 (3 Sept. 2018), p. 031086. DOI: 10.1103/PhysRevX.8.031086. URL: https://link.aps.org/doi/10.1103/PhysRevX.8.031086.

[50] Marin Bukov et al. "Reinforcement learning in different phases of quantum control". In: *Physical Review X* 8.3 (2018), p. 031086. DOI: 10.1103/PhysRevX.8.031086. URL: https://link.aps.org/doi/10.1103/PhysRevX.8.031086.

[51] Han Cai, Ligeng Zhu, and Song Han. "Proxylessnas: Direct neural architecture search on target task and hardware". In: *arXiv preprint arXiv:1812.00332* (2018).

[52] A Callison and N Chancellor. "Quantum Annealing Initialization of the Quantum Approximate Optimization Algorithm". In: *arXiv:1909.04223* (2019).

[53] Tommaso Caneva, Tommaso Calarco, and Simone Montangero. "Chopped random-basis quantum optimization". In: *Phys. Rev. A* 84 (2 2011), p. 022326. DOI: 10.1103/PhysRevA.84.022326. URL: https://link.aps.org/doi/10.1103/PhysRevA.84.022326.

[54] Giuseppe Carleo and Matthias Troyer. "Solving the quantum many-body problem with artificial neural networks". In: *Science* 355.6325 (2017), pp. 602–606.

[55] Giuseppe Carleo et al. "Machine learning and the physical sciences". In: *Rev. Mod. Phys.* 91 (4 2019), p. 045002. DOI: 10.1103/RevModPhys.91.045002. URL: https://link.aps.org/doi/10.1103/RevModPhys.91.045002.

[56] Juan Carrasquilla. "Machine learning for quantum matter". In: *Advances in Physics: X* 5.1 (2020), p. 1797528. DOI: 10.1080/23746149.2020.1797528. URL: https://doi.org/10.1080/23746149.2020.1797528.

[57] Juan Carrasquilla and Roger G Melko. "Machine learning phases of matter". In: *Nature Physics* 13.5 (2017), pp. 431–434.

[58] Francesco Casola, Toeno van der Sar, and Amir Yacoby. "Probing condensed matter physics with magnetometry based on nitrogen-vacancy centres in diamond". In: *Nature Reviews Materials* 3.1 (2018), pp. 1–13. URL: https://www.nature.com/articles/natrevmats201788.

[59] M Cerezo et al. "Challenges and opportunities in quantum machine learning". In: *Nature Computational Science* 2.9 (2022), pp. 567–576.

[60] M Cerezo et al. "Cost-function-dependent barren plateaus in shallow quantum neural networks". In: *arXiv preprint arXiv:2001.00550* (2020). URL: https://arxiv.org/abs/2001.00550.

[61] Marco Cerezo et al. "Cost-Function-Dependent Barren Plateaus in Shallow Quantum Neural Networks". In: *Nature Communications* 12.1 (2021), p. 1791.

[62] Marco Cerezo et al. "Variational quantum algorithms". In: *Nature Reviews Physics* 3.9 (2021), pp. 625–644.

[63] Marco Cerezo et al. "Variational quantum algorithms". In: *Nature Reviews Physics* 3.9 (2021), pp. 625–644.

[64] P. Chandarana et al. "Digitized-counterdiabatic quantum approximate optimization algorithm". In: *arXiv preprint arXiv:2107.02789v2* (July 2021). arXiv: 2107.02789v2 [quant-ph]. URL: http://arxiv.org/abs/2107.02789v2.

[65] Chunlin Chen et al. "Fidelity-based probabilistic Q-learning for control of quantum systems". In: *IEEE transactions on neural networks and learning systems* 25.5 (2013), pp. 920–933.

[66] Jun-Jie Chen and Ming Xue. "Manipulation of spin dynamics by deep reinforcement learning agent". In: *arXiv preprint arXiv:1901.08748* (2019).

[67] Yu-Qin Chen et al. "Optimizing Quantum Annealing Schedules: From Monte Carlo Tree Search to QuantumZero". In: *arXiv preprint arXiv:2004.02836v2* (Apr. 2020). arXiv: 2004.02836v2 [quant-ph]. URL: http://arxiv.org/abs/2004.02836v2.

[68] Wei Chen, Kazuo Hida, and B. C. Sanctuary. "Ground-state phase diagram of $S = 1$ XXZ chains with uniaxial single-ion-type anisotropy". In: *Phys. Rev. B* 67 (10 Mar. 2003), p. 104401. URL: https://link.aps.org/doi/10.1103/PhysRevB.67.104401.

[69] Ziang Chen, Yingzhou Li, and Jianfeng Lu. "On the Global Convergence of Randomized Coordinate Gradient Descent for Nonconvex Optimization". In: *SIAM Journal on Optimization* 33.2 (2023), pp. 713–738.

[70] Kyunghyun Cho et al. "Learning phrase representations using RNN encoder-decoder for statistical machine translation". In: *arXiv preprint arXiv:1406.1078* (2014).

[71] Krzysztof Choromanski et al. *Rethinking Attention with Performers*. 2020. arXiv: 2009.14794 [cs.LG].

[72] Po-Wei Chou, Daniel Maturana, and Sebastian A. Scherer. "Improving Stochastic Policy Gradients in Continuous Control with Deep Reinforcement Learning using the Beta Distribution". In: *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*. Ed. by Doina Precup and Yee Whye Teh. Vol. 70. Proceedings of Machine Learning Research. PMLR, 2017, pp. 834–843. URL: http://proceedings.mlr.press/v70/chou17a.html.

[73] Simona Colabrese et al. "Flow navigation by smart microswimmers via reinforcement learning". In: *Physical review letters* 118.15 (2017), p. 158004.

[74] Rémi Coulom. "Efficient selectivity and backup operators in Monte-Carlo tree search". In: *International conference on computers and games*. Springer. 2006, pp. 72–83.

[75] Gavin E Crooks. "Gradients of parameterized quantum gates using the parameter-shift rule and gate decomposition". In: *arXiv:1905.13311* (2019).

[76] Mogens Dalgaard, Felix Motzoi, and Jacob Sherson. "Predicting quantum dynamical cost landscapes with deep learning". In: *Physical Review A* 105.1 (2022), p. 012402.

[77] Mogens Dalgaard et al. "Global optimization of quantum dynamics with AlphaZero deep exploration". In: *arXiv preprint arXiv:1907.05672* (2019).

[78] Mogens Dalgaard et al. "Global optimization of quantum dynamics with AlphaZero deep exploration". In: *npj Quantum Information* 6.1 (Jan. 2020). DOI: 10.1038/s41534-019-0241-0. URL: https://doi.org/10.1038%2Fs41534-019-0241-0.

[79] Mogens Dalgaard et al. "Hessian-based optimization of constrained quantum control". In: *Physical Review A* 102.4 (2020), p. 042612.

[80] Nikesh S Dattani and Nathaniel Bryans. "Quantum factorization of 56153 with only 4 qubits". In: *arXiv preprint arXiv:1411.6758* (2014).

[81] Emily J Davis et al. "Protecting spin coherence in a tunable heisenberg model". In: *Physical Review Letters* 125.6 (2020), p. 060402.

[82] Alexandre GR Day et al. "Glassy phase of optimal quantum control". In: *Physical review letters* 122.2 (2019), p. 020601.

[83] Pieter-Tjerk De Boer et al. "A tutorial on the cross-entropy method". In: *Annals of operations research* 134.1 (2005), pp. 19–67.

[84] Olivier Delalleau et al. "Discrete and continuous action representation for practical rl in video games". In: *arXiv preprint arXiv:1912.11077* (2019).

[85] Mustafa Demirplak and Stuart A Rice. "Adiabatic population transfer with control fields". In: *The Journal of Physical Chemistry A* 107.46 (2003), pp. 9937–9945. URL: http://pubs.acs.org/doi/abs/10.1021/jp030708a.

[86] Mustafa Demirplak and Stuart A Rice. "Assisted adiabatic passage revisited". In: *The Journal of Physical Chemistry B* 109.14 (2005), pp. 6838–6844. URL: http://pubs.acs.org/doi/abs/10.1021/jp040647w.

[87] Jacob Devlin et al. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". In: *arXiv preprint arXiv:1810.04805* (2018).

[88] Michel H Devoret and Robert J Schoelkopf. "Superconducting circuits for quantum information: an outlook". In: *Science* 339.6124 (2013), pp. 1169–1174. URL: http://science.sciencemag.org/content/339/6124/1169.

[89] Joshua V Dillon et al. "Tensorflow distributions". In: *arXiv:1711.10604* (2017).

[90] Yongcheng Ding et al. "Breaking Adiabatic Quantum Control with Deep Learning". In: *arXiv preprint arXiv:2009.04297* (2020). URL: https://arxiv.org/abs/2009.04297.

[91] Laurent Dinh, David Krueger, and Yoshua Bengio. "NICE: Non-linear Independent Components Estimation". In: *arXiv preprint arXiv:1410.8516* (2014).

[92] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. "Density estimation using Real NVP". In: *arXiv preprint arXiv:1605.08803* (2016).

[93] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. "Density estimation using real nvp". In: *arXiv preprint arXiv:1605.08803* (2016).

[94] Pablo Dıez-Valle, Diego Porras, and Juan José Garcıa-Ripoll. "Quantum variational optimization: The role of entanglement and problem hardness". In: *Phys. Rev. A* 104 (6 Dec. 2021), p. 062426. DOI: 10.1103/PhysRevA.104.062426. URL: https://link.aps.org/doi/10.1103/PhysRevA.104.062426.

[95] Carl Doersch. "Tutorial on Variational Autoencoders". In: *arXiv preprint arXiv:1606.05908* (2016).

[96] Marcus W Doherty et al. "The nitrogen-vacancy colour centre in diamond". In: *Physics Reports* 528.1 (2013), pp. 1–45. URL: https://www.sciencedirect.com/science/article/abs/pii/S0370157313000562.

[97] Yulong Dong et al. "Robust Control Optimization for Quantum Approximate Optimization Algorithm". In: *arXiv preprint arXiv:1911.00789* (2019).

[98] Vedran Dunjko and Hans J Briegel. "Machine learning & artificial intelligence in the quantum domain: a review of recent progress". In: *Reports on Progress in Physics* 81.7 (2018), p. 074001. URL: https://iopscience.iop.org/article/10.1088/1361-6633/aab406.

[99] Thomas Elsken, Jan Hendrik Metzen, and Frank Hutter. "Neural Architecture Search: A Survey". In: *arXiv preprint arXiv:1808.05377v3* (Aug. 2018). Journal of Machine Learning Research 20 (2019) 1-21. arXiv: 1808.05377v3 [stat.ML]. URL: http://arxiv.org/abs/1808.05377v3.

[100] Suguru Endo et al. "Hybrid quantum-classical algorithms and quantum error mitigation". In: *Journal of the Physical Society of Japan* 90.3 (2021), p. 032001.

[101] Zhou Fan et al. "Hybrid Actor-Critic Reinforcement Learning in Parameterized Action Space". In: *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*. Ed. by Sarit Kraus. ijcai.org, 2019, pp. 2279–2285. DOI: 10.24963/ijcai.2019/316. URL: https://doi.org/10.24963/ijcai.2019/316.

[102] Edward Farhi, Jeffrey Goldstone, and Sam Gutmann. "A quantum approximate optimization algorithm". In: *arXiv preprint arXiv:1411.4028* (2014).

[103] Edward Farhi, Jeffrey Goldstone, and Sam Gutmann. "The Quantum Approximate Optimization Algorithm and the Sherrington-Kirkpatrick Model at Infinite Size". In: *arXiv preprint arXiv:1910.08187* (2019).

[104] Benjamin Fehrman, Benjamin Gess, and Arnulf Jentzen. "Convergence rates for the stochastic gradient descent method for non-convex objective functions". In: *The Journal of Machine Learning Research* 21.1 (2020), pp. 5354–5401.

[105] Thomas Fösel et al. "Efficient cavity control with SNAP gates". In: *arXiv preprint arXiv:2004.14256* (2020). URL: https://arxiv.org/abs/2004.14256.

[106] Thomas Fösel et al. "Quantum circuit optimization with deep reinforcement learning". In: *arXiv preprint arXiv:2103.07585* (2021).

[107] Thomas Fösel et al. "Reinforcement Learning with Neural Networks for Quantum Feedback". In: *Physical Review X* 8.3 (2018), p. 031084.

[108] Fuchang Gao and Lixing Han. "Implementing the Nelder-Mead simplex algorithm with adaptive parameters". In: *Computational Optimization and Applications* 51.1 (2012), pp. 259–277.

[109] Pan Gao et al. "Quantum gradient algorithm for general polynomials". In: *Physical Review A* 103.4 (2021), p. 042403.

[110] Artur Garcia-Saez and Jordi Riu. "Quantum Observables for continuous control of the Quantum Approximate Optimization Algorithm via Reinforcement Learning". In: *arXiv preprint arXiv:1911.09682* (2019). URL: https://arxiv.org/abs/1911.09682.

[111] Mathieu Germain et al. "MADE: Masked Autoencoder for Distribution Estimation". In: *Proceedings of the 32nd International Conference on Machine Learning*. 2015, pp. 881–889.

[112] Mathieu Germain et al. "Made: Masked autoencoder for distribution estimation". In: *International Conference on Machine Learning*. 2015, pp. 881–889.

[113] András Gilyén, Srinivasan Arunachalam, and Nathan Wiebe. "Optimizing quantum optimization algorithms via faster quantum gradient computation". In: *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM. 2019, pp. 1425–1444.

[114] Alessandro Giovagnoli, Yunpu Ma, and Volker Tresp. "QNEAT: Natural Evolution of Variational Quantum Circuit Architecture". In: *arXiv preprint arXiv:2304.06981v1* (Apr. 2023). arXiv: 2304.06981v1 [quant-ph]. URL: http://arxiv.org/abs/2304.06981v1.

[115] Fred Glover, Gary Kochenberger, and Yu Du. "A Tutorial on Formulating and Using QUBO Models". In: *arXiv preprint arXiv:1811.11538v6* (Nov. 2018). arXiv: 1811.11538v6 [cs.DS]. URL: http://arxiv.org/abs/1811.11538v6.

[116] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. URL: http://www.deeplearningbook.org.

[117] Ian Goodfellow et al. "Generative Adversarial Nets". In: *Advances in Neural Information Processing Systems* (2014), pp. 2672–2680.

[118] Edward Grant et al. "An initialization strategy for addressing barren plateaus in parametrized quantum circuits". In: *Quantum* 3 (2019), p. 214. DOI: 10.22331/q-2019-12-09-214.

[119] Alex Graves. "Generating sequences with recurrent neural networks". In: *arXiv preprint arXiv:1308.0850* (2013).

[120] Evan Greensmith, Peter L Bartlett, and Jonathan Baxter. "Variance reduction techniques for gradient estimates in reinforcement learning". In: *Journal of Machine Learning Research* 5.Nov (2004), pp. 1471–1530.

[121] Gian Giacomo Guerreschi and Mikhail Smelyanskiy. "Practical optimization for hybrid quantum-classical algorithms". In: *arXiv preprint arXiv:1701.01450* (2017).

[122] David Guéry-Odelin et al. "Shortcuts to adiabaticity: Concepts, methods, and applications". In: *Reviews of Modern Physics* 91.4 (2019), p. 045001.

[123] Ishaan Gulrajani et al. "Improved Training of Wasserstein GANs". In: *arXiv preprint arXiv:1704.00028* (2017).

[124] Xiaoxiao Guo et al. "Deep learning for real-time Atari game play using offline Monte-Carlo tree search planning". In: *Advances in neural information processing systems*. 2014, pp. 3338–3346.

[125] Tuomas Haarnoja et al. "Reinforcement Learning with Deep Energy-Based Policies". In: *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*. Ed. by Doina Precup and Yee Whye Teh. Vol. 70. Proceedings of Machine Learning Research. PMLR, 2017, pp. 1352–1361. URL: http://proceedings.mlr.press/v70/haarnoja17a.html.

[126] Tuomas Haarnoja et al. "Soft actor-critic algorithms and applications". In: *arXiv preprint arXiv:1812.05905* (2018).

[127] Tuomas Haarnoja et al. "Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor". In: *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*. Ed. by Jennifer G. Dy and Andreas Krause. Vol. 80. Proceedings of Machine Learning Research. PMLR, 2018, pp. 1856–1865. URL: http://proceedings.mlr.press/v80/haarnoja18b.html.

[128] Stuart Hadfield. "Quantum algorithms for scientific computing and approximate optimization". In: *arXiv preprint arXiv:1805.03265* (2018).

[129] Stuart Hadfield et al. "From the quantum approximate optimization algorithm to a quantum alternating operator ansatz". In: *Algorithms* 12.2 (2019), p. 34.

[130] Hartmut Häffner, Christian F Roos, and Rainer Blatt. "Quantum computing with trapped ions". In: *Physics reports* 469.4 (2008), pp. 155–203. URL: https://www.sciencedirect.com/science/article/pii/S0370157308003463?via%3Dihub.

[131] James Halverson, Brent Nelson, and Fabian Ruehle. "Branes with brains: exploring string vacua with deep reinforcement learning". In: *Journal of High Energy Physics* 2019.6 (2019), p. 3.

[132] Nikolaus Hansen and Andreas Ostermeier. "Adapting arbitrary normal mutation distributions in evolution strategies: The covariance matrix adaptation". In: *Proceedings of IEEE International Conference on Evolutionary Computation*. IEEE. 1996, pp. 312–317.

[133] Nikolaus Hansen and Andreas Ostermeier. "Completely derandomized self-adaptation in evolution strategies". In: *Evolutionary computation* 9.2 (2001), pp. 159–195.

[134] Charles R. Harris et al. "Array programming with NumPy". In: *Nature* 585.7825 (Sept. 2020), pp. 357–362. DOI: 10.1038/s41586-020-2649-2. URL: https://doi.org/10.1038/s41586-020-2649-2.

[135] Andreas Hartmann and Wolfgang Lechner. "Rapid counter-diabatic sweeps in lattice gauge adiabatic quantum computing". In: *New Journal of Physics* 21.4 (2019), p. 043025. DOI: 10.1088/1367-2630/ab14a0. URL: https://doi.org/10.1088%5C%2F1367-2630%5C%2Fab14a0.

[136] Takuya Hatomura. "Shortcuts to adiabaticity in the infinite-range Ising model by mean-field counter-diabatic driving". In: *Journal of the Physical Society of Japan* 86.9 (2017), p. 094002. URL: https://journals.jps.jp/doi/10.7566/JPSJ.86.094002.

[137] Matthew J. Hausknecht and Peter Stone. "Deep Reinforcement Learning in Parameterized Action Space". In: *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*. Ed. by Yoshua Bengio and Yann LeCun. 2016. URL: http://arxiv.org/abs/1511.04143.

[138] N. N. Hegade et al. "Portfolio Optimization with Digitized-Counterdiabatic Quantum Algorithms". In: *arXiv preprint arXiv:2112.08347v1* (Dec. 2021). arXiv: 2112.08347v1 [quant-ph]. URL: http://arxiv.org/abs/2112.08347v1.

[139] Narendra N. Hegade, Xi Chen, and Enrique Solano. "Digitized-Counterdiabatic Quantum Optimization". In: *arXiv preprint arXiv:2201.00790v1* (Jan. 2022). arXiv: 2201.00790v1 [quant-ph]. URL: http://arxiv.org/abs/2201.00790v1.

[140] Narendra N. Hegade et al. "Shortcuts to Adiabaticity in Digitized Adiabatic Quantum Computing". In: *Physical Review Applied* 15.2 (Feb. 2021). DOI: 10.1103/physrevapplied.15.024038. URL: https://doi.org/10.1103%2Fphysrevapplied.15.024038.

[141] Wen Wei Ho and Timothy H Hsieh. "Efficient variational simulation of non-trivial quantum states". In: *SciPost Phys* 6 (2019), p. 029.

[142] Sepp Hochreiter and Jürgen Schmidhuber. "Long short-term memory". In: *Neural computation* 9.8 (1997), pp. 1735–1780.

[143] John H. Holland. "Genetic algorithms". In: *Scientific American* 267.1 (1992), pp. 66–72.

[144] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. "Multilayer feedforward networks are universal approximators". In: *Neural Networks* 2.5 (1989), pp. 359–366. DOI: 10.1016/0893-6080(89)90020-8.

[145] Chin-Wei Huang et al. "Neural autoregressive flows". In: *International Conference on Machine Learning*. PMLR. 2018, pp. 2078–2087.

[146] Patrick Huembeli and Alexandre Dauphin. "Characterizing the loss landscape of variational quantum circuits". In: *Quantum Science and Technology* 6.2 (), p. 025011. DOI: 10.1088/2058-9565/abdbc9. URL: https://doi.org/10.1088/2058-9565/abdbc9.

[147] Raban Iten et al. "Discovering physical concepts with neural networks". In: *Physical Review Letters* 124.1 (2020), p. 010508.

[148] Chi Jin et al. "On nonconvex optimization for machine learning: Gradients, stochasticity, and saddle points". In: *Journal of the ACM (JACM)* 68.2 (2021), pp. 1–29.

[149] J Robert Johansson, Paul D Nation, and Franco Nori. "QuTiP 2: A Python framework for the dynamics of open quantum systems". In: *Computer Physics Communications* 184.4 (2013), pp. 1234–1240.

[150] J Robert Johansson, PD Nation, and Franco Nori. "QuTiP: An open-source Python framework for the dynamics of open quantum systems". In: *Computer Physics Communications* 183.8 (2012), pp. 1760–1772.

[151] Stephen P Jordan. "Fast quantum algorithm for numerical gradient estimation". In: *Physical Review Letters* 95.5 (2005), p. 050501.

[152] John Jumper et al. "High Accuracy Protein Structure Prediction Using Deep Learning". In: *In preparation* (2020).

[153] Velimir Jurdjevic and Héctor J Sussmann. "Control systems on Lie groups". In: *Journal of Differential equations* 12.2 (1972), pp. 313–329. URL: http://www.sciencedirect.com/science/article/pii/0022039672900356.

[154] Leslie P. Kaelbling, Michael L. Littman, and Anthony R. Cassandra. "Planning and Acting in Partially Observable Stochastic Domains". In: *Artificial Intelligence* 101 (1998), pp. 99–134.

[155] Sham M Kakade. "A natural policy gradient". In: *Advances in neural information processing systems*. 2002, pp. 1531–1538.

[156] Abhinav Kandala et al. "Hardware-efficient variational quantum eigensolver for small molecules and quantum magnets". In: *Nature* 549.7671 (2017), pp. 242–246. DOI: 10.1038/nature23879. URL: https://doi.org/10.1038/nature23879.

[157] George Em Karniadakis et al. "Physics-informed machine learning". In: *Nature Reviews Physics* 3.6 (2021), pp. 422–440. DOI: 10.1038/s42254-021-00314-5.

[158] John Kennedy and Russell Eberhart. "Particle swarm optimization". In: *Proceedings of ICNN'95 - International Conference on Neural Networks*. Vol. 4. 1995, pp. 1942–1948.

[159] Sami Khairy et al. "Learning to Optimize Variational Quantum Circuits to Solve Combinatorial Problems". In: *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*. AAAI Press, 2020, pp. 2367–2375. URL: https://aaai.org/ojs/index.php/AAAI/article/view/5616.

[160] Sami Khairy et al. "Reinforcement-learning-based variational quantum circuits optimization for combinatorial problems". In: *arXiv preprint arXiv:1911.04574* (2019). URL: https://arxiv.org/abs/1911.04574.

[161] Navin Khaneja et al. "Optimal control of coupled spin dynamics: design of NMR pulse sequences by gradient ascent algorithms". In: *Journal of magnetic resonance* 172.2 (2005), pp. 296–305.

[162] Hyungwon Kim and David A Huse. "Ballistic spreading of entanglement in a diffusive nonintegrable system". In: *Physical review letters* 111.12 (2013), p. 127205.

[163] Diederik P Kingma and Jimmy Ba. "Adam: A method for stochastic optimization". In: *arXiv preprint arXiv:1412.6980* (2014).

[164] Diederik P. Kingma, Tim Salimans, and Max Welling. "Improving Variational Inference with Inverse Autoregressive Flow". In: *arXiv:1606.04934* (2016).

[165] Diederik P. Kingma and Max Welling. "Auto-Encoding Variational Bayes". In: *arXiv preprint arXiv:1312.6114* (2013).

[166] Durk P Kingma et al. "Improved variational inference with inverse autoregressive flow". In: *Advances in neural information processing systems*. 2016, pp. 4743–4751.

[167] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. "Optimization by simulated annealing". In: *Science* 220.4598 (1983), pp. 671–680.

[168] Nikita Kitaev, Lukasz Kaiser, and Anselm Levskaya. "Reformer: The Efficient Transformer". In: *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. URL: https://openreview.net/forum?id=rkgNKkHtvB.

[169] Bobby Kleinberg, Yuanzhi Li, and Yang Yuan. "An alternative view: When does SGD escape local minima?" In: *International conference on machine learning*. PMLR. 2018, pp. 2698–2707.

[170] Taehee Ko and Xiantao Li. "A Local Convergence Theory for the Stochastic Gradient Descent Method in Non-Convex Optimization With Non-isolated Local Minima". In: *Journal of Machine Learning* 2.2 (2023), pp. 138–160.

[171] Gary Kochenberger et al. "The unconstrained binary quadratic programming problem: a survey". In: *Journal of Combinatorial Optimization* 28.1 (July 2014), pp. 58–81. ISSN: 1573-2886. DOI: 10.1007/s10878-014-9734-0. URL: https://doi.org/10.1007/s10878-014-9734-0.

[172] Levente Kocsis and Csaba Szepesvári. "Bandit based Monte-Carlo Planning". In: *European conference on machine learning*. 2006, pp. 282–293.

[173] Michael Kolodrubetz et al. "Geometry and non-adiabatic response in quantum and classical systems". In: *Physics Reports* 697 (2017), pp. 1–87. URL: https://www.sciencedirect.com/science/article/abs/pii/S0370157317301989.

[174] Vijay R Konda and John N Tsitsiklis. "Actor-Critic algorithms". In: *Advances in neural information processing systems*. 2000, pp. 1008–1014.

[175] Vijay R. Konda and John N. Tsitsiklis. "Actor-Critic Algorithms". In: *Advances in Neural Information Processing Systems*. 2000.

[176] Robert L Kosut, Matthew D Grace, and Constantin Brif. "Robust control of quantum gates via sequential convex programming". In: *Physical Review A* 88.5 (2013), p. 052326.

[177] Korbinian Kottmann et al. "Unsupervised phase discovery with deep anomaly detection". In: *arXiv preprint arXiv:2003.09905* (2020).

[178] Dieter Kraft et al. "A software package for sequential quadratic programming". In: (1988).

[179] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. "ImageNet Classification with Deep Convolutional Neural Networks". In: *Advances in Neural Information Processing Systems 25*. 2012.

[180] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. "ImageNet Classification with Deep Convolutional Neural Networks". In: *Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012. Proceedings of a meeting held December 3-6, 2012, Lake Tahoe, Nevada, United States*. Ed. by Peter L. Bartlett et al. 2012, pp. 1106–1114. URL: https://proceedings.neurips.cc/paper/2012/hash/c399862d3b9d6b76c8436e924a68c45b-Abstract.html.

[181] Tejas D. Kulkarni et al. "Hierarchical Deep Reinforcement Learning: Integrating Temporal Abstraction and Intrinsic Motivation". In: *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*. Ed. by Daniel D. Lee et al. 2016, pp. 3675–3683. URL: https://proceedings.neurips.cc/paper/2016/hash/f442d33fa06832082290ad8544a8da27-Abstract.html.

[182] En-Jui Kuo, Yao-Lung L. Fang, and Samuel Yen-Chi Chen. "Quantum Architecture Search via Deep Reinforcement Learning". In: *arXiv preprint arXiv:2104.07715v1* (Apr. 2021). arXiv: 2104.07715v1 [quant-ph]. URL: http://arxiv.org/abs/2104.07715v1.

[183] Harold J Kushner and G George Yin. "Applications to learning, state dependent noise, and queueing". In: *Stochastic Approximation Algorithms and Applications* (1997), pp. 25–46.

[184] Nathan Lacroix et al. "Improving the Performance of Deep Quantum Optimization Algorithms with Continuous Gate Sets". In: *arXiv:2005.05275* (2020). URL: https://arxiv.org/abs/2005.05275.

[185] A Langari, F Pollmann, and M Siahatgar. "Ground-state fidelity of the spin-1 Heisenberg chain with single ion anisotropy: quantum renormalization group and exact diagonalization approaches". In: *Journal of Physics: Condensed Matter* 25.40 (Sept. 2013), p. 406002. URL: https://doi.org/10.1088/0953-8984/25/40/406002.

[186] Martin Larocca et al. "Theory of overparametrization in quantum neural networks". In: *arXiv preprint arXiv:2109.11676* (2021).

[187] Author1 LastName1 and Author2 LastName2. "Learning Fast Approximations of Sparse Coding". In: *Journal or Conference Name* Volume Number (2021), Page Numbers. DOI: DOI.

[188] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. "Deep Learning". In: *Nature* 521.7553 (2015), pp. 436–444. DOI: 10.1038/nature14539.

[189] Yin Tat Lee and Aaron Sidford. "Efficient accelerated coordinate descent methods and faster algorithms for solving linear systems". In: *2013 ieee 54th annual symposium on foundations of computer science*. IEEE. 2013, pp. 147–156.

[190] Jiaqi Leng et al. "Differentiable Analog Quantum Computing for Optimization and Control". In: *arXiv preprint arXiv:2210.15812* (2022).

[191] Sergey Levine. "Reinforcement learning and control as probabilistic inference: Tutorial and review". In: *arXiv preprint arXiv:1805.00909* (2018).

[192] Sergey Levine et al. "End-to-End Training of Deep Visuomotor Policies". In: *Journal of Machine Learning Research* 17.39 (2016), pp. 1–40.

[193] Maciej Lewenstein et al. "Ultracold atomic gases in optical lattices: mimicking condensed matter physics and beyond". In: *Advances In Physics* 56.2 (2007), pp. 243–379. URL: https://www.tandfonline.com/doi/abs/10.1080/00018730701223200.

[194] Li Li et al. "Quantum optimization with a novel gibbs objective function and ansatz architecture search". In: *Physical Review Research* 2.2 (2020), p. 023074. DOI: 10.1103/PhysRevResearch.2.023074.

[195] Timothy P. Lillicrap et al. "Continuous control with deep reinforcement learning". In: *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*. Ed. by Yoshua Bengio and Yann LeCun. 2016. URL: http://arxiv.org/abs/1509.02971.

[196] Harry J Lipkin, N Meshkov, and AJ Glick. "Validity of many-body approximation methods for a solvable model:(I). Exact solutions and perturbation theory". In: *Nuclear Physics* 62.2 (1965), pp. 188–198.

[197] Chaoyue Liu, Libin Zhu, and Mikhail Belkin. "Loss landscapes and optimization in over-parameterized non-linear systems and neural networks". In: *Applied and Computational Harmonic Analysis* 59 (2022), pp. 85–116.

[198] Hanxiao Liu, Karen Simonyan, and Yiming Yang. "DARTS: Differentiable Architecture Search". In: *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. URL: https://openreview.net/forum?id=S1eYHoC5FX.

[199] Seth Lloyd. "Almost any quantum logic gate is universal". In: *Physical Review Letters* 75.2 (1995), p. 346.

[200] Seth Lloyd. "Quantum approximate optimization is computationally universal". In: *arXiv preprint arXiv:1812.11075* (2018).

[201] Guang Hao Low and Isaac L. Chuang. "Optimal Hamiltonian Simulation by Quantum Signal Processing". In: *Phys. Rev. Lett.* 118 (2017), p. 010501.

[202] Denghui Lu et al. "86 PFLOPS Deep Potential Molecular Dynamics simulation of 100 million atoms with ab initio accuracy". In: *arXiv preprint arXiv:2004.11658* (2020).

[203] Andrew Lucas. "Ising formulations of many NP problems". In: *Frontiers in physics* 2 (2014), p. 5.

[204] Alicia B Magann et al. "Digital quantum simulation of molecular dynamics and control". In: *arXiv preprint arXiv:2002.12497* (2020).

[205] Alicia B Magann et al. "From pulses to circuits and back again: A quantum optimal control perspective on variational quantum algorithms". In: *arXiv preprint arXiv:2009.06702* (2020).

[206] Horia Mania, Aurelia Guy, and Benjamin Recht. "Simple random search provides a competitive approach to reinforcement learning". In: *arXiv preprint arXiv:1803.07055v1* (Mar. 2018). arXiv: 1803.07055v1 [cs.LG]. URL: http://arxiv.org/abs/1803.07055v1.

[207] Warwick Masson, Pravesh Ranchod, and George Dimitri Konidaris. "Reinforcement Learning with Parameterized Actions". In: *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA*. Ed. by Dale Schuurmans and Michael P. Wellman. AAAI Press, 2016, pp. 1934–1940. URL: http://www.aaai.org/ocs/index.php/AAAI/AAAI16/paper/view/11981.

[208] Shumpei Masuda and Katsuhiro Nakamura. "Fast-forward of adiabatic dynamics in quantum mechanics". In: *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*. The Royal Society. 2009, rspa20090446. URL: http://rspa.royalsocietypublishing.org/content/466/2116/1135.

[209] Gabriel Matos, Sonika Johri, and Zlatko Papić. "Quantifying the Efficiency of State Preparation via Quantum Variational Eigensolvers". In: *PRX Quantum* 2.1 (2021). DOI: 10.1103/prxquantum.2.010309. URL: https://doi.org/10.1103%2Fprxquantum.2.010309.

[210] Toshiki Matsumine, Toshiaki Koike-Akino, and Ye Wang. "Channel decoding with quantum approximate optimization algorithm". In: *2019 IEEE International Symposium on Information Theory (ISIT)*. IEEE. 2019, pp. 2574–2578.

[211] Nicholas J. Mayhall et al. "Overlap-ADAPT-VQE: Practical quantum chemistry on a quantum device". In: *Nature* 600.7887 (2021), pp. 429–434.

[212] Jarrod R McClean et al. "Barren plateaus in quantum neural network training landscapes". In: *Nature Communications* 9.1 (2018), p. 4812.

[213] Jarrod R McClean et al. "The theory of variational hybrid quantum-classical algorithms". In: *New J. Phys.* 18.2 (2016), p. 023023.

[214] Jarrod R McClean et al. "The theory of variational hybrid quantum-classical algorithms". In: *New Journal of Physics* 18.2 (2016), p. 023023.

[215] Pankaj Mehta et al. "A high-bias, low-variance introduction to machine learning for physicists". In: *Physics reports* 810 (2019), pp. 1–124. URL: `https://www.sciencedirect.com/science/article/pii/S0370157319300766`.

[216] Alexey A Melnikov, Pavel Sekatski, and Nicolas Sangouard. "Setting up experimental Bell test with reinforcement learning". In: *arXiv:2005.01697* (2020).

[217] Fan-Xu Meng et al. "Quantum circuit architecture optimization for variational quantum eigensolver via monto carlo tree search". In: *IEEE Transactions on Quantum Engineering* 2 (2021), pp. 1–10.

[218] Panayotis Mertikopoulos et al. "On the almost sure convergence of stochastic gradient descent in non-convex problems". In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 1117–1128.

[219] Friederike Metz and Marin Bukov. "Self-Correcting Quantum Many-Body Control using Reinforcement Learning with Tensor Networks". In: *arXiv preprint arXiv:2201.11790* (2022).

[220] Volodymyr Mnih et al. "Human-level control through deep reinforcement learning". In: *Nature* 518.7540 (2015), pp. 529–533.

[221] Volodymyr Mnih et al. "Playing atari with deep reinforcement learning". In: *arXiv preprint arXiv:1312.5602* (2013).

[222] Nikolaj Moll et al. "Quantum optimization using variational algorithms on near-term quantum devices". In: *Quantum Science and Technology* 3.3 (2018), p. 030503.

[223] Christopher Monroe and Jungsang Kim. "Scaling the ion trap quantum processor". In: *Science* 339.6124 (2013), pp. 1164–1169. URL: http://science.sciencemag.org/content/339/6124/1164.

[224] Mauro ES Morales, Jacob Biamonte, and Zoltán Zimborás. "On the Universality of the Quantum Approximate Optimization Algorithm". In: *arXiv preprint arXiv:1909.03123* (2019).

[225] Eric Moulines and Francis R Bach. "Non-asymptotic analysis of stochastic approximation algorithms for machine learning". In: *Advances in Neural Information Processing Systems*. 2011, pp. 451–459.

[226] Zakaria Mzaouali et al. "Work statistics and symmetry breaking in an excited-state quantum phase transition". In: *Physical Review E* 103.3 (Mar. 2021), p. 032145. URL: https://doi.org/10.1103%2Fphysreve.103.032145.

[227] Ken M Nakanishi, Keisuke Fujii, and Synge Todo. "Sequential minimal optimization for quantum-classical hybrid algorithms". In: *arXiv preprint arXiv:1903.12166* (2019).

[228] Hendrik Poulsen Nautrup et al. "Optimizing quantum error correction codes with reinforcement learning". In: *Quantum* 3 (2019), p. 215. DOI: 10.22331/q-2019-12-16-215.

[229] John A Nelder and Roger Mead. "A simplex method for function minimization". In: *The computer journal* 7.4 (1965), pp. 308–313.

[230] Yu Nesterov. "Efficiency of coordinate descent methods on huge-scale optimization problems". In: *SIAM Journal on Optimization* 22.2 (2012), pp. 341–362.

[231] Yurii Nesterov and Vladimir Spokoiny. "Random gradient-free minimization of convex functions". In: *Foundations of Computational Mathematics* 17.2 (2017), pp. 527–566.

[232] Yurii Nesterov and Sebastian U Stich. "Efficiency of the accelerated coordinate descent method on structured optimization problems". In: *SIAM Journal on Optimization* 27.1 (2017), pp. 110–123.

[233] Michael Neunert et al. "Continuous-Discrete Reinforcement Learning for Hybrid Control in Robotics". In: *arXiv preprint arXiv:2001.00449* (2020).

[234] Lam M Nguyen et al. "SGD and Hogwild! convergence without the bounded gradients assumption". In: *Proceedings of the 35th International Conference on Machine Learning*. 2018.

[235] Phuong Ha Nguyen, Lam M Nguyen, and Marten van Dijk. "Tight Dimension Independent Lower Bound on Optimal Expected Convergence Rate for Diminishing Step Sizes in SGD". In: *33rd Conference on Neural Information Processing Systems (NeurIPS 2019)*. 2019.

[236] Murphy Yuezhen Niu, Sirui Lu, and Isaac L Chuang. "Optimizing QAOA: Success Probability and Runtime Dependence on Circuit Depth". In: *arXiv preprint arXiv:1905.12134* (2019).

[237] Murphy Yuezhen Niu et al. "Universal quantum control through deep reinforcement learning". In: *npj Quantum Information* 5.1 (2019), p. 33.

[238] Jorge Nocedal and Stephen Wright. *Numerical optimization*. Springer Science & Business Media, 2006.

[239] Aaron van den Oord, Nal Kalchbrenner, and Koray Kavukcuoglu. "Pixel Recurrent Neural Networks". In: *arXiv preprint arXiv:1601.06759* (2016).

[240] Mateusz Ostaszewski, Edward Grant, and Marcello Benedetti. "Structure optimization for parameterized quantum circuits". In: *Quantum* (2021).

[241] Mohit Pandey et al. "Adiabatic eigenstate deformations as a sensitive probe for quantum chaos". In: *arXiv preprint arXiv:2004.05043* (2020). URL: https://arxiv.org/abs/2004.05043.

[242] G. Passarelli et al. "Counterdiabatic driving in the quantum annealing of the $p$-spin model: A variational approach". In: *Phys. Rev. Research* 2 (1 Mar. 2020), p. 013283. DOI: 10.1103/PhysRevResearch.2.013283. URL: https://link.aps.org/doi/10.1103/PhysRevResearch.2.013283.

[243] Vivak Patel, Shushu Zhang, and Bowen Tian. "Global convergence and stability of stochastic gradient descent". In: *Advances in Neural Information Processing Systems* 35 (2022), pp. 36014–36025.

[244] Alberto Peruzzo et al. "A variational eigenvalue solver on a photonic quantum processor". In: *Nature communications* 5 (2014), p. 4213.

[245] Jan Peters and Stefan Schaal. "Natural actor-critic". In: *Neurocomputing* 71.7-9 (2008), pp. 1180–1190.

[246] David Pfau et al. "FermiNet: Quantum Physics and Chemistry from First Principles". In: *Physical Review Research* 2.3 (2020), p. 033429.

[247] Frank Pollmann et al. "Entanglement spectrum of a topological phase in one dimension". In: *Phys. Rev. B* 81 (6 Feb. 2010), p. 064439. URL: https://link.aps.org/doi/10.1103/PhysRevB.81.064439.

[248] Boris Polyak. "Gradient methods for the minimisation of functionals". In: *Ussr Computational Mathematics and Mathematical Physics* 3 (Dec. 1963), pp. 864–878.

[249] Riccardo Porotti et al. "Coherent transport of quantum states by deep reinforcement learning". In: *Communications Physics* 2.1 (2019), p. 61.

[250] Michael JD Powell. "An efficient method for finding the minimum of a function of several variables without calculating derivatives". In: *The computer journal* 7.2 (1964), pp. 155–162.

[251] John Preskill. "Quantum Computing in the NISQ era and beyond". In: *Quantum* 2 (2018), p. 79.

[252] Qiskit contributors. *Qiskit: An Open-source Framework for Quantum Computing*. 2023. DOI: 10.5281/zenodo.2573505.

[253] Alec Radford et al. "Improving Language Understanding by Generative Pre-Training". In: (2018).

[254] Colin Raffel et al. "Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer". In: *arXiv preprint arXiv:1910.10683* (2019).

[255] Alexander Rakhlin, Ohad Shamir, and Karthik Sridharan. "Making gradient descent optimal for strongly convex stochastic optimization". In: *Proceedings of the 29th International Conference on Machine Learning*. 2012.

[256] Roshan Rao et al. "Evaluating Protein Transfer Learning with TAPE". In: *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*. Ed. by Hanna M. Wallach et al. 2019, pp. 9686–9698. URL: https://proceedings.neurips.cc/paper/2019/hash/37f65c068b7723cd7809ee2d31d7861c-Abstract.html.

[257] J. Rapin and O. Teytaud. *Nevergrad - A gradient-free optimization platform*. https://GitHub.com/FacebookResearch/Nevergrad. 2018.

[258] Carl Edward Rasmussen. *Gaussian processes for machine learning*. MIT Press, 2006.

[259] Esteban Real et al. "Automl-zero: Evolving machine learning algorithms from scratch". In: *International Conference on Machine Learning*. PMLR. 2020, pp. 8007–8019.

[260] Patrick Rebentrost et al. "Quantum gradient descent and Newton's method for constrained polynomial optimization". In: *New Journal of Physics* 21.7 (2019), p. 073023.

[261] Gautam Reddy et al. "Learning to soar in turbulent environments". In: *Proceedings of the National Academy of Sciences* 113.33 (2016), E4877–E4884.

[262] Benno S Rem et al. "Identifying quantum phase transitions using artificial neural networks on experimental data". In: *Nature Physics* 15.9 (2019), pp. 917–920.

[263] Peter Richtárik and Martin Takáăź. "Iteration Complexity of Randomized Block-Coordinate Descent Methods for Minimizing a Composite Function". In: *Math. Program.* 144.1–2 (2014), pp. 1–38.

[264] Jonathan Romero, Jonathan P Olson, and Alan Aspuru-Guzik. "Quantum autoencoders for efficient compression of quantum data". In: *Quantum Science and Technology* 2.4 (2017), p. 045001.

[265] Dominic C Rose, Jamie F Mair, and Juan P Garrahan. "A reinforcement learning approach to rare trajectory sampling". In: *arXiv preprint arXiv:2005.12890* (2020). URL: https://arxiv.org/abs/2005.12890.

[266] Bodo Rosenhahn and Tobias J Osborne. "Monte Carlo Graph Search for Quantum Circuit Optimization". In: *arXiv preprint arXiv:2307.07353* (2023).

[267] Stéphane Ross, Geoffrey J Gordon, and J Andrew Bagnell. "A reduction of imitation learning and structured prediction to no-regret online learning". In: *Proceedings of the fourteenth international conference on artificial intelligence and statistics*. 2011, pp. 627–635.

[268] Ruslan Salakhutdinov. "Deep learning". In: *The 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14, New York, NY, USA - August 24 - 27, 2014*. Ed. by Sofus A. Macskassy et al. ACM, 2014, p. 1973. DOI: 10.1145/2623330.2630809. URL: https://doi.org/10.1145/2623330.2630809.

[269] Alan C Santos et al. "Counter-diabatic driving in spin systems: A shortcut to adiabaticity". In: *Physical Review A* ().

[270] Frederic Sauvage and Florian Mintert. "Optimal quantum control with poor statistics". In: (2019). arXiv: 1909.01229. URL: http://arxiv.org/abs/1909.01229.

[271] Frederic Sauvage and Florian Mintert. "Optimal quantum control with poor statistics". In: *arXiv preprint arXiv:1909.01229* (2019). URL: https://arxiv.org/abs/1909.01229.

[272] Frank Schäfer et al. "A differentiable programming method for quantum control". In: (2020). arXiv: 2002.08376. URL: http://arxiv.org/abs/2002.08376.

[273] Frank Schäfer et al. "A differentiable programming method for quantum control". In: *Machine Learning: Science and Technology* 1 (2020). DOI: 10.1088/2632-2153/ab9802.

[274] Gernot Schaller and Ralf Schützhold. "The role of symmetries in adiabatic quantum algorithms". In: *arXiv preprint arXiv:0708.1882* (2007).

[275] Romana Schirhagl et al. "Nitrogen-vacancy centers in diamond: nanoscale sensors for physics and biology". In: *Annual review of physical chemistry* 65 (2014), pp. 83–105. URL: https://www.annualreviews.org/doi/abs/10.1146/annurev-physchem-040513-103659.

[276] Julian Schrittwieser et al. "Mastering Atari, Go, chess and shogi by planning with a learned model". In: *Nature* 588.7839 (Dec. 2020), pp. 604–609. DOI: 10.1038/s41586-020-03051-4. URL: https://doi.org/10.1038%2Fs41586-020-03051-4.

[277] Maria Schuld, Ilya Sinayskiy, and Francesco Petruccione. "An introduction to quantum machine learning". In: *Contemporary Physics* 56.2 (2015), pp. 172–185.

[278] John Schulman et al. "Proximal policy optimization algorithms". In: *arXiv preprint arXiv:1707.06347* (2017).

[279] John Schulman et al. "Trust region policy optimization". In: *International conference on machine learning*. 2015, pp. 1889–1897.

[280] Dries Sels and Anatoli Polkovnikov. "Minimizing irreversible losses in quantum systems by local counterdiabatic driving". In: *Proceedings of the National Academy of Sciences* 114.20 (2017), E3909–E3916.

[281] Or Sharir et al. "Deep Autoregressive Models for the Efficient Variational Simulation of Many-Body Quantum Systems". In: *Phys. Rev. Lett.* 124 (2 2020), p. 020503. DOI: 10.1103/PhysRevLett.124.020503. URL: https://link.aps.org/doi/10.1103/PhysRevLett.124.020503.

[282] Yuhui Shi et al. "Particle swarm optimization: developments, applications and resources". In: *Proceedings of the 2001 congress on evolutionary computation (IEEE Cat. No. 01TH8546)*. Vol. 1. IEEE. 2001, pp. 81–86.

[283] David Silver et al. "A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play". In: *Science* 362.6419 (2018), pp. 1140–1144.

[284] David Silver et al. "Mastering the game of Go with deep neural networks and tree search". In: *nature* 529.7587 (2016), pp. 484–489.

[285] David Silver et al. "Mastering the game of go without human knowledge". In: *nature* 550.7676 (2017), pp. 354–359.

[286] Sukin Sim, Peter D. Johnson, and Alán Aspuru-Guzik. "Expressibility and entangling capability of parameterized quantum circuits for hybrid quantum-classical algorithms". In: *Adv. Quantum Technol.* 2.12 (2019), p. 1900070.

[287] Sukin Sim et al. "Adaptive pruning-based optimization of parameterized quantum circuits". In: *Quantum Science and Technology* 6.2 (Mar. 2021), p. 025019. DOI: 10.1088/2058-9565/abe107. URL: https://doi.org/10.1088%2F2058-9565%2Fabe107.

[288] Vegard B Sørdal and Joakim Bergli. "Deep reinforcement learning for robust quantum optimization". In: *arXiv preprint arXiv:1904.04712* (2019).

[289] James Stokes et al. "Quantum natural gradient". In: *Quantum* 4 (2020), p. 269.

[290] Michael Streif and Martin Leib. "Training the quantum approximate optimization algorithm without access to a quantum processing unit". In: *Quantum Science and Technology* (2020).

[291] Helmut Strobel et al. "Fisher information and entanglement of non-Gaussian spin states". In: *Science* 345.6195 (2014), pp. 424–427.

[292] Kevin Sung. "Towards the First Practical Applications of Quantum Computers". PhD thesis. University of Michigan, 2020.

[293] Kevin J Sung et al. "Using models to improve optimizers for variational quantum algorithms". In: *Quantum Science and Technology* 5.4 (2020), p. 044008.

[294] Kevin J Sung et al. "Using models to improve optimizers for variational quantum algorithms". In: *Quantum Science and Technology* 5.4 (Sept. 2020), p. 044008. DOI: 10.1088/2058-9565/abb6d9. URL: https://dx.doi.org/10.1088/2058-9565/abb6d9.

[295] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[296] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 2018.

[297] Richard S. Sutton et al. "Policy Gradient Methods for Reinforcement Learning with Function Approximation". In: *Advances in Neural Information Processing Systems*. 2000.

[298] Ryan Sweke et al. "Stochastic gradient descent for hybrid quantum-classical optimization". In: *Quantum* 4 (2020), p. 314.

[299] Ryan Sweke et al. "Stochastic gradient descent for hybrid quantum-classical optimization". In: *Quantum* 4 (2020), p. 314.

[300] Zsolt Tabi et al. "Quantum Optimization for the Graph Coloring Problem with Space-Efficient Embedding". In: *2020 IEEE International Conference on Quantum Computing and Engineering (QCE)*. IEEE. 2020, pp. 56–62.

[301] Shiro Tamiya and Hayata Yamasaki. "Stochastic gradient line Bayesian optimization for efficient noise-robust optimization of parameterized quantum circuits". In: *npj Quantum Information* 8.1 (2022), p. 90.

[302] Ho Lun Tang et al. "qubit-ADAPT-VQE: An adaptive algorithm for constructing hardware-efficient ansatze on a quantum processor". In: *arXiv preprint arXiv:1911.05416* (2019).

[303] Yunhao Tang and Shipra Agrawal. "Boosting trust region policy optimization by normalizing flows policy". In: *arXiv preprint arXiv:1809.10326* (2018).

[304] David J Tannor, Vladimir Kazakov, and Vladimir Orlov. "Control of photochemical branching: Novel procedures for finding optimal pulses and global upper bounds". In: *Time-dependent quantum molecular dynamics*. Springer, 1992, pp. 347–360.

[305] Yi Tay et al. "Efficient transformers: A survey". In: *arXiv:2009.06732* (2020).

[306] Simone Tibaldi et al. "Bayesian optimization for qaoa". In: *arXiv preprint arXiv:2209.03824* (2022).

[307] A.N. Tikhonov. "Solution of incorrectly formulated problems and the regularization method". In: *Soviet Mathematics Doklady* 4 (1963), pp. 1035–1038.

[308] Jules Tilly et al. "The Variational Quantum Eigensolver: a review of methods and best practices". In: *arXiv preprint arXiv:2111.05176v1* (Nov. 2021). arXiv: `2111.05176v1 [quant-ph]`. URL: `http://arxiv.org/abs/2111.05176v1`.

[309] Giacomo Torlai et al. "Neural-network quantum state tomography". In: *Nature Physics* 14.5 (2018), pp. 447–450.

[310] Brandon Trabucco et al. "Conservative objective models for effective offline model-based optimization". In: *International Conference on Machine Learning*. PMLR. 2021, pp. 10358–10368.

[311] Agnes Valenti et al. "Hamiltonian learning for quantum error correction". In: *Physical Review Research* 1.3 (2019), p. 033092.

[312] Aaron Van den Oord et al. "Conditional image generation with pixelcnn decoders". In: *Advances in neural information processing systems* 29 (2016).

[313] Evert PL Van Nieuwenburg, Ye-Hua Liu, and Sebastian D Huber. "Learning phase transitions by confusion". In: *Nature Physics* 13.5 (2017), pp. 435–439.

[314] Ashish Vaswani et al. "Attention is all you need". In: *Advances in neural information processing systems*. 2017, pp. 5998–6008.

[315] Guillaume Verdon, Michael Broughton, and Jarrod Biamonte. "A Quantum Approximate Optimization Algorithm for Continuous Problems". In: *arXiv preprint arXiv:1902.00409* (2019).

[316] Pascal Vincent et al. "Extracting and Composing Robust Features with Denoising Autoencoders". In: *Proceedings of the 25th International Conference on Machine Learning*. ACM. 2008, pp. 1096–1103.

[317] Oriol Vinyals et al. "Grandmaster level in StarCraft II using multi-agent reinforcement learning". In: *Nature* 575.7782 (2019), pp. 350–354.

[318] Guoming Wang et al. "Bayesian inference with engineered likelihood functions for robust amplitude estimation". In: *arXiv preprint arXiv:2006.09350* (2020).

[319] Hanrui Wang et al. "QuantumNAS: Noise-Adaptive Search for Robust Quantum Circuits". In: *arXiv preprint arXiv:2107.10845v5* (July 2021). arXiv: 2107.10845v5 [quant-ph]. URL: http://arxiv.org/abs/2107.10845v5.

[320] Hanrui Wang et al. "RoQNN: Noise-Aware Training for Robust Quantum Neural Networks". In: *arXiv preprint arXiv:2110.11331v1* (Oct. 2021). arXiv: 2110.11331v1 [cs.LG]. URL: http://arxiv.org/abs/2110.11331v1.

[321] Lei Wang. "Discovering phase transitions with unsupervised learning". In: *Phys. Rev. B* 94 (19 2016), p. 195105. DOI: 10.1103/PhysRevB.94.195105. URL: https://link.aps.org/doi/10.1103/PhysRevB.94.195105.

[322] Linnan Wang, Rodrigo Fonseca, and Yuandong Tian. "Learning search space partition for black-box optimization using monte carlo tree search". In: *arXiv preprint arXiv:2007.00708* (2020).

[323] Sinong Wang et al. *Linformer: Self-Attention with Linear Complexity*. 2020. arXiv: 2006.04768 [cs.LG].

[324] Christopher J. C. H. Watkins and Peter Dayan. "Q-learning". In: *Machine Learning* 8 (1992), pp. 279–292.

[325] Matteo M Wauters et al. "Reinforcement Learning assisted Quantum Optimization". In: *arXiv preprint arXiv:2004.12323* (2020).

[326] Ermo Wei, Drew Wicke, and Sean Luke. "Hierarchical approaches for reinforcement learning in parameterized action space". In: *arXiv:1810.09656* (2018).

[327] Ee Weinan. "A proposal on machine learning via dynamical systems". In: *Communications in Mathematics and Statistics* 1.5 (2017), pp. 1–11.

[328] Phillip Weinberg and Marin Bukov. "QuSpin: a Python package for dynamics and exact diagonalisation of quantum many body systems part I: spin chains". In: *SciPost Phys* 2.1 (2017).

[329] Phillip Weinberg and Marin Bukov. "QuSpin: a Python package for dynamics and exact diagonalisation of quantum many body systems. Part II: bosons, fermions and higher spins". In: *SciPost Phys.* 7.arXiv: 1804.06782 (2019), p. 020.

[330] David Wierichs et al. "General parameter-shift rules for quantum gradients". In: *Quantum* 6 (2022), p. 677.

[331] Roeland Wiersema et al. "Exploring entanglement and optimization within the hamiltonian variational ansatz". In: *PRX Quantum* 1.2 (2020), p. 020319.

[332] Daan Wierstra et al. "Natural evolution strategies". In: *2008 IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence)*. IEEE. 2008, pp. 3381–3387.

[333] Daan Wierstra et al. "Natural evolution strategies". In: *Journal of Machine Learning Research* 15.1 (2014), pp. 949–980.

[334] Ronald J Williams. "Simple statistical gradient-following algorithms for connectionist reinforcement learning". In: *Machine learning* 8.3-4 (1992), pp. 229–256.

[335] S. J. Wright. "Coordinate Descent Algorithms". In: *Mathematical Programming, Series B* 151.1 (2015), pp. 3–34.

[336] Re-Bing Wu et al. "End-to-End Quantum Machine Learning with Quantum Control Systems". In: *arXiv preprint arXiv:2003.13658* (2020). URL: `https://arxiv.org/abs/2003.13658`.

[337] Re-Bing Wu et al. "Learning robust and high-precision quantum controls". In: *Physical Review A* 99.4 (2019), p. 042327.

[338] Dian Wu, Lei Wang, and Pan Zhang. "Solving Statistical Mechanics Using Variational Autoregressive Networks". In: *Physical Review Letters* 122.8 (2019), p. 080602.

[339] Yadong Wu et al. "Active Learning Approach to Optimization of Experimental Control". In: *arXiv preprint arXiv:2003.11804* (2020).

[340] Jonathan Wurtz, Pieter W. Claeys, and Anatoli Polkovnikov. "Variational Schrieffer-Wolff transformations for quantum many-body dynamics". In: *Phys. Rev. B* 101 (1 2020), p. 014302. DOI: 10.1103/PhysRevB.101.014302. URL: https://link.aps.org/doi/10.1103/PhysRevB.101.014302.

[341] Jonathan Wurtz and Peter J. Love. "Counterdiabaticity and the Quantum Approximate Optimization Algorithm". In: *arXiv preprint arXiv:2101.07808* (2021).

[342] Ze-Liang Xiang et al. "Hybrid quantum circuits: Superconducting circuits interacting with other quantum systems". In: *Rev. Mod. Phys.* 85 (2 2013), pp. 623–653. DOI: 10.1103/RevModPhys.85.623. URL: https://link.aps.org/doi/10.1103/RevModPhys.85.623.

[343] Jiechao Xiong et al. "Parametrized deep q-networks learning: Reinforcement learning with discrete-continuous hybrid action space". In: *arXiv preprint arXiv:1810.06394* (2018).

[344] Nanyang Xu et al. "Quantum factorization of 143 on a dipolar-coupling nuclear magnetic resonance system". In: *Physical review letters* 108.13 (2012), p. 130501.

[345] Zhi-Cheng Yang et al. "Optimizing variational quantum algorithms using Pontryagin's minimum principle". In: *Physical Review X* 7.2 (2017), p. 021027.

[346] Jiahao Yao, Marin Bukov, and Lin Lin. "Policy gradient based quantum approximate optimization algorithm". In: *Mathematical and Scientific Machine Learning*. PMLR. 2020, pp. 605–634.

[347] Jiahao Yao, Lin Lin, and Marin Bukov. "Reinforcement Learning for Many-Body Ground State Preparation based on Counter-Diabatic Driving". In: *arXiv preprint arXiv:2010.03655* (2020). URL: https://arxiv.org/abs/2010.03655.

[348] Jiahao Yao, Lin Lin, and Marin Bukov. "Reinforcement Learning for Many-Body Ground-State Preparation Inspired by Counterdiabatic Driving". In: *Phys. Rev. X* 11 (3 Sept. 2021), p. 031070. DOI: 10.1103/PhysRevX.11.031070. URL: https://link.aps.org/doi/10.1103/PhysRevX.11.031070.

[349] Jiahao Yao et al. "Monte Carlo Tree Search based Hybrid Optimization of Variational Quantum Circuits". In: *Proceedings of Mathematical and Scientific Machine Learning*. Ed. by Bin Dong et al. Vol. 190. Proceedings of Machine Learning Research. PMLR, 15–17 Aug 2022, pp. 49–64. URL: https://proceedings.mlr.press/v190/yao22a.html.

[350] Jiahao Yao et al. "Noise-Robust End-to-End Quantum Control using Deep Autoregressive Policy Networks". In: *arXiv preprint arXiv:2012.06701* (2020).

[351] Jiahao Yao et al. "Noise-Robust End-to-End Quantum Control using Deep Autoregressive Policy Networks". In: *Proceedings of the 2nd Mathematical and Scientific Machine Learning Conference*. Ed. by Joan Bruna, Jan Hesthaven, and Lenka Zdeborova. Vol. 145. Proceedings of Machine Learning Research. PMLR, 16–19 Aug 2022, pp. 1044–1081. URL: https://proceedings.mlr.press/v145/yao22a.html.

[352] Weirui Ye et al. "Mastering Atari Games with Limited Data". In: *arXiv preprint arXiv:2111.00210v2* (Oct. 2021). arXiv: 2111.00210v2 [cs.LG]. URL: http://arxiv.org/abs/2111.00210v2.

[353] Tom Young et al. "Recent Trends in Deep Learning Based Natural Language Processing". In: *IEEE Computational Intelligence Magazine* 13.3 (2018), pp. 55–75. DOI: 10.1109/MCI.2018.2840738.

[354] Xiao Yuan et al. "Theory of variational quantum simulation". In: *Quantum* 3 (2019), p. 191.

[355] Richard Zhang, Phillip Isola, and Alexei A. Efros. "Colorful Image Colorization". In: *European Conference on Computer Vision (ECCV)*. Springer, 2016, pp. 649–666.

[356] Shi-Xin Zhang et al. "Differentiable quantum architecture search". In: *Quantum Science and Technology* 7.4 (Aug. 2022), p. 045023. URL: https://doi.org/10.1088/2058-9565/ac87cd.

[357] Shi-Xin Zhang et al. "Neural Predictor based Quantum Architecture Search". In: *arXiv preprint arXiv:2103.06524v1* (Mar. 2021). arXiv: 2103.06524v1 [quant-ph]. URL: http://arxiv.org/abs/2103.06524v1.

[358] Xiao-Ming Zhang et al. "When reinforcement learning stands out in quantum control? A comparative study on state preparation". In: *arXiv preprint arXiv:1902.02157* (2019).

[359] Tianchen Zhao et al. "Natural evolution strategies and quantum approximate optimization". In: *arXiv preprint arXiv:2005.04447* (2020).

[360] Tianchen Zhao et al. "Natural evolution strategies and variational Monte Carlo". In: *Machine Learning: Science and Technology* 2.2 (Dec. 2020), 02LT01. DOI: 10.1088/2632-2153/abcb50. URL: https://dx.doi.org/10.1088/2632-2153/abcb50.

[361] Hui Zhou et al. "Experimental realization of shortcuts to adiabaticity in a nonintegrable spin chain by local counterdiabatic driving". In: *Physical Review Applied* 13.4 (2020), p. 044059.

[362] Leo Zhou et al. "Quantum Approximate Optimization Algorithm: Performance, Mechanism, and Implementation on Near-Term Devices". In: *Physical Review X* 10.2 (2020), p. 021067.

[363] Xiangzhen Zhou, Yuan Feng, and Sanjiang Li. "A Monte Carlo Tree Search Framework for Quantum Circuit Transformation". In: *arXiv preprint* (Aug. 2020). arXiv: 2008.09331v2 [quant-ph]. URL: http://arxiv.org/abs/2008.09331v2.

[364] Daiwei Zhu et al. "Training of quantum circuits on a hybrid quantum computer". In: *Science advances* 5.10 (2019), eaaw9918.

[365] Linghua Zhu et al. "An adaptive quantum approximate optimization algorithm for solving combinatorial problems on a quantum computer". In: *arXiv preprint arXiv:2005.10258* (2020). URL: https://arxiv.org/abs/2005.10258.

[366] Brian D Ziebart. "Modeling purposeful adaptive behavior with the principle of maximum causal entropy". In: (2010). URL: https://www.cs.cmu.edu/~bziebart/publications/thesis-bziebart.pdf.

[367] Barret Zoph and Quoc V. Le. "Neural Architecture Search with Reinforcement Learning". In: *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. URL: https://openreview.net/forum?id=r1Ue8Hcxg.