Optimal control and machine learning of quantum device dynamics

Francesco Preti

Schlüsseltechnologien / Key Technologies
Band / Volume 302
ISBN 978-3-95806-856-8

JÜLICH
Forschungszentrum

Schriften des Forschungszentrums Jülich
Reihe Schlüsseltechnologien / Key Technologies		Band / Volume 302

Forschungszentrum Jülich GmbH
Peter Grünberg Institut (PGI)
Quantum Control (PGI-8)

Optimal control and machine learning of quantum device dynamics

Francesco Preti

Schriften des Forschungszentrums Jülich
Reihe Schlüsseltechnologien / Key Technologies Band / Volume 302

ISSN 1866-1807 ISBN 978-3-95806-856-8

Bibliografische Information der Deutschen Nationalbibliothek.
Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte Bibliografische Daten sind im Internet über http://dnb.d-nb.de abrufbar.

Herausgeber und Vertrieb:	Forschungszentrum Jülich GmbH Zentralbibliothek, Verlag 52425 Jülich Tel.: +49 2461 61-5368 Fax: +49 2461 61-6103 zb-publikation@fz-juelich.de www.fz-juelich.de/zb
Umschlaggestaltung:	Grafische Medien, Forschungszentrum Jülich GmbH
Titelbild:	Flux Pro by Black Forest Labs
Druck:	Grafische Medien, Forschungszentrum Jülich GmbH
Copyright:	Forschungszentrum Jülich 2025

Schriften des Forschungszentrums Jülich
Reihe Schlüsseltechnologien / Key Technologies, Band / Volume 302

D 38 (Diss. Köln, Univ., 2025)

ISSN 1866-1807
ISBN 978-3-95806-856-8

Vollständig frei verfügbar über das Publikationsportal des Forschungszentrums Jülich (JuSER) unter www.fz-juelich.de/zb/openaccess.

To see a world in a grain of sand
And a heaven in a wild flower,
Hold infinity in the palm of your hand
And eternity in an hour.

– William Blake, Auguries of Innocence

The rapid evolution of quantum computers requires more and more refined optimization strategies to enhance the performance of present and future quantum devices. Such strategies span from pulse-level optimization in quantum control and variational circuits to combinatorial optimization in system design and quantum circuit compilation. As research progresses, such optimization tasks will also become more and more relevant in the design of single and interconnected quantum devices, such as, e.g., quantum networks. In this work, we present several examples of potentially relevant optimization problems in quantum devices. We start with meta-optimization and system identification in quantum control and later move on to hybrid continuous-discrete compilation in quantum circuits. We also consider the optimization of entanglement purification protocols. We show how variational protocols can be optimized successfully to purify families of quantum states and, more specifically, how optimized protocols can surpass previous proposals for random two-qubit states. We conclude by considering different types of estimators for observables of quantum systems. Such estimators are relevant to all aforementioned optimization tasks, ranging from applications in variational quantum circuits to the estimation of cost functions for control problems.

Contents

Introduction

The last three decades have seen the rapid evolution of quantum information. Its theoretical foundations were elaborated in the seminal works of Paul Benioff [Ben80] and Richard Feynman [Fey82]. Today quantum information encompasses physicists, mathematicians, computer scientists, and engineers all over the world. Quantum algorithms that promise potential speedup over classical counterparts are among the most relevant achievements of the field [Sho94; Gro96; DJ92; BV97]. Some of these discoveries are causing concern because quantum computers could potentially break the most modern cryptographic schemes [IAA&al21]. Simultaneously, quantum communication emerged as a potential tool to achieve an extremely secure exchange of information over networks [BB14]. Today, private companies implement quantum-key distribution protocols over distances of hundreds of kilometers [RZV&al23].
The first models of quantum computations implemented abstract operations. However, a quantum computer must be constructed on a physical quantum setup that uses universal sets of gates, and in particular entangling gates. The first realization of the Cirac-Zoller proposal [CZ95] of a controlled-NOT gate in trapped-ion systems dates back to 2003 [Sch&al03]. This achievement was the first attempt to realize an entangling gate in a quantum system. At present, several companies attempt to implement fault-tolerant quantum computation on different types of quantum devices [RAC&al24; AAA&al24].
At the lowest level, a quantum computing device consists of qubits, which are the basic units of quantum information. Qubits are prepared in an initial quantum state and a series of operations can be performed on the state itself. Implementing an arbitrary operation on a register of qubits is not a trivial task. First, the types of interactions available on the system may not allow a direct implementation of the operation itself. For instance, a system with only two-qubit interactions cannot directly apply an n-qubit interaction. However, an n-qubit interaction can be implemented through layers of single and two-qubit operations [NC10]. In addition, even in the presence of the correct type of interaction, the approach to performing a certain operation that starts from an arbitrary Hamiltonian is often unclear. Second, quantum systems require careful engineering to become potential candidates for quantum computers. The detrimental effects of decoherence are unfortunately ubiquitous and therefore algorithms must be carefully designed that can implement quantum operations successfully. Quantum control and quantum compiling are sub-fields of quantum computation and technology that try to address the problem of implementing operations on qubits efficiently.
In quantum systems, optimization is not a single-task process but requires the cooperation of multiple levels of careful engineering [YSK&al20]. At the lowest levels, the quantum system undergoes a dynamic evolution governed by its Hamiltonian. In some systems, such as superconducting qubits, various types of qubits can be engineered with

different properties [Men&al21; KMS&al21; GGD&al21; CRC&al23]. Maximizing useful properties of the qubits, and at the same time reducing detrimental effects, is one of the targets of optimization algorithms. This task falls under the realm of so-called system design optimization [MMD&al21]. The quantum system usually interacts with external time-dependent electromagnetic fields that can be used to control the dynamics of the system itself. Optimal quantum control studies how to implement and optimize time-dependent signals to generate certain types of interactions using both system dynamics and driving fields [Koc&al22]. However, the implementation of arbitrary quantum algorithms requires layers of such interactions. Generating a quantum algorithm starting from basic gates is the task studied by quantum compilation algorithms [Mar&al22; YIL&al21].

In recent years, a new class of quantum algorithms based on classical optimization has emerged. These algorithms are referred to as (variational) algorithms for NISQ (Noisy Intermediate Scale Quantum) [Pre18] devices. Such algorithms could potentially achieve quantum speedup already on noisy devices that do not exhibit fault-tolerance, although this claim remains unproven and has been recently disputed [McC&al18; CLG&al24].

Optimization in quantum systems has become particularly important with the rise of variational circuits [Cer&al21b] and related applications such as quantum machine learning [Bia&al17]. The necessity to optimize parametric gates has led to the development of specialized algorithms used to optimize quantum systems [OGB21; WIW&al22]. On the other hand, developments in quantum control have led to the necessity of optimizing parametric pulses to achieve high-fidelity operations in quantum experiments [PCM22].

In summary, we have several potential types of optimization tasks in quantum systems that are relevant for quantum science and technology. The design of qubit systems with particular properties, e.g., in superconducting qubits, can benefit from optimization methods. Quantum control algorithms optimize time-dependent control signals in the form of electromagnetic fields acting on the qubits. Variational algorithms make use of optimization routines to solve a vast range of problems using layers of entangling and non-entangling gates. Discrete optimization in the sense of compilation is the task of designing an optimal sequence of such gates so that the corresponding quantum circuit generates a target quantum algorithm.

Summary

In this work, we first consider standard quantum control problems for superconducting transmon qubits. Such quantum systems have been studied extensively both from an analytical and a numerical perspective [MSG&al11; TMW16]. Analytical pulses usually show dependencies from system parameters that are not easily found in numerical solutions. Therefore, our approach is then to employ neural networks to learn the functional dependence of optimal quantum control solutions from system parameters. We show that we can optimize such solutions analytically for single and two-qubit gates using either very few parameter samples or large numbers of pulse frequencies for large parameter ranges. Afterward, we move to a higher level of optimization. We consider variational gates in trapped-ion quantum computers [MMN&al16]. In this setting, we employ a hybrid scheme that uses both reinforcement learning and continuous optimization to optimize both circuit structure and variational angles concurrently [BDS&al18; SEL&al22]. We show that our reinforcement learning algorithm assisted by a continuous optimizer can construct effective solutions to the gate synthesis problem that matches and surpasses standard circuit compilers.
Next, we consider variational optimization of non-linear maps [HCS&al23] acting on quantum states such as entanglement purification protocols. An entanglement purification protocol acts as a highly non-linear map, but it still outputs a faithful representation of a quantum system. Optimizing it for a specific family of states requires reducing the number of operations needed to retrieve the original state. This optimization has the potential to make purification protocols easier to implement on a quantum device. In fact, the exponential scaling of the purification protocol with the number of states [DB07] implies that such protocols may not be applied directly without careful engineering. We also show that the performance of standard purification protocols for arbitrary two-qubit input states leads to poor output values of the concurrence. Our optimized protocols prove instead able to increase the value of the concurrence above the maximum limit of traditional analytical protocols. We also show how the twirling operation becomes an obstacle to the performance of the protocol itself when considering random two-qubit states, although it is a useful tool in the design of entanglement purification protocols.
Finally, we study parameter sampling in quantum circuits, focusing in particular on the LCU methods. Such problems are particularly interesting for meta-variational settings [CKA21] where we compute the average over observables evaluated at different points in the parameter space. They are also relevant for optimal quantum control algorithms because the computation of the fidelity with respect to a target operation is the basis of most optimal quantum control routines.
In conclusion, we analyze several optimization problems that are relevant for quantum science and technology. We show that machine learning-assisted solutions can be ap-

plied successfully to engineer optimal quantum control pulses and compilation strategies based on variational angles. Also in the context of entanglement purification, we show how our optimized protocols can surpass current strategies for multiparametric families of states.

1 Quantum computation

1.1 Introduction

Quantum computers promise to revolutionize computer science as they are believed to be faster than classical computers in performing specific tasks [NC10]. During the course of the last three decades, several algorithms with provable speedup compared to their classical counterparts have been discovered. Examples of such cases are the Shor's algorithm [Sho94], which could in principle allow to break RSA cryptographic keys, Grover search and amplitude amplification [Gro96; Bra&al02], the HHL algorithm for the solution of linear systems of equations [HHL09], etc. Quantum algorithms also offer potential speedups in the simulation of quantum systems [GAN14] and differential equations [LPG&al20]. The last years have also seen the emergence of variational algorithms [Cer&al21b] that encode specific optimization problems on quantum circuits and then use classical opimization methods to try and reach the ground state of the problem. Such methods are of particular relevance for several branches of quantum science, e.g., quantum chemistry and quantum machine learning. Speedups offered by such algorithms are of particular interest for both academia and industry [BBB&al21]. Unfortunately, quantum computers are not easy to build in practice. They are prone to errors and affected by decoherence phenomena. As a result, considerable effort has to be invested in developing quantum error-correction schemes [DMN13] and in suppressing decoherence processes acting on qubits. At the same time reducing the overall number of unitary operations and measurements required by quantum algorithms is extremely important. In fact, the use of shorter circuits helps limit the influence of decoherence on the register of qubits and increases the speed of calculations.

1.2 Quantum computation

Quantum computation deals with the execution of operations and the readout of information using a quantum system. On a classical computer, information is stored on a bit, which is usually the internal state of a transistor (either zero or one). The transistor-transistor logic allows for the implementation of arbitrary logical operations [VMV&al22]. Due to the extreme speed in performing operations, transistors have become ubiquitous, but, in principle, the logical operations they provide can be implemented using different types of electronical and mechanical computation models. On a more theoretical level, a classical computer can be realized using mechanical systems, such as the billiard ball computer of Toffoli and Fredkin [FT82], although these models are of limited practical utility. Classical bits are acted upon by logical operations, such as, e.g., the NOT operation, which negates the logical state it acts upon and can be represented using a 2×2 matrix acting on a two-element vector representing the digital state in a matrix-vector multiplication fashion [Han]:

$$\text{NOT} : \{0,1\} \mapsto \{0,1\}, x \longmapsto \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \cdot x. \tag{1.1}$$

The NOT operation acts on a single logical bit, that is it does not perform binary additions or multiplications of binary digits, and it is reversible, since a second application of the gate brings the state to its original form. Other gates typical of classical computation do not have this property. This is the case, for instance, of the AND gate, which performs binary addition

$$\text{AND} : \{0,1\} \otimes \{0,1\} \mapsto \{0,1\}, x \otimes x \longmapsto \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix} \cdot (x \otimes x). \tag{1.2}$$

The reason for the lack of invertibility lies in the fact that the AND gate erases part of the information contained in the initial binary digit vector. This is connected to the Landauer principle [Lan61] and the amount of energy dissipated when deleting a bit of information, which leads to an increase in the overall entropy of the universe. Nonetheless, operations that are not invertible are not unheard of in classical computation. It was proved by Tommaso Toffoli that the Toffoli gate [BBC&al95], a three-qubit logical gate is universal for classical computation and can implement classical reversible computation (as it is an orthogonal symmetrical gate, so its inverse is just the gate iself). It turns out that this gate, together with the Hadamard gate, is also universal for quantum computation. Quantum computation instead uses the degree of freedom of a quantum system to store information. The minimum degree of freedom of a quantum system is given by a two-level system, i.e., a qubit

$$|\psi\rangle = \alpha |0\rangle + \beta |1\rangle, |\psi\rangle \in \mathbb{C}^2, \tag{1.3}$$

with the normalization condition $|\alpha|^2 + |\beta|^2 = 1$. The time evolution of a pure quantum state in absence of interaction with any external environment is given by the Schrödinger equation for a pure state and a Hamiltonian H that describes the dynamics of the system (setting $\hbar = 1$):

$$i \frac{\mathrm{d}}{\mathrm{d}t} |\psi(t)\rangle = H |\psi(t)\rangle. \tag{1.4}$$

and the Von Neumann equation for a mixed state

$$\frac{\mathrm{d}}{\mathrm{d}t} \rho(t) = -i[H, \rho(t)]. \tag{1.5}$$

For a single-qubit Hamiltonian, the most general parametrization uses three real parameters a_x, a_y, a_z:

$$H = a_x\sigma_x + a_y\sigma_y + a_z\sigma_z, \tag{1.6}$$

where σ_x, σ_y and σ_z are the three Pauli matrices

$$\sigma_x = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad \sigma_y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}, \quad \sigma_z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}. \tag{1.7}$$

Quantum-mechanically, coherent operations on a qubit are described by a single-qubit gate $U \in \mathrm{SU}(2)$, which has the following general three-dimensional standard parametrization:

$$U(a_x, a_y, a_z) = e^{-i(a_x\sigma_x + a_y\sigma_y + a_z\sigma_z)}, \tag{1.8}$$

which can be rewritten using the Euler-angle representation [NC10] of the same group to obtain standard single-qubit gate of IBM-*Qiskit* [Qis23; NC10]:

$$U(\theta, \phi, \lambda) = \begin{bmatrix} \cos\left(\frac{\theta}{2}\right) & -e^{i\lambda}\sin\left(\frac{\theta}{2}\right) \\ e^{i\phi}\sin\left(\frac{\theta}{2}\right) & e^{i(\phi+\lambda)}\cos\left(\frac{\theta}{2}\right) \end{bmatrix}, \tag{1.9}$$

Single qubits have two degrees of freedom and can therefore be represented as points on a sphere, called Bloch sphere [NC10]. Single-qubit operations can bring a single qubit to any point of the Bloch sphere, but they are not enough to generate all possible unitaries for n-qubit systems. For an overview of the more general parametrization of $\mathrm{SU}(d)$ acting on n-qubits, where $d = 2^n$ – see also Ref. [TS02]. Non-unitary operations on qubits can be implemented by including measurements on so-called ancillary systems. An operation involving measurements and unitaries acting on mixed states is a general quantum operation [Cho75; Sti55].

1.2.1 Universal gate sets

In classical logic, a small subset of logical operations can be used to represent arbitrary binary functions of the type $f : \{0,1\}^n \mapsto \{0,1\}$ [NC10]. This set is not unique: for example the triple of operations AND, OR and NOT forms a universal set of logical operations, whereas the Toffoli gate alone, a three-bit operation, is universal. In quantum computing this concept can be extended to the one of universal sets of quantum gates. An example is given by the two-qubit controlled-NOT (CNOT) gate:

$$\mathrm{CNOT} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}, \tag{1.10}$$

together with the Hadamard, the S and the T gate [NC10]:

$$S = \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix}, \; T = \sqrt{S}, \; H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}. \tag{1.11}$$

One should note that the Clifford group can be constructing using only CNOTs, Hadamard and S gates [Got98]. These gates, however, and therefore any element of this group can be simulated efficiently on classical computers due to the Gottesman-Knill theorem [Got98]. Another example of universal sets of gates for quantum computation is given by the Toffoli and the Hadamard gate [Aha03]. We see here that, compared to the classical case, the Toffoli gate alone is not universal for quantum computation. The importance of such universal sets of quantum gates is connected to a milestone result in quantum computation, the Solovay-Kitaev theorem [DN06; NC10]. This states that we can use a set of gates that generates a dense subgroup of $SU(2)$ to construct arbitrary quantum gates up to a precision ϵ using $O(\log^c(1/\epsilon))$ gates, where c is a positive finite constant [NC10; DN06]. As a result of this theorem, if it is possible to engineer platform-dependent sets of universal gates that exploit the specific features of the quantum systems considered, then it is also possible to implement arbitrary quantum operations. However, the overall depth of a circuit implementing a specific operation may vary from one system to another. For example, multi-qubit GHZ states are comparatively easier to generate in trapped-ion gates compared to superconducting qubits, thanks to the Mølmer-Sørensen interaction [SM99; MS99; Pre20].
Quantum computing is often introduced using the so-called circuit model of quantum computation, but this is not the only model available. Quantum computation can also be performed by carrying out single-qubit measurements on highly entangled multi-particle states – 2D cluster states [Bri&al09].

1.2.2 Quantum computing platforms

There exist several quantum systems that can be used to build quantum computers thanks to their intrinsic features. The guidelines to identify whether a quantum system is suitable for quantum computation are given by the so-called Di Vincenzo's criteria [DiV00]:

- The quantum system considered as a candidate for quantum computation should have well-defined qubits and should be scalable, that is, it should be possible to add more and more qubits without encountering significant difficulties.
- On the same system, the preparation of a specific state, in particular the zero qubit, should be feasible. In fact, the zero qubit is the conventional starting point for most quantum algorithms.
- The system should have coherence times that are longer than the execution times of logical operations. In fact, the action of decoherence dilutes entanglement and

makes it impossible to exploit its properties for quantum computation [Zur03; Zeh02].

- We should be able to implement a universal set of quantum gates efficiently on the quantum system of interest.
- There should be an effective measurement operation that characterizes the state of the qubit after performing quantum gates for, e.g., logical operations.

Over the course of the last decade, we have witnessed an enormous progress in the realization of primitive quantum devices that, while still unsuitable for fault-tolerant quantum computation, could be still employed as so-called NISQ (Noisy Intermediate Scale Quantum) devices [BDL24]. Applications of such devices range from annealing [HKL&al20] to hybrid quantum-classical optimization such as that considered in QAOA [FGG14] (Quantum Approximate Optimization Algorithm) and quantum machine learning. Some of these applications have proved to be more challenging than expected [CLG&al24], in particular due the phenomenon of so-called Barren plateaus [McC&al18; LTW&al24], i.e., large areas where the gradient of the quantities sampled from the quantum systems quickly approaches zero.
Platforms for quantum computing considered so-far include: trapped ions [BCM&al19] in radiofrequency traps, cold atoms in optical lattices [SWM10], superconducting quantum circuits driven by microwave pulses [HWF&al20], quantum dots in semiconductors [LD98; KL13], cavity-based photonic quantum computers [SP19] and NV centers in diamond [ROM&al20]. The results obtained by the different platforms vary wildly: some of these quantum systems offer incredibly long coherence times [BCM&al19], but lack, e.g., scalability or a proper measurement procedure. In the case of photonic quantum computers, it is not possible to easily isolate and store qubits, so that these systems are more suitable for, e.g., measurement-based quantum computation [Bri&al09], and specific quantum computing tasks such as boson sampling [BQA20]. In the last years, superconducting and photonic quantum computers have been used to claim quantum advantage [Mad&al22; Aru&al19].
In the following sections, we introduce the basics of these quantum computing platforms, so to emphasize their strength and weaknesses, such as the upcoming challenges that, if solved, could significantly advance the state of quantum computing research. In particular, we briefly outline the principles of quantum computers based on trapped ions in radiofrequency traps and superconducting quantum circuits.

1.2.2.1 Trapped-ion quantum computers

Trapped-ion quantum computers use a Paul trap to confine cold ions at a temperature of $T \sim 20$mK thanks to laser and Doppler cooling [BCM&al19]. The Paul trap [Pau90] is a device that uses a quadrupole with static and time-varying electric fields to trap charged particles at the center of the potential. The type of ions employed usually show a convenient fine- or hyperfine structure which can be easily addressed with lasers, e.g., $^{43}\mathrm{Ca}^{+}$ or $^{171}\mathrm{Yb}^{+}$ [BCM&al19]. Qubits in trapped-ion systems are characterized by the type of energy splitting considered: Zeeman, fine-structure, hyperfine-structure or optical [BCM&al19]. State preparation and readout are achieved by addressing the qubits with laser pulses. In the latter case, photons at a particular wavelength are emitted during scattering. Although one of the original proposals for a CNOT gate for quantum computers, the Cirac-Zoller gate, was developed for trapped ions [CZ95] and was realized experimentally in 2003 [Sch&al03], the current established entangling gate for this platform is the Mølmer-Sørensen (MS) gate [SM99; MS99]. Trapped-ion gates have a gate time of around 100 μs [BCM&al19], which is significantly shorter than the coherence time of ions. However, the scalability remains a problem, due to the fact that the trap has a limited size and can only keep a given maximum number of ions confined at the same time. Current attempts at finding a solution for this issue rely on interconnecting multiple microtraps [CZ00; KMW02]. Nonetheless, the architecture has proved particularly successful and it is one of the current most promising candidates for the realization of a quantum computer.

1.2.2.2 Superconducting quantum circuits

Superconducting quantum circuits are solid-state devices that can be used to engineer qubits. A LC circuit made of solid-state materials that exhibit superconducting properties will show zero dissipation as its resistance drops to zero when cooled down below its critical temperature. The addition of a non-linearity, such as the one Josephson junction [Jos62], can be used to create a qubit. In fact, if only an LC circuit without additional modifications were used, the resulting system would be a perfect harmonic oscillator [BGG&al21], which cannot be used as qubits, as driving one transition would drive all transitions at the same time, giving rise to a coherent state. The non-linearity of the Josephson junction modifies the structure of the eigenstates, thus rendering the system suitable for quantum computing. Other non-linear elements that can be used to create a qubit are SQUIDs (Superconducting Quantum Interference Devices) [BGG&al21]. There are several types of qubits that have been implemented in the last decades using different physical systems: transmon [KYG&al07], flux qubit and phase qubit [MKG&al09], $0-\pi$-qubit [BKP13], unimon [Hyy&al22], etc. Qubits can be addressed by driving them with microwave fields [HWF&al20]. Qubits based on superconducting circuits exhibit coherence times of up to a few milliseconds [SFM&al23] in the most recent implementations, which is considerably shorter than the coherence time of qubits in trapped ions. This is one of the reasons why quantum control [TMM&al18] and optimized compilation and transpilation procedures are so important for this type of architecture. Leakage in the $|1\rangle \mapsto |2\rangle$ transition [HWF&al20] is also a relevant issue. Specific pulses to control the qubit and at the same time mitigate the effect of leakage have been developed [MGR&al09]. More recent implementations have successfully tried to generalize the approach to the mitigation of leakage in two-qubit gates [LCM22; LCM24].

1.3 Counteracting errors

Whenever we find ourselves manipulating, storing or transmitting quantum information, we always have to deal with the transmission of such information over noisy channels. On a typical quantum computing platform, sources of noise include environmental decoherence caused by thermal fluctuations [ICJ&al05], excitation caused by cosmic rays [McE&al22], distortions in the experimental control lines [GZB&al13], etc. In quantum communication, that is for tasks such as quantum teleportation over long distances, we have different types of noise, such as depolarization, dephasing and amplitude damping [NC10]. Over the years, different solutions have been developed to protect the information stored in the quantum system from being erased by noise: on the one hand quantum error correction makes use of ancillary qubits to store the information redundantly (analogously to the classical case) and to detect and eventually correct the action of channel operators on the quantum state. Quantum error correction schemes encode the state $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle, |\psi\rangle \in \mathbb{C}^2$, with $|\alpha|^2 + |\beta|^2 = 1$ in a different state lying in a larger Hilbert space $\mathbb{C}^{2^n}$ – see Ref. [DB07]:

$$|\psi\rangle|0\rangle^{\otimes n-1} \longmapsto |\tilde{\psi}\rangle = \tilde{\alpha}|0_n\rangle + \tilde{\beta}|1_n\rangle, \tag{1.12}$$

with different amplitudes $|\tilde{\alpha}|^2 + |\tilde{\beta}|^2 = 1$. However, the encoding should not erase the information originally stored in α, β, so that it can be retrieved after executing the quantum algorithm. Quantum error correction [DMN13] studies the properties of such encoding schemes that allow for error detection and eventual correction. The main obstacle of using quantum error correction schemes in quantum networks is given by the amount of error that such schemes can counteract. This low error threshold restricts the application of quantum correction schemes to quantum communication channels of relatively short length [DB07]. Entanglement purification instead uses an ensemble of N pairs of noisy states over Hilbert spaces denoted by $A_1, B_1, A_2, B_2, ..., A_N, B_N$:

$$\rho = \rho_{A_1B_1} \otimes \rho_{A_2B_2} \otimes \ldots \otimes \rho_{A_NB_N}, \tag{1.13}$$

which undergo the influence of a noisy channel and a series of entangling and LOCC operations [CLM&al14] on the states and measurements on a sub-system. The repeated application of such operations generates each time a new state after the collapse induced by the measurements. The goal is to perform the sequence of operations in such a way that the fidelity of this output state with respect to a target (usually an entangled pure state) is larger than the initial fidelity. In the following, we are providing a short introduction to the main protocols employed in entanglement purification.

1.3.1 Entanglement purification

Quantum communication requires sharing information encoded in quantum states between two nodes name Alice (quantum system A) and Bob (quantum system B) over noisy channels. The action of a quantum channel turns, e.g., a pure state shared by

A and B into a mixed state. The idea of entanglement purification is to increase the amount of entanglement in a quantum state at the expense of the entanglement in other quantum states. The fundamental principle is not dissimilar to quantum error correction. The final goal is to extract an entangled pure state, such as one of the Bell states shared by A and B:

$$|1\rangle = \frac{1}{\sqrt{2}}\left(|0_{A_1}0_{B_1}\rangle + |1_{A_1}1_{B_1}\rangle\right) \tag{1.14}$$

$$|2\rangle = \frac{1}{\sqrt{2}}\left(|1_{A_1}0_{B_1}\rangle + |0_{A_1}1_{B_1}\rangle\right) \tag{1.15}$$

$$|3\rangle = \frac{1}{\sqrt{2}}\left(|0_{A_1}1_{B_1}\rangle + |1_{A_1}0_{B_1}\rangle\right) \tag{1.16}$$

$$|4\rangle = \frac{1}{\sqrt{2}}\left(|0_{A_1}0_{B_1}\rangle - |1_{A_1}1_{B_1}\rangle\right), \tag{1.17}$$

from an ensemble of mixed states. The oldest protocol for entanglement purification was proposed by Bennet et al. [BBP&al96] for Werner states and by Deutsch et al. [DEJ&al96] for Bell diagonal states. These are states that resemble some of the typical forms of quantum channels acting on density matrices. If we consider for instance the depolarizing channel [NC10] for two qubits

$$\Lambda(\rho) = \lambda\rho + \frac{1-\lambda}{4}\mathbb{I}_4, \tag{1.18}$$

and compute its action on a Bell pair $\rho = |1\rangle\langle 1|$, we obtain:

$$\rho' = \Lambda(\rho) = \frac{3}{4}\lambda\,|1\rangle\langle 1| + \frac{1-\lambda}{4}\left(\mathbb{I}_4 - |1\rangle\langle 1|\right). \tag{1.19}$$

If we consider now the overlap between this state and the $|1\rangle$ state, we have

$$F_1 = \mathrm{Tr}\left\{\rho'\,|1\rangle\langle 1|\right\} = \lambda\left(1 + \frac{1}{4}\right) - \frac{\lambda}{4} = \lambda. \tag{1.20}$$

2 Quantum control and optimization

2.1 Introduction

In quantum computation based on the circuit model, we always assume to have access to a universal set of entangling gates. However, in many quantum systems it is not obvious how to generate a specific gate, such as a CNOT gate, on an arbitrary system. Moreover, qubits used for quantum computation are generally not isolated systems: there are, e.g., higher energy levels where excitations can occur and the qubit is also coupled to an external environment [BGG&al21]. Moreover, there are state preparation and measurement errors (SPAM) [JE15; LMK&al22], which are inevitable when operations are executed and information extracted from the qubits. When dealing with operations on quantum systems, one often uses the concept of fidelity, which quantifies the overlap between the actual operation implemented on the device and the target operation, e.g., a specific entangling gate, that should be implemented. Different sources of noise prevent us from implementing high-fidelity operations, which are a fundamental requirement for fault-tolerant quantum computation. Quantum control deals with the optimal implementation of operations on quantum systems. This is particularly important for gate synthesis and state preparation [GBG&al19; ROM&al20].

2.2 Classical optimal control theory

In classical control theory [Kir04], a dynamical system is given which undergoes a temporal evolution of the kind:

$$\dot{\boldsymbol{x}}(t) = f(\boldsymbol{x}(t), \boldsymbol{u}(t), t), \tag{2.1}$$

with $t \in \mathbb{R}_{\geq 0}$ and time-dependent functions $\boldsymbol{x} : \mathbb{R}_{\geq 0} \mapsto \mathbb{R}^N, t \longmapsto \boldsymbol{x}(t)$ and $\boldsymbol{u} : \mathbb{R}_{\geq 0} \mapsto \mathbb{R}^M, t \longmapsto \boldsymbol{u}(t)$. We use here the formalism given in Ref. [Kir04]. The goal of the optimal control task is to find an appropriate optimal function $\boldsymbol{u}^*$ that minimizes the functional $J(\boldsymbol{x}, \boldsymbol{u}, T)$:

$$J(\boldsymbol{x}, \boldsymbol{u}, T) = C(\boldsymbol{x}(T), T) + \int_{t_0}^{T} E(\boldsymbol{x}(t), \boldsymbol{u}(\tau), \tau) d\tau, \tag{2.2}$$

where C is a appropriate cost functional for time T and the integral of E is the function whose integral needs to be minimized over the trajectory. Thus, the function C generally describes where the final state of the system should be and E is a running cost that describes, e.g., what are the constraint for the optimization of the time-dependent trajectories $\boldsymbol{u}(t)$. For example, the dynamical system should reach a certain point in the phase space at time T, and at the same time the energy that is used to control its dynamic evolution through the input $\boldsymbol{u}(t)$ should be minimal. Minimizing the functional with respect to $\boldsymbol{u}(t)$ requires that the minimum satisfies the relation $J^* \leq J(\boldsymbol{x}, \boldsymbol{u}, T)$ for any admissible, properly constrained functions $\boldsymbol{u}$ and $\boldsymbol{x}$. As an example for the meaning of these constraints, the force steering a wheel should not cause a vehicle to capsize. Alternatively, an electric signal should not exceed some values of amplitude and phase because it could damage the system, etc. In classical optimal control there are usually two approaches to the solution of control problems: closed-loop control (or feedback control), in which the action is modified as a result of measuring outputs from the environment, and open-loop control, in which the action is optimized independently from the system's output. In feed-forward control, instead, we build a model that anticipates disturbances of the system to predict and correct them. Formally, if a solution to the optimal control problem has a form $\boldsymbol{u}^*(t) = b(\boldsymbol{x}(t_0), t)$ [Kir04], it is said to be an open-loop solution, i.e., it is optimal only for a specific initial state. A closed-loop solution, instead, should be valid for arbitrary input states. The first model is extensively used in the theory of regulators, the most prominent ones being linear quadratic regulators [Son98], which deal with designing and optimizing feedback systems for engineering purposes using PID (Proportional Integral Derivative) controllers. The main disadvantage of such regulators is that they often require to linearize the dynamical system considered. A special case in optimal control theory is given by linear systems, e.g., systems whose dynamics is described by a linear inhomogeneous system of differential equations [Kir04]:

$$\dot{\boldsymbol{x}}(t) = A\boldsymbol{x}(t) + B\boldsymbol{u}(t). \tag{2.3}$$

Such linear systems of differential equations can be solved using time integration. The analysis of the resolvent operator of the system of differential equations (the so called state-transition matrix [Kir04]) can also be employed. In particular, the Laplace transform can be used to find the expression of the propagator analytically. If the control problem is digital, i.e., if it employs a difference equation with discrete time-steps instead of a differential equation in time, the Laplace transform becomes the Z-transform [Kir04], but the treatment is analogous. The optimization of the functional with respect to the control solutions uses different approaches. The theory of dynamic programming [Bel52] defines rigorously the differential equations for both discretized and continuous controls. Other approaches include calculus of variations, which leads to Pontryagin maximum principle [Pon18]. Some techniques, such as feedback through pole assignment [GEK19], can be applied both numerically and analytically. However, and in particular for non-linear systems, these approaches usually have to be integrated using numerical techniques and time-discretization methods, such as collocation [Yad20], which parametrizes continuous time-dependent problems using basis functions – e.g., interpolation polynomials in time with real coefficients – and maximizes the figure of merit of the control problem with respect to the polynomial parametrization.

2.2.1 Example: Trajectory optimization

We give here an example taken from Ref. [FGP&al21], a library for optimal control of various systems that uses trajectory optimization. We consider the famous reinforcement-learning problem of the Cartpole [BCP&al16], which is often used in artificial intelligence as a test bed to compare reinforcement learning algorithms. In this case, the problem is considered from the perspective of of optimal control theory. For a cost functional of the type [FGP&al21]:

$$J(u,T) = \int_{t_0}^{T} [u(t)]^2 \, dt, \tag{2.4}$$

where $u(t)$ is a time-dependent horizontal force that moves the cart around and can be used to balance the pole. The equations of motion of the system can be derived using Lagrangian mechanics. This cost function is not properly an energy functional, however it basically plays the same role in this context. The solution is provided in [Kel17]. The idea of collocation [Yad20] is now to represent the state vector of the system as a polynomial of a certain grade and then define the constraints for the trajectory optimization. First, the time dynamics is discretized in time steps $t_0, t_1, ..., t_N$, where $t_N = T$. Afterwards, the optimization algorithm optimizes the functional, which is now a discrete sum over the N samples, as a function of the polynomial coefficients and the control parameters. This methodology is used extensively in non-linear control problems and presents analogies with the current practical approaches to quantum optical control theory. In controlled quantum systems the non-linear dependence of the typical functionals from the control fields makes it unfeasible, in most of the cases, to find general, globally optimal analytical solutions to the control problems. As we outline in the following sections, ideas such as collocation are of fundamental importance to address problems in quantum optimal control. Sometimes, however, it is not easy to define the dynamics of the system in non-linear control: this is particularly true when dealing with abstract tasks in robotics, such as teaching a robotic hand to rotate certain objects [SNR&al21]. In such cases, machine learning algorithms, and in particular reinforcement learning – see Chapter 3, can be employed instead.

2.3 Quantum optimal control theory

Optimal quantum control (OQC) [Koc&al22; Wil&al20] deals with the optimization of quantum dynamics in a time-dependent setting, i.e., where the goal is the optimization of a functional rather than a simple cost function. Usually the experiment uses a quantum system that evolves according to its own internal degrees of freedom with a so-called *drift* Hamiltonian that describes the basic dynamics of the quantum system without external inputs. The drift Hamiltonian could describe, e.g., an Ising chain of nearest-neighbour interacting spins or a Heisenberg chain [MRS24], an an-harmonic oscillator with Kerr-type non-linearities [BGG&al21; Bou&al12], etc. Driving fields represent external inputs, such as laser and microwave fields, etc. The full time-dependent

Hamiltonian $H(t)$ of a (bilinear) quantum control problem with drift Hamiltonian H_0, associated *control* fields $u_1(t), ..., u_m(t)$ and *control* Hamiltonians $H_1, ..., H_M$ is given by:

$$H(t) = H_0 + \sum_{m=1}^{M} u_m(t) H_m. \tag{2.5}$$

The evolution of the corresponding (isolated) quantum system is given by the time-ordered propagator

$$U(T, t_0) = \hat{\mathcal{T}} \exp\left\{-i \int_{t_0}^{T} H(\tau) d\tau\right\}, \tag{2.6}$$

where $\hat{\mathcal{T}}$ is the time-ordering operator [SN20]. The evolution of the corresponding quantum state is given by

$$|\psi(T)\rangle = U(T, t_0) |\psi(t_0)\rangle, \tag{2.7}$$

in the case of pure states and

$$\rho(T) = U(T, t_0)\rho(t_0)U(t_0, T), \tag{2.8}$$

in the case of mixed states. When dealing with open quantum systems where the dynamics can be considered approximately Markovian, the time evolution is given by the Lindblad superoperator

$$\mathcal{L}_{\text{evo}}(\rho) = i[H(t), \rho] + \sum_j \left(L_j \rho L_j^\dagger - \frac{1}{2}\{L_j^\dagger L_j, \rho\} \right), \tag{2.9}$$

where L_j are specific *jump* operators [Ste07] which control the evolution of the density matrix

$$\frac{\mathrm{d}}{\mathrm{d}t}\rho = \mathcal{L}_{\text{evo}}(\rho), \tag{2.10}$$

according to the master equation. The goal of OQC algorithms and analytical methods is to find one or more optimal driving fields that allow us to perform different tasks:

- State preparation: Starting from an initial state $\rho(t_0)$, find the optimal dynamics such that the state at time T, $\rho(T)$ is equal to some target state ρ_{target}. A possible cost function for state preparation reads [Koc&al22]:

$$C = 1 - \mathrm{Tr}\left\{\rho(T)\rho^*_{\text{target}}\right\}. \tag{2.11}$$

In experiments, a figure of merit for quantum control needs to be estimated by measuring the overlap between the evolved state and a target state. This can be accomplished using, e.g., state tomographies [Cra&al10] or in some cases by measuring certain populations in the energy levels.

- Gate synthesis: Starting from an initial unitary propagator $U(t=0) = U_0$ (usually the identity) we want the unitary propagator $U(t=T)$ at time $t=T$ to match exactly a target unitary U_{target}. A special case of this problem is the creation of a perfect entangler [Wat&al15]. To generate a target gate we need to minimize the distance between the unitary operator generated by the dynamics and the target gate:

$$F = \frac{1}{d^2}\left|\mathrm{Tr}\left\{U(T)U^\dagger_{\text{target}}\right\}\right|^2. \tag{2.12}$$

  In experimental situations, measuring the gate fidelity can be challenging, particularly for large numbers of qubits. Existing strategies involve different types of tomography [NGR&al21], randomized benchmarking [KLR&al08; GMT&al12; CGT&al09; KBC&al14], estimators such as shadow tomography [HKP20], etc.

- Entanglement maximization: We want to maximize the amount of entanglement in a quantum state. Possible figures of merit to maximize entanglement for bipartite quantum states [PV07] are the concurrence and the entanglement of formation [Woo98] and in some specific cases – see also Ref. [SB22] – the entanglement entropy.

- Optimized transport of qubits (shuttling): This problem arises in different contexts, for example in solid-state systems, and in optical lattices, where the goal is usually to move spin qubits inside the solid-state device [DPS&al24] and cold atoms placed inside the lattice using, e.g., optical tweezers [MBD&al12].

Let us suppose that we want to control a quantum system in two different situations (single-qubit and two-qubit), as shown in Fig. 2.1. These single-qubit case can be solved analytically, at least in its simplest formulation. A general single-qubit system with time-dependent fields is given by:

$$H(t) = \sigma_x u_1(t) + \sigma_y u_2(t) + \sigma_z u_3(t), \tag{2.13}$$

which can represent any single-qubit rotation [NC10]. This approach is theoretically valid, but in practice ignores leakage in the higher energy levels that are typical of, e.g., superconducting qubits [MGR&al09; LCM22]. We want now to consider the state-preparation task of steering the dynamics from the ground state at $|\psi(t=0)\rangle = |0\rangle$ to the state $|\psi(t=T)\rangle = |1\rangle$. In Fig. 2.1 (a) the single-qubit is evolving according to Eq. (2.13). An example of control signal is given by a Gaussian pulse on the Pauli σ_x component for which we have $\int_0^T u_1(t)dt = \frac{\pi}{2}$ and $u_2(t) = u_3(t) = 0$. This will implement a σ_x operation on the qubit and will therefore invert the population. This is exactly the

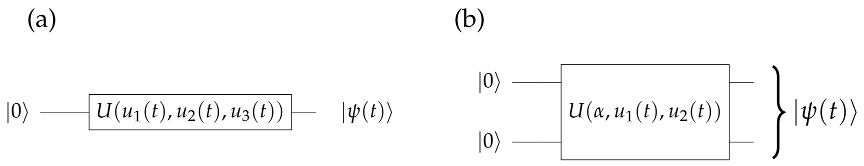


Figure 2.1: Representation of two quantum control problems: single-qubit (a) and two-qubit (b). The goal is to prepare a quantum state using time-dependent control fields. In (a) the qubit is simply evolving according to a unitary $U(u_1(t), u_2(t), u_3(t))$ generated by the Hamiltonian in Eq. (2.13), so the state preparation problem has an immediate analytical solution, while the gate synthesis problem given in Eq. (2.14) and (b) is less trivial, but it can be solved using OQC methods [KRK&al05; CCM11; GBG&al19].

NOT gate used in classical logic. If we instead implement a $\frac{\pi}{4}$-pulse, the state will end up in an eigenstate of the σ_x operator, i.e., a superposition of $|0\rangle$ and $|1\rangle$. As we see, these control problems can be solved immediately in the single qubit case. However, they can become more challenging if higher levels are considered [MGR&al09]. Let us now consider the two-qubit system – see Fig. 2.1 – (b) and, e.g., a Hamiltonian of the type:

$$H(\alpha,t) = \alpha\left[\sigma_x \otimes \sigma_y + \sigma_y \otimes \sigma_x\right] + u_1(t)(\sigma_z \otimes \mathbb{I}_2) + u_2(t)(\mathbb{I}_2 \otimes \sigma_z), \tag{2.14}$$

where $\mathbb{I}_2$ is the single-qubit identity matrix and $u_1(t)$ and $u_2(t)$ are driving fields. The problem is to find optimal shapes for such pulses to generate a specific parametric two-qubit gate, such as a cross-resonance (CR) gate [MMM20; KKL&al18; LCM24]. Another interesting question is whether it possible to construct parameter-dependent pulses $u_1(\alpha,t)$ and $u_2(\alpha,t)$ that generate a target gate for every value of the parameter α. The last topic, in particular, is discussed in depth in Chapter 4. In the next section, we introduce some of the most relevant approaches in quantum optimal control theory. Such approaches aim to solve the pulse optimization problem exactly on arbitrary quantum systems.

2.3.1 Pulse shaping and time discretization

Quantum control deals with the optimization of the quantum dynamics according to a figure of merit, e.g., the gate or state fidelity. In numerical approaches to quantum optimal control problems, a time discretization for the control fields can be defined, such as $t_0, t_1, ..., t_N = T$, where, e.g., $t_i = t_0 + i\frac{T}{N}$. We see here the similarity between the approaches of quantum optimal control and non-linear classical optimal control. The pulse profiles are usually also discretized in values $u_{j0} = u(t_0), u_{j1} = u(t_1), ..., u_{jN} = u(t_N)$ for $1 \le j \le M$. A standard approach is to divide the total evolution time T in N equal time intervals $\Delta t = \frac{T}{N}$ and use as parametrization a step function of the type

[MGM&al11]:

$$u_j(t) = \sum_{k=1}^{N} f_k(t)\theta_{jk}, \tag{2.15}$$

with pulse values $\theta_{jk} \in \mathbb{R}^{M\times N}$, where

$$f_k(t) = \begin{cases} 1, & t_{k-1} \le t \le t_k \\ 0, & \text{otherwise} \end{cases} \tag{2.16}$$

is a rectangular function acting on the interval $[t_k, t_{k+1})$ with $1 \le k \le N$. This approach uses N pulse values of length Δt. However, this type of pulse shaping gives often rise to discontinuous pulses and therefore requires a very fine time grid. An alternative to this parametrization is to, similar to the collocation ansatz, find an appropriate time-dependent basis to model the evolution of the driving fields. A common parametrization in optimal control of quantum systems is the Fourier basis, which uses sinusoidal waveforms:

$$u_j(t) = \sum_{k=1}^{K} \theta_{jk} \sin\left[\frac{k\pi(t-t_0)}{N}\right], \tag{2.17}$$

with Fourier amplitudes $\theta_{jk} \in \mathbb{R}^{M\times K}$. Eq. (2.17) represents the Fourier expansion of the control pulse, assuming $u(t_0) = 0$ and $u(t-t_0) = -u(t_0-t)$. This parametrization starts and ends in zero, which can be beneficial in some experimental implementations. Further examples of parametrizations are wavelets [Mal98] and the sinc function [PMC&al24]. In this formulation, the control problem is characterized by the parameter vector $\boldsymbol{\theta}$, whose dimension depends on the nature of the basis functions. Once the parametrization is defined, the goal of the opimization routine is to maximize the figure of merit with respect to the parameter vector $\boldsymbol{\theta}$.
Several of the known theoretical solutions in optimal control assume a perfect correspondence between the theoretical pulse shapes and the pulses that act on the experimental setup. In most of the experimental scenarios, however, it is necessary to consider distortions in the control lines [GZB&al13] caused by the electronics. These distortions can often be modeled as the effect of a convolution with a filter function ϕ:

$$v(t) = \int_{-\infty}^{+\infty} \phi(t-t')u(t')dt', \tag{2.18}$$

where $v(t)$ is the output pulse distorted by the filter and $u(t)$ is the input pulse. These distortions can be compensated if the filter can be modelled appropriately [SZC&al23]. The corresponding transfer function of the distortion can also be included in the numerical simulation of the optimal control problem and its subsequent optimization. Modelling the effects of filters can actually increase the chances of succeeding in discovering a high-fidelity solution that has experimental relevance. In fact, an optimal quantum control model that takes distortions into account can hopefully better represent the physical reality that experimental physicists are confronted with on a daily basis.

2.4 Algorithms

2.4.1 Gradient-based optimization

Given the pulse in an appropriate parametrization, the question arises whether it is possible to extract the gradient of the figure of merit with respect to the pulse parameters. Gradients are the basic ingredient of some of the most efficient optimization algorithms available today [Rud17; LN89; KB15]. Let us consider for examples the gate fidelity given in Eq. (2.12). Its derivative with respect to a pulse parameter θ_i, one of the components of the parameter vector $\boldsymbol{\theta}$, that governs the quantum dynamics is given by:

$$\frac{\partial F}{\partial \theta_i} = \frac{2}{d^2} \operatorname{Re}\left\{ \operatorname{Tr}\left\{ U_{\boldsymbol{\theta}}(T) U_{\text{target}}^{\dagger} \right\} \operatorname{Tr}\left\{ \frac{\partial U_{\boldsymbol{\theta}}(T)}{\partial \theta_i} U_{\text{target}}^{\dagger} \right\} \right\}. \tag{2.19}$$

The problem is now to calculate the gradient of the propagator U defined in Eq. (2.6). To simplify the problem, it is often assumed that the controls are piecewise constant pulses over a time span Δt – see Eq. (2.15). Solvers for quantum dynamics based on Trotterization [Tro59] make use of products of unitary gates to compute the dynamics. In Trotter-based approaches, the time-ordered propagator is written as a product of N unitaries corresponding to a time interval $t_i - t_{i-1} = \Delta t$. More sophisticated approximations make use of the Magnus expansion [BCO&al09; DM22] or the Dyson series in the interaction picture [SGD&al21]. The use of ODE solvers such as Runge-Kutta is also possible, but it may violate the unitarity of the dynamics. Assuming that the unitary propagator can be written as a product of unitaries with sufficient precision, the gradient can be computed separately for each piecewise constant pulse:

$$U_{\boldsymbol{\theta}}(T, t_0) \approx \prod_{i=1}^{N} U_{i,\boldsymbol{\theta}^{(i)}}(\Delta t), \tag{2.20}$$

using a unitary Trotter step

$$U_{i,\boldsymbol{\theta}^{(i)}}(\Delta t) = \exp\left\{ -iH_0 \Delta t - i \sum_{m=1}^{M} \theta_{im} H_m \Delta t \right\}, \tag{2.21}$$

where $\boldsymbol{\theta}^{(i)} = (\theta_{i1}, ..., \theta_{iM})^T$ is the ith control pulse vector and $\theta_{im} = u_m(t_i), 0 \leq i \leq N, 1 \leq m \leq M$. In this formulation, the gradient-based optimization algorithm requires the gradient of each unitary step $U_i(\Delta t)$. In fact, the control parameters are local, i.e., the ith and mth control parameter only appears in the unitary step $U_i(\Delta t)$. Differentiating Eq. (2.21) with respect to θ_{im} results in the analytical formula [MSG&al11; Aiz63;

KRK&al05; DMJ&al20]:

$$\langle\lambda_k| \frac{\partial U_{i,\boldsymbol{\theta}^{(i)}}}{\partial\theta_{im}} |\lambda_j\rangle = \begin{cases} -i\Delta t \langle\lambda_k| H_m |\lambda_j\rangle e^{-i\Delta t\lambda_k}, & \text{if } \lambda_k = \lambda_j \\ \langle\lambda_k| H_m |\lambda_j\rangle \frac{e^{-i\Delta t\lambda_k} - e^{-i\Delta t\lambda_j}}{(\lambda_k - \lambda_j)}, & \text{if } \lambda_k \neq \lambda_j, \end{cases} \tag{2.22}$$

where the values λ_k are the n_λ eigenvalues and $|\lambda_k\rangle$ the corresponding eigenvectors of U_i for $0 \leq k \leq n_\lambda$. This expression requires knowing the eigenvalues of the full Hamiltonian for each time step and uses diagonaliztion, which is computationally expensive. For non-piecewise constant pulses – see Eq. (2.17) – an option is to treat again the pulse as piecewise constant, with the only difference that now the piecewise constant values are interpolated using an appropriate function basis – e.g., a truncated Fourier basis – evaluated at discrete points $t_i, 0 \leq i \leq N$. The gradient can then be computed using the chain rule on Eq. (2.22) [MGM&al11]. Alternatives are numerical differentiation strategies or the analytical formula, i.e., [LPQ&al24]:

$$\frac{\partial U_{\boldsymbol{\theta}}(T)}{\partial\theta_i} = -i \int_{t_0}^{T} U_{\boldsymbol{\theta}}(T,\tau) \frac{\partial H_{\boldsymbol{\theta}}(\tau)}{\partial\theta_i} U_{\boldsymbol{\theta}}(\tau, t_0) d\tau, \tag{2.23}$$

which under certain conditions can be evaluated on quantum devices using Monte Carlo integration [LPQ&al24; KK23]. Gradient-based optimization of optimal quantum control problems can be performed using, e.g., GRAPE [KRK&al05] or Krotov [GBG&al19]. The GRAPE algorithm computes the full gradient of the fidelity with respect to the figure of merit using backpropagation. Nonetheless, implementing this algorithm in closed-loop on physical devices is particularly challenging, because we cannot naively perform backpropagation in quantum experiments – see also the analysis given in Ref. [BWP24]. The GRAPE algorithm uses concurrent update of all the piecewise constant control fields, while the Krotov algorithm updates them sequentially. Ref. [MSG&al11] analyzes the performance of concurrent and sequential updates of the control values in several control problems. Overall, applying gradient descent concurrently over the entire control pulse seems to be more efficient in most of the problems considered.

2.4.2 CRAB

Quantum control methods based on either solving (numerically or analytically) Lagrange equations [BCR10] or computing the gradients of the quantum cost function with respect to the pulse parameters (GRAPE, Krotov) can produce useful ansätze for experimental purposes. However, their direct implementation on real quantum hardware implies that both the cost functions and their derivatives can be estimated with sufficient precision [KK23; LPQ&al24] from experimental measurements. Such estimators can be constructed for arbitrary unitary dynamics [WIW&al22; WLW&al24], they are however only efficient for gates that have highly degenerate spectra, which is generally not the case for the unitaries of quantum control problems. In fact, the number of shifts needed to evaluate the partial derivative with respect to a variational parameter scales linearly with the number of distinct eigenvalues of the quantum gate – see Section 2.7.1

and Eq. (2.32) in particular. In the case of time-depedent Hamiltonians, an additional level of complexity is present, as the derivative of the cost function with respect to the pulses is proportional to the integrated time dynamics, which makes it particularly challenging to sample the gradient in reasonable time. Furthermore, multiple sources of errors present at the level of the experiment (including cross-talk between qubits and control lines [ZBL23], coherent errors on the qubits, leakage in higher energy levels [MGR&al09], SPAM errors [HFW20], etc.) make it challenging to achieve the desired numerical precision in the sampling, therefore preventing the gradient-based optimization from converging to the desired optimum. As a result, gradient-free methods represent a valid alternative for quantum control experiments. More specifically, the CRAB approach (Chopped RAndom Basis) [CCM11; MSJ&al22] to quantum optimization allows for gradient-free control by constructing an appropriate randomized ansatz. The idea is to first choose a suitable basis to represent the time-dependent control pulse, e.g., a (truncated) Fourier basis ansatz or a polynomial ansatz:

$$u(t) = g(t) \sum_{k=1}^{K} a_k f_k(\omega_k, t), \tag{2.24}$$

where f_k are suitable basis functions with relevant parameters ω_k that can approximate any time-dependent function u on $[t_0, T]$ and $g(t)$ is a suitable envelope function that enforces the pulse constraints for $t = t_0$ and $t = T$. Here the maximum number of basis components used, K, is a free parameter of the problem and it is usually bounded from above by the experimental constraints of the quantum system. We consider here only the case in which a Fourier basis is used. Similar approaches can be used to include other types of control bases, such as systems of orthogonal polynomials. The parameters ω_k for the Fourier basis can be chosen to be the principal harmonics $\omega_k = \frac{2\pi k(t-t_0)}{T}$, they are however randomized during the optimization to improve the convergence of the algorithm: $\omega'_k = \omega_k(r_k + 1)$, where $r_k \sim \mathcal{U}\left(-\frac{1}{2}, \frac{1}{2}\right)$ [CCM11]. If more detailed information about the physical system is available, it can be used to improve the standard randomized ansatz. The CRAB ansatz can be further enhanced to construct a meta-optimization method that tries to search for the optimal pulse in the control landscape by varying the number and the structure of the Fourier basis functions. This procedure is known as the dCRAB method [RMC&al15]. In this approach, on top of the main optimization routine, a meta-optimization loop with steps $j = 1, ..., N_{\mathrm{dCRAB}}$ is introduced to modify the number and the structure of the function basis [MSJ&al22]:

$$u^{(j)}(t) = c_0^{(j)} u^{(j-1)}(t) + \sum_{k=1}^{K_j} c_k^{(j)} f_k^{(j)}(t), \tag{2.25}$$

where $u^{(j-1)}$ is the optimized pulse from the previous step of the external loop and $f_i^{(j)}$ are the new basis functions at the super-iteration j. The coefficients for each step of the meta-optimization loop are given by $c_i^{(j)}$ for $i = 0, ..., K_j$. The number of Fourier components K_j can be constant at each super-iteration, i.e., $K_j = K$ or it can be progressively increased if the cost function minimum does not fall below a desired threshold. The CRAB framework has been also extended to work remotely over the internet (redCRAB [HVS&al18]) using ansätze provided by remote users in the context of the so-called gamified citizen science [HVS&al18].

2.5 Adiabatic control techniques

In the context of optimal quantum control, several analytical solutions to produce different types of quantum gates and states have been discovered. The nature of these solutions vary depending on the quantum computing platform considered. The adiabatic theorem [Ber87] offers us the possibility to construct such solutions based on the principle that for slow time evolution, a system starting in an eigenstate will end up in the same eigenstate of the Hamiltonian and all the effects of the time dynamics will be contained in a phase [SN20; TMM&al18]. Most of the analytical solutions attempt to find ansätze that can, e.g., produce a particular entangling gate using the Hamiltonian available in the system [SM99] or suppress specific sources of errors [LCM24; MM22]. At the same time, analytical solutions usually allow for the optimization of different quantum gates using experimental free parameters.

2.5.1 Analytical pulses

Adiabatic transformations [TMM&al18; AL18] can be used to design optimal control pulses. In particular superadiabaticity [TMM&al18; Ber87; DKM&al08] provides a framework to suppress unwanted terms in the Hamiltonian interaction by performing iterative frame transformations. Each transformation can potentially remove a specific error term and introduce a less detrimental one. This procedure cannot be repeated indefinitely, as the expansion diverges for a large number of iterations [TMM&al18]. The condition for the transformation to be successful can be connected to certain conditions in the control pulses. Let us, e.g., consider a single-qubit transmon device with leakage [MGR&al09]. In the interaction frame, after performing a rotating wave approximation (RWA), this device can be represented as a three-level system addressed by microwave pulses:

$$H(t) = \delta_1 |1\rangle\langle 1| + \delta_1 |2\rangle\langle 2| + \frac{u_1(t)}{2}(\sigma_x^{(01)} + \lambda\sigma_x^{(12)}) + \frac{u_2(t)}{2}(\sigma_y^{(01)} + \lambda\sigma_y^{(12)}), \tag{2.26}$$

where δ_1 and δ_2 are the detunings. The transitions $|0\rangle - |1\rangle$ and $|1\rangle - |2\rangle$ have frequencies ω_1 and ω_2 and are driven by a signal with carrier frequency ω_d, s.t. $\delta_1 = \omega_1 - \omega_d$ and $\delta_2 = \alpha + 2\delta_1$, where α is the anharmonicity of the transmon. The operators $\sigma_x^{(ij)} = |i\rangle\langle j| + |j\rangle\langle i|$ and $\sigma_y^{(ij)} = i(|j\rangle\langle i| - |i\rangle\langle j|)$ describe the transitions between the energy levels. The signals $u_1(t)$ and $u_2(t)$ are the quadratures of the signal and λ quantifies the effectiveness of the signal in driving each transition [MGR&al09]. The goal is to find control pulses $u_1(t)$ and $u_2(t)$ that can maximize the fidelity of the corresponding unitary evolution $U(T)$ for $t_0 = 0$ – see Eq. (2.6) – with respect to a target single-qubit gate and efficiently suppress the unwanted leakage in the higher energy levels. The DRAG (Derivative Removal by Adiabatic Gate) ansatz is an analytical perturbative solution to the problem presented above that makes use of the following adiabatic unitary transformation [MGR&al09]:

$$V(t) = \exp\left\{-\frac{i}{2\alpha}u_1(t)(\sigma_y^{(01)} + \lambda\sigma_y^{(12)})\right\}, \tag{2.27}$$

applied on the Hamiltonian in Eq. (2.26), which transforms according to the known rule $H' = VHV^\dagger + i\dot{V}V$. If one expands the resulting expression as a function of the parameter λ, one obtains that for the leakage to be suppressed, the quadratures $u_1(t)$ and $u_2(t)$ and the detuning $\delta_1(t)$ have to fulfill the following equations [MGR&al09]:

$$\delta_1(t) = \frac{(\lambda^2 - 4)}{4\alpha}u_1^2(t),\ u_2(t) = -\frac{1}{\alpha}\frac{\partial}{\partial t}u_1(t). \tag{2.28}$$

This expression can be used in combination with an appropriate ansatz for the quadrature $u_1(t)$, whose parameters can be optimized with an appropriate algorithm, such as GRAPE [KRK&al05]. Higher order corrections can be applied to obtain more refined expressions for the pulse shapes. This approach has been successfully generalized in the form or *recursive* DRAG to optimize two-qubit entangling gates [LCM22; LCM24] in superconducting quantum systems.

2.6 GOAT

In the method GOAT [MAT&al18], the differential equation for the derivative of the quantum state or gate with respect to the pulse parameters is propagated forward in time together with the state. The differential equation [Goe15] reads:

$$\frac{\partial}{\partial t}\begin{bmatrix} U_\alpha(t,t_0) \\ \frac{\partial}{\partial\alpha}U_\alpha(t,t_0)\end{bmatrix} = -i\begin{bmatrix} H(t,\alpha) & 0 \\ \frac{\partial}{\partial\alpha}H(t,\alpha) & H(t,\alpha)\end{bmatrix}\begin{bmatrix} U_\alpha(t,t_0) \\ \frac{\partial}{\partial\alpha}U_\alpha(t,t_0)\end{bmatrix}, \tag{2.29}$$

with $U_\alpha(t,t_0) = \hat{\mathcal{T}}\exp\left\{-i\int_{t_0}^{t} H(\tau,\alpha)d\tau\right\}$. The term H is any time-dependent Hamiltonian that depends on a parameter α, such as, $H(t,\alpha) = H_0 + \sum_{m=1}^{M} u_m(\alpha,t)H_m$. We see that this method requires to calculate the propagator of a $2n\times 2n$ time-dependent matrix instead of a $n\times n$ matrix, which is more time-consuming. Because the gradient of a cost function such as the state overlap or the gate fidelity depends on the unitary $U_\alpha(t,t_0)$, Eq. (2.29) can be used to calculated the gradient of the fidelity. The original proposal suggests using a Runge-Kutta solver to integrate the expanded Schrödinger equation. However, any solver can be implemented in principle.

2.7 Variational quantum circuits

In recent years, due to the increased interdisciplinary interest in quantum computers from both academia and industry, some approaches have been developed to exploit the computational power of quantum processors even in their current or near-future NISQ format. The goal is to use these devices to address relevant problems in chemistry, drug design, classical computer science, finance and artificial intelligence. In general, solving an optimization problem on a quantum computer requires the preparation of an appropriate n-qubit ansatz consisting of entangling and single-qubit gates, which depend on so-called variational parameters. Mathematically, these parameters are rotation angles of unitary gates. From a physical point of view, the parameters are physically tunable quantities such as free parameters of laser or microwave pulses used to drive the qubits. An example of a parametric circuit model would be a circuit structure composed of layers of pairwise entangling gates and layers of single-qubit rotations [FGG14], but several other ansätze have been developed [Cho&al21; MAG&al21]. The other element needed in a variational algorithm is the Hamiltonian that encodes the problem to solve. The Hamiltonian can usually be represented in terms of fermionic interactions and then mapped to spins (qubits) using the Jordan-Wigner transformation [JW28; SOG&al02]. The expected value of the energy can be estimated using multiple instantiations of quantum circuits or by using the LCU method [CW12].
Due to the presence of the aforementioned variational parameters, the expected value of the energy of the problem Hamiltonian depends now on the variational angles. As a result, we can in principle minimize the estimated quantity with respect to the parameters to find the ground state of the Hamiltonian. The QAOA approach [FGG14], which is particularly ambitious, attempts to solve hard optimization problems, such as the MAX-CUT or the traveling-salesman problem [CK97], using a variational circuit. For a general overview of variational quantum algorithms, see Ref. [Cer&al21b]. A variational quantum cost function is a mapping

$$C(\boldsymbol{\theta}) = \mathrm{tr}\Big\{V(\boldsymbol{\theta})\rho V^{\dagger}(\boldsymbol{\theta})\mathcal{O}\Big\}, \tag{2.30}$$

that encodes a specific optimization problem using multiple quantum circuits and real parameters $\boldsymbol{\theta} \in \mathbb{R}^{n_p}$, an observable $\mathcal{O}$, an input density matrix ρ and a n-qubit variational quantum circuit $V(\boldsymbol{\theta}) \in \mathrm{U}(d)$, $d = 2^n$, given by the product of L parametric gates V_l:

$$V(\boldsymbol{\theta}) = \prod_{l=1}^{L} V_l(\boldsymbol{\theta}_l), \tag{2.31}$$

where $\boldsymbol{\theta} = (\boldsymbol{\theta}_1, ..., \boldsymbol{\theta}_L)^T$. The dimension of the parameter space n_p depends on the circuit structure, i.e., how many gates are employed per layer and whether each gate has more than one parameter, which is the case in quantum optimal control [MAG&al21]. Generally, the number of parameters scales linearly with the number of layers L.
Ideally, by optimizing over the parameter vector $\boldsymbol{\theta}$, the expected value of the energy should reach its minimum, which corresponds to the solution of the optimization prob-

lem. Due to the relatively low depths of these circuits compared to quantum algorithms such as Grover search or phase estimation, there is hope that variational algorithms could provide quantum speedup already in noisy devices. However, the initial ambitions of these approaches were frustrated by the presence of Barren plateaus [McC&al18], which render the optimization exponentially hard as the number of qubits and the complexity of the problems considered increases. In particular, it has been argued that optimization problems for QAOA either exhibit Barren plateaus or are classically simulable [CLG&al24]. Overcoming the limitations represented by Barren plateaus has become an important research direction in variational quantum algorithms [LTW&al24; VC21; Cer&al21a].

2.7.1 Optimization and gradients for variational circuits

A variational algorithm first estimates the energy of a quantum system as a function of variational parameters $\boldsymbol{\theta}$. Therefore, the algorithm must employ a suitable estimator for the parameter-dependent observable. Afterwards, the optimization is carried out with respect to the parameters. The parameter-dependent energy of the encoded optimization problem is often referred to as quantum cost function. If one treats the so-called quantum cost function as a black box, it is possible, in principle, to use a suitable gradient-free optimization method [OGB21; CCM11] to minimize it. However, such methods often show poor performance for large numbers of parameters [MSG&al11]. The alternative is given by gradient-based methods. The problem here is how to estimate the gradient of the aforementioned cost function. For a standard gate set composed of single-qubit Pauli rotations $R_x(\theta) = \exp\{-i\sigma_x\theta\}$, $R_y(\theta) = \exp\left\{-i\sigma_y\theta\right\}$ and $R_z(\theta) = \exp\{-i\sigma_z\theta\}$ and a CNOT gate, this problem can be solved elegantly using a so-called parameter-shift rule [Li17; WIW&al22]. With the same method, one can also sample higher derivatives and estimate the Hessian [WIW&al22; KE21], which is needed for methods such a BFGS [BRO70]. For more general gate sets with structure $V(\theta) = \exp\{-iH\theta\}$, it is possible to generalize the aforementioned approach by using trigonometric interpolation. To implement such interpolation schemes, the eigenvalue decomposition of the gate Hamiltonians needs to be known. The gradient is then given by:

$$\frac{\partial C(\theta)}{\partial \theta_i} = \sum_{i=1}^{R} a_i C(\theta + s_i), \tag{2.32}$$

where s_i are the parameter shifts and a_i are appropriate real coefficients that are related to the eigenvalues of H – see also [WIW&al22; BWK22] and Chapter 5. The minimum total number of shifts R depends on the number of eigenvalues. However, we see that parameter-shift rules scale linearly with the total number of parameters and the number of distinct eigenvalues of the gate Hamiltonians. This is unavoidable for this type of estimation, since in a general quantum circuit, such as the one in Eq. (2.32) each parameter has to be shifted independently while the others are kept constant. Theoretically, parameter-shift rules could also be parallelized embarrassingly on multiple quantum processors, similarly to what can be accomplished for evolutionary strategies

[WSG&al14; SHC&al17] and finite-difference methods in classical optimization. For general gates $\exp\{-iH(\boldsymbol{\theta})\}$ with parameter vector $\boldsymbol{\theta}$ and where $H(\boldsymbol{\theta})$ is an arbitrary parametric Hamiltonian, the gradient estimation procedure uses the properties of the exponential map [Aiz63]. Therefore, the construction of the adjoint operator [WLW&al24] becomes necessary. An alternative approach could be represented by strategies that use unitary interpolation [SPM&al24].

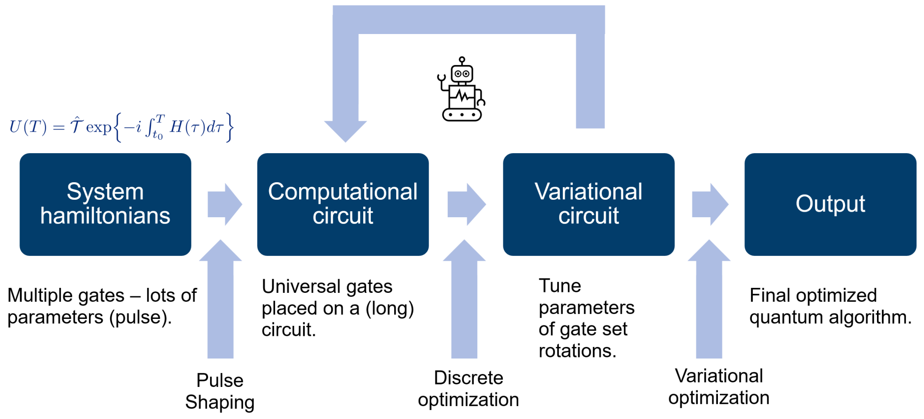

$U(T) = \hat{\mathcal{T}} \exp\left\{-i\int_{t_0}^{T} H(\tau)d\tau\right\}$
System hamiltonians
Computational circuit
Variational circuit
Output
Multiple gates – lots of parameters (pulse).
Universal gates placed on a (long) circuit.
Tune parameters of gate set rotations.
Final optimized quantum algorithm.
Pulse Shaping
Discrete optimization
Variational optimization

Figure 2.2: A representation of an optimization workflow between quantum control and compilation. First, the pulse parameters need to be optimized to generate an appropriate set of entangling gates. Some parameters can be ignored in the initial quantum control optimization, as we may want the pulse to represent a family of quantum gates. Afterwards, one can build a computational circuit which is composed by multiple layers of entangling and non-entangling gates acting on several qubits. Discrete optimization algorithms can be used to prune the initially large circuit to reduce its size while keeping its fidelity high. Meanwhile, some remaining variational parameters can be set to appropriate values. The resulting optimized quantum algorithm is the product of multiple layers of optimization that can potentially interact with each other [YLB21; SEL&al22]. A discrete optimization algorithm – graph-traversal algorithms [HN19] , reinforcement learning [Mor&al21], etc. – can interact with continuous optimization algorithms – dCRAB [CCM11], LBFGS [LN89], etc. – to improve the final solution iteratively. The image was realized by the author using PowerPoint™.

In some cases, we may want to connect the parameters of a set of quantum gates to some measurable quantities external to the problem, in order to establish a connection between the parameters themselves and, e.g., the physical quantum computing setup. In this sense, the new gate parameter α will exhibit a functional dependence from a collection of external parameters $\boldsymbol{\lambda}$ and variational parameters $\boldsymbol{\theta}$, that is

$$\alpha := f(\boldsymbol{\theta}, \boldsymbol{\lambda}), \tag{2.33}$$

where f is a suitable function, e.g., a linear model or a neural network. Let us further suppose that we want to determine the optimal function parameter $\boldsymbol{\theta}$. As we will see in the context of SOMA (Single-Optimization-Multiple-Application) – see also Chapter 4, there are two different ways of achieving this goal: By performing a regression over input-ouput data coming from the experiment, or by training a model directly on the cost function. We want to consider here the second case, since it is the one where parameter-shift rules may be actually beneficial. Using the chain rule on the quantum cost function in Eq. (2.30), we obtain:

$$\frac{\partial C(\alpha = f(\boldsymbol{\theta}, \boldsymbol{\lambda}))}{\partial \theta_i} = \frac{\partial C(\alpha)}{\partial \alpha} \frac{\partial \alpha}{\partial \theta_i}\bigg|_{\alpha = f(\boldsymbol{\theta}, \boldsymbol{\lambda})}, \tag{2.34}$$

where the first gradient is evaluated using parameter-shift rules, and the second gradient can be computed classically using backpropagation. Therefore, we can relate the gradient with respect to the rotation angles to the gradient computed with respect to the function parameters. This allows us to compute derivatives for problems that involve meta-optimization. As we will see in Chapter 4, these problems are relevant for modern applications, for example in cases where our goal is to optimize a family of control pulses, quantum gates or Hamiltonians.

2.8 Quantum compilation

Quantum compilation [Mar&al22] deals with the execution of quantum algorithms on real quantum hardware. Usually, quantum algorithms are given in terms of abstract, multi-qubit unitaries, such as the Quantum Fourier Transform [NC10; Sho94], the Grover operator [Gro96], a quantum simulation algorithm such as Trotterization [Tro59], etc. Such unitary operations are generally not available on quantum devices. Hence, complex quantum algorithms need to be decomposed into layers of more basic operations. Similarly, a classical processor decomposes a complex numerical algorithm in basic operations such as addition and subtraction. In any quantum processor there is a transition from the hardware level, where qubits are driven with electromagnetic fields, to the more abstract mathematical level of unitary operations and digital gates – see also Fig. 2.2. At the lowest level, on the left in Fig. 2.2, we have quantum control pulses, which can be used to generate digital or analog gates. An example of a framework that can be used to optimize different classes of gates using quantum control in *QuTip* [JNN13] is given in Ref. [LAS&al22]. On an intermediate level, quantum control can also be used solely to implement a universal set of gates with few free parameters – see also Chapter 4. Once the gate set is available, the Solovay-Kitaev algorithm [DN06] provides us with an efficient approach to decompose an arbitrary unitary using universal gates. This approach can be combined with graph-traversal algorithms [HN19; YIL&al21]. Some more recent approaches make use of reinforcement learning – see also Chapter 3 – and deliver highly optimized solutions at the cost of long training times [SEL&al22; Ost&al21; Mor&al21]. An optimization method based on both reinforcement learning and continuous optimization of variational angles is introduced in Chapter 5. This method lies at the intersection between full experimental quantum control and discrete optimization of digital gates. At the highest level, on the right in Fig. 2.2, we have a full (digital) quantum algorithm, such as amplitude amplification or the Shor's algorithm. The overall fidelity of the final quantum circuit depends on the efficiency of all the methods discussed above. To conclude, improving and developing new optimization methods for quantum circuits and quantum dynamics is a fundamental step in the path towards fault-tolerant quantum computation.

3 Machine Learning

3.1 Introduction

Machine learning [Alz&al21; Bis07; GBC16] is a domain of artificial intelligence that studies algorithms to approximate functions using data. Due to the importance of learning data patterns in statistics long before the most recent developments in artificial intelligence as an autonomous research field, several terms that find application in machine learning often find correspondence in traditional statistics [Was10]. For example, the process of *learning* from data is usually described as data fitting in statistical contexts. The reason for these discrepancies are arguably due to the different questions that these two fields seek to answer and their parallel development throughout the years: on the one hand, machine learning usually tries to underline the connection between learning a functional mapping from a data set and, e.g., neurobiological processes in mammal's brains [Hay09] and sea creatures [Ant&al03] or psychological concepts, such as learning from rewards. In statistics, instead, the question is often about how to suitably represent the available data or, e.g., whether a certain hypothesis about patterns in the data can be rigorously verified with sufficient precision. This difference in focus is often reflected in the terms used in the two fields: for example, in statistics, the term *(test set) validation* is used to talk about the performance of a trained function or model on new data. The goal here is simply to test whether the model really represents the data or not. But in machine learning the term used is *generalization* [HTF09], a more abstract notion that envisions an agent trying to first gather some knowledge about a concept and then trying to apply this knowledge on new, unseen phenomena. Mathematically, this concept is often formalized in the so-called generalization error [Hay09], or generalization gap, that quantifies the performance of the learning algorithm on novel inputs. The two fields, statistics and machine learning meet and partially merge in the context of statistical learning theory [HTF09], which provides a rigorous framework for machine learning models.
Leaving aside psychology and neurobiology, which despite their importance are outside the scope of our discussion, another field of research shows deep connections with both machine learning and statistics, i.e., control theory [Kir04]. In fact, control systems often needs to gather data about sensors and other electromechanical components inside industrial facilities or engines. The data has to be then used so to adapt the system parameters to the underlying changing conditions, which often involves the optimization of parametric models, such as neural networks. A summary of the differences in terminology between machine learning and statistics is given in Table 3.1 and can be found in Ref. [Was10] in a more comprehensive version.

Statistics	Machine learning
estimation	learning
classification	supervised learning
clustering	unsupervised learning
covariates	features
classifier	hypothesis

Table 3.1: A comparison of some of the terms used in machine learning and statistics to describe similar concepts (modified from Ref. [Was10]).

3.2 Multi-layered perceptrons

In deep learning, one usually relies on families of functions $f : \mathbb{R}^d \times \mathbb{R}^{n_p} \mapsto \mathbb{R}^k, (\boldsymbol{x}, \boldsymbol{\theta}) \mapsto f(\boldsymbol{x}; \boldsymbol{\theta})$, where $\boldsymbol{\theta} \in \mathbb{R}^{n_p}$ are real parameters [FHI&al18]. The families of functions considered usually in deep learning tasks are neural networks. The reason is that these computational models have been shown empirically to perform extremely well in an enormous number of tasks [Alz&al21; Mni&al15; Sil&al16; Jum&al21; CT17] due to generalization capabilities and are comparatively easy to train thanks to the improvements in computation, in particular in GPU computing. Feed-forward neural networks are also universal function approximators [HSW89]. The simplest neural network can be described by an affine function applied on the input values, followed by a non-linear function σ, called activation function:

$$y_i = \sigma\left(\sum_{i=1}^{d} W_{ij} x_j + b_i\right) \tag{3.1}$$

$$\boldsymbol{y} = \sum_{i=1}^{k} y_i \hat{e}_i. \tag{3.2}$$

The matrix entries $W_{ij}, 1 \le j \le d, 1 \le i \le k$ multiplied with the input values are referred to as weights, whereas the values $b_j, 1 \le i \le k$ represent the bias vector. The (feed-forward) neural network has parameters $\boldsymbol{\theta} = (\bar{\boldsymbol{W}}, \boldsymbol{b})$ and $n_p = k(d+1)$. Eq. (3.1), for the case $i = 1$, i.e., with just one output values, shows the original model of Rosenblatt's perceptron [Ros58], where the activation function is a sigmoid. Other activation functions, such as ReLUs [Aga19], SeLUs [KUM&al17] have becomes more popular in the last decade. While Rosenblatt's perceptron has one single output and a single set of weights and biases, modern neural networks benefit from deeper structures. Consequently, the network is equipped with several layers, each one with appropriate dimensionality of weights and biases, acting on the output of Eq. (3.1) recursively. The layers before the last one are then referred to as hidden layers, whereas the final one is the output layer. More sophisticated neural network layers have been developed throughout the years to tackle specific tasks. Convolutional layers [LB98], for instance, have been extensively used for image and video recognition tasks. Recurrent neural networks resemble the structure of dynamical systems [HS97; GBC16] and are generally used for sequential inputs, such as texts or speech. Graph-neural networks [ZCH&al21] resemble the computational structure of relational graphs and have recently found use in modelling complex physical systems [SHS&al18], such as proteins [FBS&al17].

3.3 Automatic differentiation

Automatic differentiation is a powerful tool to compute derivatives of functions by using the chain rule. This approach is fundamentally different than numerical differentiation, which is prone to instabilities, and symbolic differentiation, whose derivative expression can quickly become unmanageable due the large redundancies they produce [BPR&al18]. Any mathematical operation executed on a computer uses a specific set of basis functions, for example the sinus function or the square root function [Ver00]. More complicated operations are executed as a recursive application of the fundamental functions. This makes it possible to differentiate any such computation with respect to an input value by knowing the derivatives of the elementary functions and then applying the chain rule on the recursive computations. The emergence of automatic differentiation has greatly simplified the application of machine learning models to scientific problems. For instance, all four Chapters 4 - 7 rely on automatic differentiation in JAX [BFH&al18] to optimize different types of cost functions, some of which are particularly complex from a numerical perspective and would be intractable without AD. Backward-mode AD can arguably be considered for discrete systems what the Portraying maximum principle is for continuous systems [BPR&al18] and has found application, for instance, in the study of neural ordinary differential equations [CRB&al18]. Forward-mode AD has also been considered [BPS&al22], but its efficiency in high-dimensional systems has been questioned [Bel22].

3.4 Supervised Learning

Supervised learning (usually referred to as *regression* or *classification* in the context of statistics) is a sub-field of machine learning that deals with learning from a data set $\mathcal{D}$. The data sets consists of labels $\boldsymbol{y}_i \in \mathbb{R}^k$ and corresponding data points $\boldsymbol{x}_i \in \mathbb{R}^d$, s.t. $\mathcal{D} = \{(\boldsymbol{x}_i, \boldsymbol{y}_i) \mid i = 1, ..., L\}$. The goal is to find a suitable function $\tilde{f} : \mathbb{R}^d \mapsto \mathbb{R}^k, \boldsymbol{x} \longmapsto \tilde{f}(\boldsymbol{x})$ that can represent the relation between the data. The optimal function is usually selected by minimizing an appropriate functional over the data. This functional quantifies how well a family of functions can map input data to output data.
Arguably, the simplest supervised learning algorithm is the linear regression. In this framework, we are given an unknown stochastic environment [Hay09] with input vector $\boldsymbol{x} \in \mathbb{R}^d$. The action of the environment perturbs the vector to produce a response $\boldsymbol{y} \in \mathbb{R}^k$. Sampling from the black-box environment multiple times gives us samples $\boldsymbol{x}_1, ..., \boldsymbol{x}_L$,

that is

$$\boldsymbol{y}_i = f(\boldsymbol{x}_i) + \boldsymbol{\epsilon}_i, \tag{3.3}$$

where $f : \mathbb{R}^d \mapsto \mathbb{R}^k, \boldsymbol{x} \longmapsto f(\boldsymbol{x})$ is an unknown function and $\boldsymbol{\epsilon}_i, i = 1, ..., L$ is an error term with $\mathbb{E}[\boldsymbol{\epsilon}_i] = 0$ [HTF09]. The goal is to approximate the input-output map of the stochastic environment by optimizing appropriate parameters. We limit ourselves to *linear* models, that is, functions of the type

$$\tilde{\boldsymbol{y}} = \overline{\boldsymbol{W}}\boldsymbol{x} + \boldsymbol{b}, \tag{3.4}$$

where $\overline{\boldsymbol{W}} \in \mathbb{R}^{k \times d}$ is a matrix and $\boldsymbol{b} \in \mathbb{R}^k$ is a bias vector. Bayesian statistics [Was10] allows to characterize the uncertainty in the parameters [Hay09]. In fact, the posterior probability distribution of the output given the input and the parameters can be written as:

$$p(\boldsymbol{y}|\boldsymbol{\theta}, \boldsymbol{x}) = \frac{p(\boldsymbol{\theta}|\boldsymbol{y}, \boldsymbol{x})p(\boldsymbol{\theta})}{p(\boldsymbol{y})}. \tag{3.5}$$

The Bayesian framework provides us with two approaches to obtain the optimal parameter $\boldsymbol{\theta}^*$. On the one hand, we can derive the maximum likelihood estimator by minimizing the so-called likelihood function l [Hay09; Was10]:

$$\boldsymbol{\theta}^*_{\text{ML}} = \underset{\boldsymbol{\theta}}{\text{argmax}} \left[l(\boldsymbol{\theta}|\boldsymbol{y}, \boldsymbol{x}) \right], \tag{3.6}$$

and the maximum-a-posteriori optimization

$$\boldsymbol{\theta}^*_{\text{MAP}} = \underset{\boldsymbol{\theta}}{\text{argmax}} \left[p(\boldsymbol{\theta}|\boldsymbol{y}, \boldsymbol{x}) \right]. \tag{3.7}$$

It can be shown that for Gaussian stationary environments [Hay09], the maximum a-posteriori estimator resulting from Eq. (3.7) leads us to the minimization of the (regularized) mean-squared loss function. The mean-squared-loss function is defined as:

$$\mathcal{L}(\boldsymbol{\theta}) = \frac{1}{L} \sum_{i=1}^{L} L(\boldsymbol{\theta}, \boldsymbol{x}_i, \boldsymbol{y}_i) \tag{3.8}$$

with

$$L(\boldsymbol{\theta}, \boldsymbol{x}, \boldsymbol{y}) = \frac{1}{2}(\overline{\boldsymbol{W}}\boldsymbol{x} + \boldsymbol{b} - \boldsymbol{y})^2 \tag{3.9}$$

for $\boldsymbol{\theta} = (\overline{\boldsymbol{W}}, \boldsymbol{b})$. The generalization of the least-square-regression to dynamic systems leads us to the Least-Mean-Square (LMS) filter [Kay93], which is the paradigmatic adaptive filtering algorithm for estimation and control. The algorithm can be implemented recursively to adapt to changing environments and gives rise to the recursive Least-

Mean-Square filter and the Kalman filter [Kay93].
So far, we have only considered models that are linear in the input data. In principle, arbitrary parametrized functions can be implemented. In this case, due to the non-linearity of the resulting cost function, the optimized parameters cannot be expressed analytically, so it is custom to implement different types optimization algorithms such as gradient descent, in which the model parameters are updated recursively in the direction of the gradient:

$$\boldsymbol{\theta}_{i+1} = \boldsymbol{\theta}_i + \eta \nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}_i), \tag{3.10}$$

where η is the so-called learning rate and $\boldsymbol{\theta}_0$ is the initial guess for the parameters. If successful, the optimization should converge to the parameters that minimize the loss function, that is $\boldsymbol{\theta}^* = \text{argmin}_{\boldsymbol{\theta}} [\mathcal{L}(\boldsymbol{\theta})]$. Often, however, reaching the so-called global minimium of the loss function proves particularly challenging and only local minima can be reached [GG24]. More refined algorithms have been developed for non-linear least square, such as the Levenberg-Marquardt algorithm [Lev44], BFGS [BRO70], L-BFGS-B [LN89], etc. However, most of these algorithms are too expensive from a computational point of view to be implemented for the optimization of neural networks with potentially large numbers of parameters and enormous data sets. For such problems, stochastic gradients [Rud17] and in particular mini-batch gradient descent algorithms have been used to optimize cost functions sampled over large data sets. These gradients are computed over a subset of the whole data set:

$$\nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}) \approx \frac{1}{N_B} \sum_{i=1}^{N_b} \nabla_{\boldsymbol{\theta}} L(\boldsymbol{\theta}, \tilde{\boldsymbol{x}}_i, \tilde{\boldsymbol{y}}_i), \tag{3.11}$$

where N_b is the batch size and the inputs $\tilde{\boldsymbol{x}}_i, \tilde{\boldsymbol{y}}_i$ are N_b pairs randomly sampled from the data set. Due to the small size of the batch compared to the whole data set, these gradients are much faster to evaluate. Nonetheless, they have been showed to perform extremely well in a variety of tasks compared to standard gradients, in particular in combination of momentum-based algorithms such as Adam [KB15]. These algorithms do not only make use of the first moment of the gradient estimate with respect to the batch, but also of its second moments. In the case of Adam, in particular, the approach is equivalent to sign descent [BH18].

3.5 Unsupervised Learning

Unsupervised learning (usually referred to as *clustering* in the context of statistics) describes a class of algorithms that attempts to find patterns in a data sets without the help of so-called labels. Instead, transformations are performed on the data to infer its mathematical properties. The main idea is to map an input $\boldsymbol{x}$, which is often high-dimensional, to a lower dimensional representation $\boldsymbol{z} = f(\boldsymbol{x}, \boldsymbol{\theta})$ [FHI&al18; Nau&al22] and then map it back to the original vector using a second function $g(\boldsymbol{z}, \boldsymbol{\theta}')$. In the clustering approach, we want to map high-dimensional data to a finite number of clusters. However, in contrast with supervised learning, we do not have labels $\boldsymbol{y}_i, i = 1, ..., L$ available for each input $\boldsymbol{x}_i, i = 1, ..., L$ from the data set. Before the rise of deep neural networks, several relevant algorithms were developed, such as (naive) k-means clustering algorithms [Llo82], mixture models trained with the expectation maximization algorithm [Bis07] and the principal component analysis (PCA) [Bis07], all of which come in different variants. Modern approaches make use of multilayered perceptrons tailored for unsupervised learning such as auto-encoders [GBC16; KW22] and normalizing flows [KPB21; CRB&al18]. The latter have proved particularly successful in tasks such as image, text and sound generation and are employed by several AI companies [RBL&al22].

3.6 Reinforcement Learning

The term reinforcement learning (RL) characterizes a family of learning algorithms where an agent, equipped with various computational structures usually in the form of neural networks, receives a reward signal by interacting with an environment in discrete steps. The goal of the agent it to maximize the long-term return of these rewards. The concept of return describes the overall amount of reward obtained by the agent during its interaction with the environment, weighted according to the importance of the reward in a specific moment of the training. During its interaction with the environment, the agent should not just aim for immediate rewards, but also develop long-term strategies. In recent years, RL has proved to be a successful tool in different branches of science and technology, with results spanning from beating humans at various board games [Sil&al16] and computer games [Vin&al19] to predicting the structure of proteins [Jum&al21] and designing drugs and molecules [Kor&al22]. We give here a brief overview of some relevant RL methods. For a more comprehensive overview of the most important results of the field, see Ref. [FHI&al18] and for a detailed treatment Ref. [SB18]. An introduction to RL is also given in Ref. [Pre20]. In the RL framework, the state space S describes all possibles states $s \in S$ that the agent can find itself in with respect to the environment and the action space A contains all possible actions $a \in A$ that the agent can execute. The interaction of the agent with the environment is divided in time-steps $t = 0, 1, ..., T$, where T is the maximum time. Usually, it is assumed that the process that governs the transition [FHI&al18] between any state-action pairs at time t, $(s_t,\, a_t)$ and the next state s_{t+1} is Markovian, as well as the assigned reward for each state-action-new state transitions, which means that the next state that the agent experiences in its interaction with the environment and the corresponding reward of the transition will be dependent only on the previous state and the chosen action, without memory of the previous choices of the agent. This allows to derive some important results regarding the convergence of the policy, such as the Bellman equation [Bel52]. The function that describes how the agent behaves in its interaction with the environment, i.e., which states are mapped to which action, is called policy. The policy can be deterministic, i.e., $\pi : S \mapsto A, s \longmapsto \pi(s)$, which means that the action of the agent at each step is completely determined by the state, or stochastic, i.e., $\pi : S \times A \mapsto [0, 1], (s, a) \longmapsto \pi(s, a)$, where the policy is a probability distribution over states and actions. The expected return of the reward associated with the policy π at time t [FHI&al18]:

$$V^{\pi}(s) = \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k r_{t+k} | s_t = s, \pi\right], \tag{3.12}$$

which is the so-called state-value function V^{π} associated with the policy π and where $\gamma \in [0,1)$ is a so-called discount factor that ensures the convergence of the sum and represents the diminishing weight of rewards that are located far in the future. The reward r_t is determined by the reward assignment function [FHI&al18]:

$$R : S \times A \times S \longmapsto \mathbb{R}, (s_t, a_t, s_{t+1}) \longmapsto R(s_t, a_t, s_{t+1}) \tag{3.13}$$

$$r_t = \mathbb{E}_{a \sim \pi(s_t, \cdot)}\left[R(s_t, a, s_{t+1})\right], \tag{3.14}$$

under the state transition function [FHI&al18]:

$$T : S \times A \times S \mapsto [0,1], (s_t, a_t, s_{t+1}) \longmapsto T(s_t, a_t, s_{t+1}). \tag{3.15}$$

Both the state-transition function and the reward-assignment function can be stochastic or deterministic. To determine the optimal policy π^*, an option is to define the so-called state-action value function [FHI&al18]:

$$Q^{\pi}(s,a) = \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k r_{t+k} | s_t = s, a_t = a\right], \tag{3.16}$$

from which the optimal policy can also be derived

$$\pi^*(s) = \underset{a \in A}{\operatorname{argmax}} \left[Q^{\pi}(s,a)\right]. \tag{3.17}$$

RL algorithms are classified based on how they use the information coming from the environment and how they optimize the policy towards the optimal policy [FHI&al18]. In *model-based* RL, for instance, a model for the environment is first built and then adapted using information collected from the environment. In *model-free* RL, instead, no such model is present, and the environment itself is treated as a (partial) black-box that feeds information to the agent. RL algorithms are also classified based on the nature of the training. So-called *value-based* methods rely primarily on the state-value or state-action-value functions (score functions that assign to states and actions a numerical value to determine their effectiveness) to optimize the policy. *Policy-based* methods, instead, optimize the policy directly without relying on score functions for the reward. Combinations of the different methods are possible and can significantly increase the proficiency of the algorithms. In the following sections we briefly outline some algorithms for RL and their limitations. We also discuss the importance that deep neural networks have acquired in the context of RL in the course of the last decade.

3.6.1 Tabular methods

Tabular methods are value-based RL algorithms that store the values associated to a discrete set of actions A and states S in a vector $V^{\pi}_{s} \in \mathbb{R}^{|S|}$ for the value-function and in a matrix $Q^{\pi}_{s,a} \in \mathbb{R}^{|S|\times|A|}$ for the state-action-value function. Some tabular RL algorithms use state-value functions, e.g., TD-0 and TD-λ [SB18], whereas others, such as Q-learning, use action-value functions in their optimization routines. In Q-learning, for instance, the optimal policy is determined by maximizing over the action dimension, while the table of values is updated using the rewards coming from the environment [WD92; SB18]:

$$Q^{t+1}(s,a) = Q^{t}(s,a) + \alpha\left(R_t + \gamma \max_{a_{t+1}} Q(s_{t+1},a_{t+1}) - Q(a_t,s_t)\right). \tag{3.18}$$

Another relevant tabular algorithm is SARSA (an acronym for state-action-reward-state-action) [RN94], which modifies the update rule for the Q-function to:

$$Q^{t+1}(s,a) = Q^{t}(s,a) + \alpha\left(R_t + \gamma Q(s_{t+1},a_{t+1}) - Q(a_t,s_t)\right). \tag{3.19}$$

Monte Carlo Tree Search (MCTS) can be used to enhance the exploration of the state and action spaces [SB18; Mni&al15] and has also found implementation in more recent algorithms [Sil&al17; Dal&al20]. More recently, the framework of Projective Simulation (PS) has been introduced [BD12]. This approach to intelligent agents inspired by physics also retains similarities with the aforementioned algorithms [Mau&al15], but it strengthens the connection between the underlying physical model and the learning algorithm. These algorithms represent the building blocks of modern RL, however their convergence is limited by the tabular structure, whose number of entries scales with $O(|A||S|)$. This scaling is unfeasible for large discrete action and state spaces. Board games such as Go [Sil&al16] are example of environments that are intractable with tabular methods. The issue is even more evident in continuous state and action spaces, e.g., real-world control problems that cannot be discretized easily. Moreover, the update rule given by the Bellman equation quickly becomes intractable as the number of states and actions grows [FHI&al18].

3.6.2 Deep reinforcement learning - value-based

Some early attempts to learn Q functions through regression make use of radial basis function networks [SB18]. One possible alternative approach is to represent Q as a guess $Q'(\boldsymbol{\theta})$ that depends on real parameters $\boldsymbol{\theta}$ [FHI&al18; Rie05]:

$$\mathcal{L}(\boldsymbol{\theta}) = (r + \gamma \max_{a\in A} Q(s,a,\boldsymbol{\theta}) - Q'(\boldsymbol{\theta}))^2. \tag{3.20}$$

The fitting parameters $\boldsymbol{\theta}$ are minimized using a gradient descent algorithm with respect to the loss $\mathcal{L}$. The deep Q-learning approach uses a target network that is updated only once every few iterations and therefore stabilizes training and replay memory that is used as a data set to sample the actions and states to train the Q function [Mni&al15; FHI&al18]. In deep RL, we use a feed-forward neural network or a convolutional neural network [GBC16] to represent either the policy, the value function or both. More specifically, in the latter case, the agent is able to learn from visual inputs of the environment state, such as, e.g., the display output of Atari games [MKS&al13; Mni&al15].

3.6.3 Deep reinforcement learning - policy-based

Policy gradients date back to the original REINFORCE algorithm introduced by Williams [Wil92]. The idea is to evaluate gradients of black-box functions using Monte Carlo gradient estimation, they build the basis of several black-box optimization and neuroevolution algorithms. Let us consider a function $f : \mathbb{R}^n \longrightarrow \mathbb{R}, \boldsymbol{\theta} \mapsto f(\boldsymbol{\theta})$ and a probability distribution $\pi(\boldsymbol{x};\boldsymbol{\theta})$ with parameter vector $\boldsymbol{\theta}$. Then we have [WSG&al14]:

$$\nabla_{\boldsymbol{\theta}} \mathbb{E}_{\boldsymbol{x}\sim\pi(\boldsymbol{x};\boldsymbol{\theta})} \left[f(\boldsymbol{x})\right] = \int f(\boldsymbol{x}) \nabla_{\boldsymbol{\theta}} \pi(\boldsymbol{x};\boldsymbol{\theta}) d^n x = \int f(\boldsymbol{x}) \nabla_{\boldsymbol{\theta}} \log\left[\pi(\boldsymbol{x};\boldsymbol{\theta})\right] \pi(\boldsymbol{x};\boldsymbol{\theta}) d^n x, \tag{3.21}$$

so the estimator for the gradient can be written as

$$\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}) \approx \frac{1}{N} \sum_{i=1}^{N} f(\boldsymbol{x}_i) \nabla_{\boldsymbol{\theta}} \log\left[\pi(\boldsymbol{x}_i;\boldsymbol{\theta})\right], \tag{3.22}$$

for samples $\boldsymbol{x}_1, ..., \boldsymbol{x}_N$ from the probability distribution $\pi(\boldsymbol{x};\boldsymbol{\theta})$. This type of gradient estimation is used in various areas of machine learning and optimization [MRF&al20]. It builds the basis of so-called NES (Natural Evolution Strategies), that are a class of black-box optimization algorithms [WSG&al14] that have found various applications in machine learning [SHC&al17; KT18] and of the policy gradient algorithm in RL. In the context of RL, one usually considers a parametrized stochastic policy $\pi(a, s;\boldsymbol{\theta})$ that determines the probability of choosing an action a given a state s. The score function that needs to be maximized is given by the reward $R(s, a)$, which guides the agent towards the optimal policy. Gradients of RL policies can be formulated for both stochastic [SMS&al99; FHI&al18] and deterministic policies [SLH&al14]. This formulation is used in the DDPG algorithm [LHP&al19]. The policy gradient-based optimization can also be enhanced by using natural policy gradients [WSG&al14; FHI&al18]. This approach is used, with some modifications, in state-of-the-art algorithms such as TRPO [SLM&al17] and PPO [SWD&al17]. The methods can be combined with actor-critic schemes: the agent policy is parametrized by a neural network with either stochastic or deterministic output, whereas the state-value or action-value functions are parametrized by a different neural network [SB18]. The updates of the policy are based on the values of the Q-functions. Here the difference is determined by each specific algorithm, but the combination of policy-gradient based updates and value function updates proves particularly efficient.

3.7 Machine learning for quantum science

In this Section we give a brief overview of some applications of machine learning to quantum science and technology, focusing in particular on supervised and reinforcement learning – see also Ref. [KLF&al23] for a detailed analysis. When we refer to machine learning (and more generally artificial intelligence) for quantum science, we usually consider a variety of applications of neural networks and learning algorithms to the study of quantum systems, the design and optimization of experiments both in theory and practice, i.e., where the agent actively interacts with a physical lab. This domain is therefore different than quantum machine learning [Bia&al17], which studies instead how to design quantum algorithms for artificial intelligence that can run on fault-tolerant and potentially also NISQ devices. Among the most notable achievements of using machine learning in quantum science are the approximation of quantum states and their dynamics [CT17; GM22]. In quantum matter, unsupervised learning has been successfully employed to detect phase transitions [KHL&al20]. RL has proved to be particularly successful in delivering solutions to problems in quantum science and technology. These applications include design of measurement strategies for quantum systems, state preparation and gate synthesis tasks [CRS&al21; PSJ&al24] with variational circuits and quantum control techniques [Dal&al20; BAH&al21; GST24], transport of quantum states [Por&al19], optimization of quantum error correction schemes [NDD&al19] and feedback strategies [PPM23]. Particularly interesting are applications involving generalization, that is when the machine learning model is able to learn an optimal strategy and then apply it in different contexts: as an example, in Ref. [MPK&al18], the agent is trained to generate a (photonic) multipartite entangled state with a specific Schmidt rank vector [EB01]. It is shown that the same agent can discover states with different Schmidt rank vectors faster compared to an agent which is trained only to generate these other states. Therefore, identifying underlying patterns in the behaviour of intelligent agents has become an interesting research topic [Nau&al22; Mel&al17; RMB19]. This work adds another contribution to the application machine learning concepts in quantum technology: The algorithms provided in Chapter 4 use a supervised learning approach with different types of cost functions to discover adaptive pulses for quantum optimal control, whereas in Chapter 5 RL is employed together with continuous optimization to discover optimal quantum circuit compilation strategies. Chapter 6 and 7 do not make direct use of neural network approaches. However, the optimization problems considered therein could potentially benefit from machine learning approaches, as given, e.g., in Ref. [WMD&al20]. Additionally, Chapter 8 mentions quantum machine learning as one of the possible applications of the sampling method discussed therein.

4 Continuous quantum gate sets and generalized pulse meta-optimization

Disclaimer: A modified version of this chapter was published in Ref. [PCM22]. FP and FM developed the theoretical framework of the paper. FP developed the code and ran all the simulations. The paper was written by FP with inputs from FM. TC provided useful guidance and revisions to the manuscript.

Reduction of the circuit depth of quantum circuits is a crucial bottleneck to enabling quantum technology. This depth is inversely proportional to the number of available quantum gates that have been synthesized. Moreover, quantum gate-synthesis and control problems exhibit a vast range of external parameter dependencies, both physical and application specific. In this paper, we address the possibility of learning families of optimal-control pulses that depend adaptively on various parameters, in order to obtain a global optimal mapping from the space of potential parameter values to the control space and hence to produce continuous classes of gates. Our proposed method is tested on different experimentally relevant quantum gates and proves capable of producing high-fidelity pulses even in the presence of multiple variables or uncertain parameters with wide ranges.

4.1 Introduction

The standard view of quantum computation [NC10] uses the classical-computing abstraction of a subdivision into a finite set of gates, measurements, and state reset tasks. This paradigm has a number of benefits: notably, it permits formal derivation of universal computation [DN06; BBC&al95], that is, the composition of a quantum circuit for any desired unitary operation, as well as error-correcting codes [Per85; Got98], where specific error syndromes can be measured and corrected. On the other hand, this abstraction abandons the essential analog character of quantum devices, from which they have the most to gain or lose in terms of their expressibility or fragility, respectively. That is, the power of quantum processors depends strongly on the number of usable gates available to them.
Practically speaking, the usage of discrete gate sets falls short in at least four important respects. First, the analog character is a more complete (and therefore efficient) description of variability between different qubits, which is inevitable, for instance, in solid-state qubits [Aru&al19]. The use of qubit-agnostic gate sets as the computational

primitive means that each qubit must be individually optimized and calibrated to yield each such gate [Aru&al19], where parametric description of the gates would naturally capture the variations. Second, these variations can occur for a given qubit as a function of time [Pro&al20], e.g., due to time-dependent noise, and require complete recalibrations where, typically, drift involves only a single parameter. This is very taxing on the routines for error suppression, mitigation, and correction.
Third, the complexity arising from parameter variations in space and time is exacerbated by the subsequent circuit complexity in composing useful circuits. It is well known that, while discrete gate sets can be universal, the required number of discrete constituent gates can be very large even for a simple circuit [BBC&al95; Fox&al20; YLB21]. Allowing for analog parameter tuning can dramatically increase the controllability of the system and thereby necessarily reduce the depth of the quantum circuit for arbitrary tasks (see, e.g., Appendix 4.7.3). Since errors accrue with circuit depth (notably due to decoherence), this increase in the circuit success probability may be highly beneficial to both short-term and long-term (i.e., fault-tolerant) approaches.
Fourth, a further optimization layer is currently ubiquitous in noisy intermediate-state quantum (NISQ) algorithms [Pre18]. Such optimizable circuits include adiabatic [AL18; BSL&al16], annealing [ACd89; KN98], and variational [Per&al14; FGG14] quantum algorithms. The common denominator in these approaches is the circuit being treated as a black box, with a set of analog parameters acting as knobs to tune as inputs for the respective algorithm. These analog inputs act as terms in the Hamiltonian and thus may generate various quantum continuous gate sets. These cannot realistically be compiled with digital gates, and, moreover, to ensure that the gate set is correctly specified requires a formal approach for their general construction.
Our contribution, in this work, is to present a unified framework to efficiently describe and optimize continuous quantum gate sets in these scenarios. This framework allows learning of the parametric gates that can be tuned to very high fidelity across a large number and wide range of parameter values. We refer to our method as single-optimization multiple-application (SOMA) quantum gate synthesis. This kind of learning can be understood as an instance of meta-optimization [HAM&al20; DVB19] or adaptive-trajectory learning [AKM84; TSH18; DVS01; JT09; CW19; OSF&al91; BAA22] and can be related to more recent uses of neural networks in quantum physics [BDS&al18; YYW18; Niu&al19; WDD&al19; LKB&al20; YLB21]. That is, we find a solution for an optimizer that itself provides solutions to specific problem instances or, with a different formulation, automatically discovers heuristics to construct solutions for a specific optimization problem. In particular, we see that our algorithms are able to synthesize heuristics for general Hamiltonians.
We break down the problem of generalized gate synthesis into the following components. We parametrize our quantum gate set using continuous indices that represent either physical system parameters or desired angles of a continuous Lie group. We then present two different machine-learning methods for obtaining continuous control parametrizations that generate the indexed gate set. The first is a supervised approach where traditional optimal-control theory is used to generate an operational data set from which a generalized gate-set recipe can be trained. The second is an unsupervised

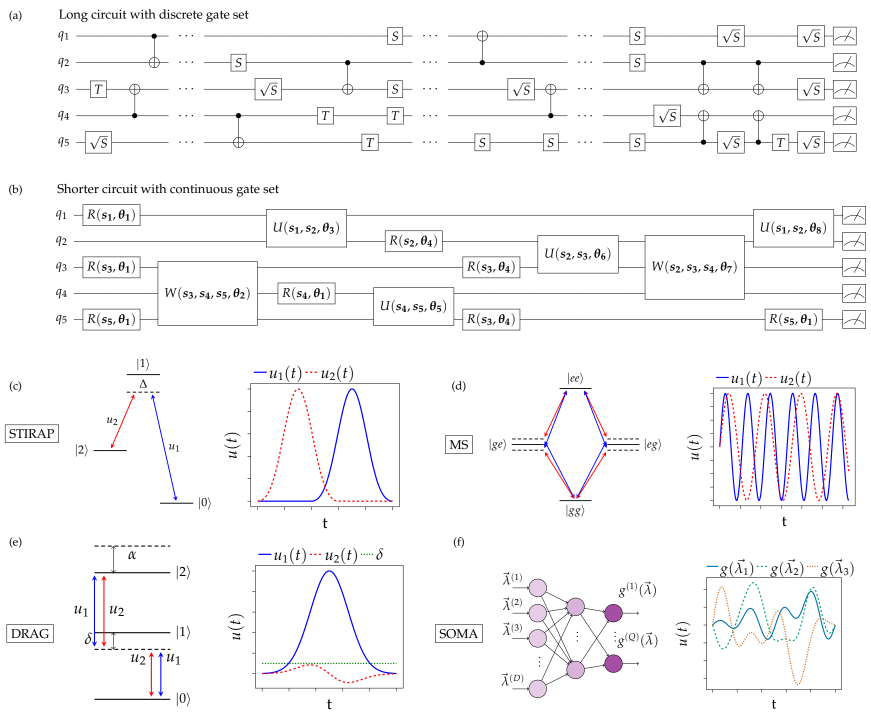

(a)
Long circuit with discrete gate set
(b)
Shorter circuit with continuous gate set
(c)
STIRAP
(d)
MS
(e)
DRAG
(f)
SOMA

Figure 4.1: An explanatory diagram of the concepts discussed in Section 4.2. (a) A general quantum circuit containing a long sequence of discrete unitaries (Clifford + T set), which do not exhibit any dependence on continuous parameters. (b) A general quantum circuit containing different unitaries with different continuous parameters $s_1, s_2, ..., s_5$ for different qubits $q_1, q_2, ..., q_5$ and angles $\theta_1, \theta_2, ..., \theta_8$ parametrizing each gate. R, U, and W represent an analog single-, two- and three-qubit gate, respectively. Here, the same unitary parametrization is capable of representing all the different gates needed in the circuit. We show some notable analytical solutions used to engineer specific gate operations, which are usually implemented due to their adaptive character and simplicity (see Eqs. 4.5,4.6, and 4.7). (c) STIRAP [KGH&al89; VRS&al17]. (d) The Mølmer-Sørensen gate [SM99; SM00]. (e) DRAG [MGR&al09; TMM&al18]. (f) The method that we propose, SOMA, does not make strict assumptions on how the pulse depends on the problem parameters, but, rather, discovers it more generally through training.

approach using back propagation from which the continuous gate set can be incrementally learned over the entire training population. Finally, we show that our approaches encompass the various situations discussed above, including general solutions for gates given generic physical architectures with wide parameter ranges, noise-adaptive optimal-control theory, and compilation of a Lie group instead of a single element.

The paper is organized as follows. In Sec. 4.2, we introduce the notation for supervised and unsupervised training of parametrized pulses. In Sec. 4.3, we discuss the results obtained by applying these methods to single-qubit and two-qubit gates in the presence of leakage, showing that they display similarities with known analytical-solution families. Furthermore, we also investigate how our methods perform compared to other existing algorithms. Finally, in Sec. 4.4 we analyze the dependence on the parameter variability range, the training data set size and batch size, the system and network size and again compare our algorithms to other numerical robust approaches. We summarize our conclusions in Sec. 4.5.

4.2 Continuous gate set learning

4.2.1 Definitions

We define a continuous gate set in terms of some continuous sets or distributions of n system parameters $s_1, s_2, \ldots, s_n$ and m gate specifications $\theta_1, \theta_2, \ldots, \theta_m$. An element of such a gate set is a unitary transformation between two Hilbert spaces H_1 and H_2:

$$U(s_1, s_2, \ldots, s_n, \theta_1, \theta_2, \ldots, \theta_m) : H_1 \mapsto H_2. \tag{4.1}$$

Obtaining a single element of this set is a well-known problem in quantum information. Depending on whether the unitary is synthesized from discrete or analog dynamics, its composition is referred to as circuit compilation [BBC&al95; DN06; BRS15; MKW17] or optimal-control theory [WRD93; Fre98; KRK&al05; GBG&al19; CCM11; GBC&al15], respectively.
In Fig. 4.1(a), we see an example of a generic circuit acting on different qubits. Each qubit in space (and time) will have different values of the common system parameters $\{s_k\}$. In addition, the different unitaries in the gate set $\{U_i\}$, each additionally characterized by rotation angles $\{\theta_{i,j}\}$, may vary both throughout the circuit and in iterated uses of the circuit (e.g., in NISQ algorithms). For compactness we now regroup the continuous indices into a common array of indices $\vec{\lambda} = (s_1, s_2, \ldots, s_n, \theta_1, \theta_2, \ldots, \theta_m)$.
Such a generalization of the gate-synthesis problem can be framed as solving for the inverse function of the general dynamics given in Eq. (4.1), that is, for

$$g : \vec{\lambda} \mapsto \boldsymbol{u}(t), \tag{4.2}$$

where $\boldsymbol{u}(t)$ is a wave-form function in the standard case of optimal-control theory, but can also be thought of as a discrete sequence of hard pulses, as in NMR applications, or of unitary gates in circuit compilation. Importantly, the optimal $\boldsymbol{u}(t)$ changes for each parametrization $\vec{\lambda}$ of the unitary, which can be generated from e.g. the Schrödinger equation.

$$\dot{U}(\vec{\lambda}, t) = -iH(\vec{\lambda}, \boldsymbol{u}(t))U(\vec{\lambda}, \boldsymbol{u}(t)), \tag{4.3}$$

or other equations of motion defining the system. This task can be cast as an instance of meta-optimization [Gre86] or trajectory learning for control [SA10; AM86; OSF&al91]. We formalize the meta-optimization problem as list of problem-parameter vectors $\vec{\lambda}_1, ..., \vec{\lambda}_L \in V_\lambda \subset \mathbb{R}^D$ with figures of merit $F_1(\boldsymbol{x}) = F(\boldsymbol{x}, \vec{\lambda}_1), ..., F_L(\boldsymbol{x}) = F(\boldsymbol{x}, \vec{\lambda}_L)$ and initial guess $\boldsymbol{x}_0$, the physical parameters of which vary somewhat from each other, such that $\vec{\lambda}_i \sim \pi(\vec{\lambda}; \boldsymbol{v})$ for $0 \leq i \leq L$ and $\boldsymbol{v} \in V_\lambda$, is drawn from a parameter distribution. Throughout the paper we assume, without loss of generality, π to be a multidimensional uniform distribution, such that $\pi(\vec{\lambda}|\boldsymbol{v}) = U(\vec{\lambda}_{\min}, \vec{\lambda}_{\max})$, with $\boldsymbol{v} = (\vec{\lambda}_{\min}, \vec{\lambda}_{\max})$ defining the parameter space V_λ. The objective is to find optimal parameters $\boldsymbol{w}^*$,

$$\boldsymbol{w}^* = \underset{\boldsymbol{w}}{\operatorname{argmin}} \; \{1 - F_i(\boldsymbol{w})| \; \forall i = 1, ..., L\} \tag{4.4}$$

which allow for simultaneous optimization of all the systems considered within the range of sampled parameters.

4.2.2 Analytical adaptive control

The most straightforward way to generate classes of solutions to quantum gates has been the development of analytical solutions for particular quantum systems.
They have in common the knowledge of the relevant state of the system at all times during the evolution. In particular, analytical knowledge of the eigenvalues [LCM22] allows reverse engineering of the pulses to provide (near) exact solutions for the desired states.
Fig. 4.1(c)-(f) shows several celebrated examples where such general classes of solutions have been found. We quickly review some of their main features. Given a trial pulse shape such as a Gaussian envelope $p(t, t_1, t_2, \theta) = \mathcal{A}e^{(t - t_2/2 + t_1/2)^2/\sigma^2}$, where θ denotes the area under the curve, the following dynamical solutions have been found.
Fig. 4.1(c) shows the stimulated Raman adiabatic passage (STIRAP) solution for transfering population between disconnected states $|0\rangle$ and $|2\rangle$, while avoiding any (nonvanishing) temporary population in the intermediary connecting state $|1\rangle$. The pulse shaping is given by

$$\begin{gathered} \text{STIRAP}(|0\rangle \to |2\rangle) : (\Omega_1, \Omega_2, \Delta, \theta, T) \mapsto \boldsymbol{u}(t), \\ u_1(t) = \Omega_1 e^{i\theta} p(t, T/3, T, \pi) e^{i\Delta t} |0\rangle\langle 1| \\ u_2(t) = \Omega_2 p(t, 0, 2T/3, \pi) e^{-i\Delta t} |1\rangle \langle 2| \end{gathered} \tag{4.5}$$

where the order of the pulses u_1 and u_2 is famously counterintuitive [KGH&al89; VRS&al17].
Fig. 4.1(d) shows the Mølmer-Sørensen (MS) method for trapped-ion gates [SM99;

SM00]. The required laser pulses are given by

$$\begin{aligned}\mathrm{MS}(\sigma_x \otimes \sigma_x \otimes \cdots \otimes \sigma_x) &: (\Omega_1, \Omega_2, \delta, \theta_1, \theta_2) \mapsto \boldsymbol{u}(t), \\ u_1(t) &= \Omega_1 p(t, 0, T, \theta_1) e^{i\theta_2} e^{i\delta t} \left|g, n\right\rangle\!\left\langle e, n-1\right| \\ u_2(t) &= \Omega_2 p(t, 0, T, \theta_1) e^{-i\delta t} \left|g, n\right\rangle\!\left\langle e, n+1\right|,\end{aligned} \tag{4.6}$$

where the first state index denotes the state of the relevant qubit and second index denotes the phonon occupation number. Because of the symmetry of the gate, it can in theory be used on an arbitrarily large number of qubits, i.e., it is a collective gate [MMN&al16; MKW17].
Fig. 4.1(e) shows the Derivative Removal for Adiabatic Gate (DRAG) solution, for leakage suppression in multi-level systems [MGR&al09; TMM&al18]. This pulse profile is given by

$$\begin{aligned}\mathrm{DRAG}(|0\rangle \leftrightarrow |1\rangle) &: (\Omega_1, \Omega_2, \alpha, \delta, \theta_1, \theta_2) \mapsto \boldsymbol{u}(t), \\ u_1(t) &= e^{i\theta_2} e^{i\delta t} p(t, 0, T, \theta_1)(\Omega_1 \left|0\right\rangle\!\left\langle 1\right| + \Omega_2 \left|1\right\rangle\!\left\langle 2\right|) \\ u_2(t) &= i e^{i\theta_2} e^{i\delta t} \partial_t p(t, 0, T, \theta_1)/\alpha(\Omega_1 \left|0\right\rangle\!\left\langle 1\right| + \Omega_2 \left|1\right\rangle\!\left\langle 2\right|).\end{aligned} \tag{4.7}$$

We see, with these families of solutions, the common trend that they allow for different known system parameters or for different rotation or phase angles. Naturally, this is just a representative set, but where the equations of motion are integrable such solutions are numerous in the literature.
However, there are a few evident concerns about finding such analytical solutions. Firstly, it is a labour-intensive task with no guaranteed result, where even particular solutions do not preclude that a more systematic search would produce better results. Secondly, it is important to emphasize that these are solutions to idealized models, and in practice the more accurate physical models are not exactly solved by these ansätze. Finally, most physical systems have to date not been able to find general analytical solutions beyond qubits, qutrits and highly symmetric systems, both because of the larger state space and the larger parameter space. This limits their viability for quantum computing, which requires much larger Hilbert spaces.

4.2.3 GRAPE

GRAPE (GRadient Ascent Pulse Engineering) [KRK&al05] is a method originally developed in the context of quantum chemistry for the optimization of dynamical evolution of NMR systems. The algorithm assumes a unitary dynamics $U(t) = e^{-\int_{t_0}^{t} iH(\tau)d\tau}$, governed by a Hamiltonian of type:

$$H(t) = H_0 + \sum_{m=1}^{M} u_m(t) H_m \tag{4.8}$$

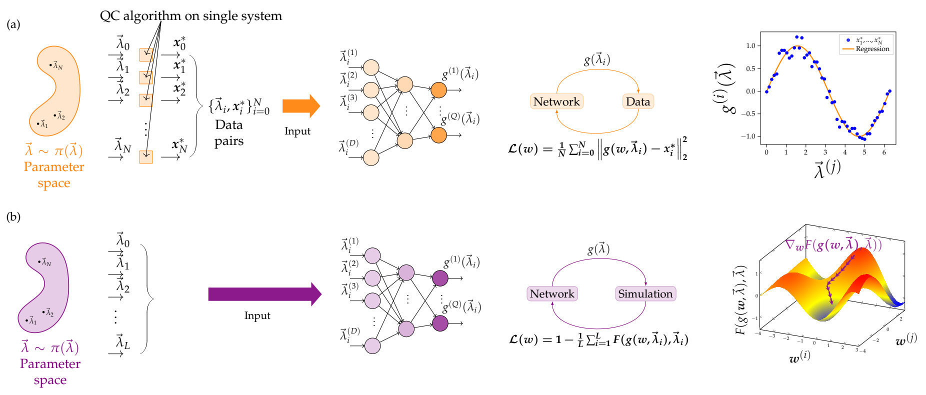

(a)
QC algorithm on single system
$\vec{\lambda} \sim \pi(\vec{\lambda})$
Parameter space
$\{\vec{\lambda}_i, x_i^*\}_{i=0}^N$
Data pairs
Input
$g(\vec{\lambda}_i)$
Network
Data
$\mathcal{L}(w) = \frac{1}{N}\sum_{i=0}^N \left\| g(w,\vec{\lambda}_i) - x_i^* \right\|_2^2$
Regression
(b)
Simulation
$\mathcal{L}(w) = 1 - \frac{1}{L}\sum_{i=1}^L F(g(w,\vec{\lambda}_i),\vec{\lambda}_i)$
$\nabla_w F(g(w,\vec{\lambda}),\vec{\lambda}))$

Figure 4.2: The Single-Optimization Multiple-Application gate synthesis method represented in its two variants. (a) In SOMA SL we first optimize N different QOC problems with different problem parameters $\vec{\lambda}_1, \vec{\lambda}_2, ..., \vec{\lambda}_N$ using an optimal quantum control algorithm such as [KRK&al05; GBG&al19; Fra&al17; DMJ&al20] to obtain optima $\boldsymbol{x}_1^*, \boldsymbol{x}_2^*, ..., \boldsymbol{x}_N^*$, then use a function approximator to learn the mapping $g : \mathbb{R}^D \mapsto \mathbb{R}^Q, \vec{\lambda} \longrightarrow \boldsymbol{x}^*(\vec{\lambda})$ between the problem parameters and the optimal pulses. (b) In SOMA BP, we sample L OQC problems $\vec{\lambda}_1, \vec{\lambda}_2, ..., \vec{\lambda}_L$ and train the function approximator to minimize the average infidelity of the ensemble of problems using back propagation, without generating optimal solutions for a single problem with a standard quantum control method.

where H_0 is a time-independent drift Hamiltonian and H_m, with $m = 1, ..., M$, are different suitable control Hamiltonians with corresponding control fields $u_m(t)$. In simulations, $U(t)$ is often computed through different types of Trotterization [Oli08].
GRAPE provides us with an efficient gradient of the merit function with respect to the control pulse values. The merit function is usually given by the gate fidelity:

$$F = \frac{1}{d^2} |\operatorname{Tr}\left\{UG^\dagger\right\}|^2, \tag{4.9}$$

where the normalization factor d corresponds to the dimension of the Hilbert space and G is a target unitary, which we would like to generate using the unitary dynamics. We assume a Trotter-like unitary evolution of the system of type:

$$U(T) = \prod_{j=1}^{N_{\text{evo}}} U_j(t_j, t_{j-1}), \tag{4.10}$$

where $U_j(t_j, t_{j-1}) = e^{-iHdt}$, with $dt = T/N_{\text{evo}} = t_j - t_{j-1}\ \forall j = 1, ..., N_{\text{evo}}$, where N_{evo} defines the number of time steps used in the Trotterization. In particular, considering

Eq. (4.8), the unitary step U_j reads

$$U_j = \exp\left\{-idt\left(H_0 + \sum_k u_k(j)H_k\right)\right\}. \tag{4.11}$$

The gradient of the fidelity can be computed iteratively starting from the so-called propagated optimal state [KRK&al05]:

$$O_j = U_{j+1}...U_{N_{evo}}GU_1...U_{j-1}, \tag{4.12}$$

so that the gradient approximately results in:

$$\frac{\partial F}{\partial u_k(j)} \approx \frac{2}{d^2}\mathrm{Re}\left\{\mathrm{Tr}\left\{UG^\dagger\right\}\mathrm{Tr}\left\{idtH_kO_j\right\}\right\}. \tag{4.13}$$

This approach, however, cannot directly account for variations of the underlying dynamics due to e.g., stochastic noise [BHC10], field inhomogeneity [MSZ&al06], or Hamiltonian uncertainties [KMM&al12] and the optimal pulses output by following the native gradient direction can prove significantly worse than expected if some of the underlying problem parameters vary. A possible way around this is to switch to a robust control approach, in which the cost function (4.9) accounts for parameter shifts. A simple way [KRK&al05] is to use an average fidelity over the parameter space sampled with quasi-Monte Carlo,

$$\bar{F} = \frac{1}{L}\sum_{l=1}^{L} F(\boldsymbol{w}, \vec{\lambda}_l), \tag{4.14}$$

where L is the number of samples.

4.2.4 Robust control

Similar to the adaptive solutions using analytical methods, solutions for controls robust to uncertainty in parameters have been found both by analytical and numerical means [Fre98; MSZ&al06; BHC10; KMM&al12; Bie&al09; MHW&al16; Sch&al22; MGC18; YYW18]. Since this involves only a single solution and not a family thereof, this has been the preferred method for parameter variability in quantum gate sets, as they are easier to design.
Robust solutions are generally defined slightly differently from the adaptive solutions, using the figure of merit

$$\boldsymbol{w}^* = \underset{\boldsymbol{w}}{\mathrm{argmin}}\left(1 - \frac{1}{L}\sum_{i=1}^{L} F_i(\boldsymbol{w})\right). \tag{4.15}$$

This cost function is more tractable from an optimization point of view because we can simply account for an ensemble of individual cost functions by taking the average of the cost functions as the objective of the optimization.
Robust solutions have also been found using analytic methods. In particular, a common pulse sequence for gates with robustness to amplitude noise is the Broad-Band 1 (BB1) sequence

$$\begin{aligned} \mathrm{BB1}(|0\rangle \leftrightarrow |1\rangle) &: (\Omega, \Delta\Omega, \theta_1, \theta_2) \mapsto \boldsymbol{u}(t), \\ u_1(t) &= \Omega p(t, 0, T/4, \theta_1) e^{i\phi} |0\rangle\langle 1| \\ u_2(t) &= \Omega p(t, T/4, T/2, \pi) e^{i\theta_2 + i\cos^{-1}(-\theta_1/4\pi)} |0\rangle\langle 1| \\ u_3(t) &= \Omega p(t, T/2, 3T/4, 2\pi) e^{i\theta_2 + i3\cos^{-1}(-\theta_1/4\pi)} |0\rangle\langle 1| \\ u_4(t) &= \Omega p(t, 3T/4, T, \pi) e^{i\theta_2 + i\cos^{-1}(-\theta_1/4\pi)} |0\rangle\langle 1| \end{aligned} \tag{4.16}$$

which is independent of offsets in Rabi frequency $\Delta\Omega$ up to sixth order, $1 - F = O(\Delta\Omega^6)$ [Wim94].
Likewise, when applying gates with unknown frequency offsets, the Compensation for Off-Resonance with a Pulse SEquence (CORPSE) method [CJ00] given by

$$\begin{aligned} \mathrm{CORPSE}(|0\rangle \leftrightarrow |1\rangle) &: (\Omega, \Delta, \theta_1, \theta_2) \mapsto \boldsymbol{u}(t) \\ u_1(t) &= \Omega e^{i\theta_2} p\left(t, 0, \frac{T}{3}, 2\pi + \frac{\theta_1}{2} - \sin^{-1}\left(\frac{\beta}{2}\right)\right) |0\rangle\langle 1| \\ u_2(t) &= \Omega e^{i\theta_2} p\left(t, \frac{T}{3}, \frac{2T}{3}, 2\pi - 2\sin^{-1}\left(\frac{\beta}{2}\right)\right) |0\rangle\langle 1| \\ u_3(t) &= \Omega e^{i\theta_2} p\left(t, \frac{2T}{3}, T, \frac{\theta_1}{2} - \sin^{-1}\left(\frac{\beta}{2}\right)\right) |0\rangle\langle 1|, \end{aligned} \tag{4.17}$$

with $\beta = \sin\left(\frac{\theta}{2}\right)$ is robust to the exact value of Δ, for small enough Δ.
Robustness has found widespread use in quantum computation where fabrication uncertainty, use of ensemble systems, and noise have made control challenging. The use of robust control is especially important where small deviations occur over time scales roughly on par with gate durations.
Nevertheless, if deviations are not small, if they are over a much longer (or much shorter) timescale, or if very high fidelity is sought after, then typically they have limited value. This is especially the case where variability occurs as result of design uncertainty or slow parameter drift, or when continuous gate sets are needed as for NISQ algorithms. To understand why maximum fidelities suffer as a result of improved robustness, notice that a longer pulse sequence (with multiple pulses) will necessarily incur more decoherence. Thus while pulses such as BB1 and CORPSE will reduce drift error, the overall fidelity will not be as high as a single pulse at a fraction of the duration could have yielded. We will also show this quantitatively in the next sections.

4.2.5 Supervised training method: SOMA SL

Rather than constructively producing such classes, or relying solely on optimal control theoretic methods, here we pursue the approach of machine learning a functional approximation to the general solutions. Function approximators are mathematical objects capable of reproducing arbitrary functions using families of functions [GBC16]. They are normally identified with neural networks and find extensive application in machine learning, data analysis, etc.
For the supervised approach, which is sometimes referred to as trajectory learning in the robotic control literature [AM86], we employ an optimizer to find the corresponding minima for a set of problems and a regressor $g : \mathbb{R}^D \mapsto \mathbb{R}^Q$ which maps the problem parameter space to the space of solutions to the given problem. We refer to this approach as SOMA SL (SOMA with supervised learning). A sketch of the algorithm is provided in Fig. 4.2 (a). Starting from a seed problem with solution $\boldsymbol{x}_0^*$, generated previously, for N different optimal quantum control problems parametrized by $\vec{\lambda}_1, ..., , \vec{\lambda}_N$, we generate N solutions $\boldsymbol{x}_1^*, ..., \boldsymbol{x}_N$. Then we train the neural network to find the best non-linear mapping between the original parameters and the solutions. Training is performed via a standard mean squared error (MSE) loss:

$$\mathcal{L}(\boldsymbol{w}) = \sum_{i=1}^{N} \left\| \boldsymbol{z}_i^* - g(\boldsymbol{w}, \vec{\lambda}_i) \right\|_2^2 \tag{4.18}$$

$$\boldsymbol{z}_i^* = \frac{\boldsymbol{x}_i^* - \bar{\boldsymbol{x}}}{\boldsymbol{\sigma_x}} \tag{4.19}$$

where $\bar{\boldsymbol{x}}$ is the mean value of the generated data, $\boldsymbol{\sigma_x}$ its standard deviation, and $\boldsymbol{z}_i^*$ the normalized data.

Algorithm 1 SOMA SL

```
    **Input** $\boldsymbol{w}, \vec{\lambda}_0, \vec{\lambda}_{\min}, \vec{\lambda}_{\max}$, optimizer OPT
    **Output** $\boldsymbol{w}^*$
1: $\boldsymbol{x}^* = OPT(F(\boldsymbol{x}, \vec{\lambda}_0), \nabla_x F(\boldsymbol{x}, \vec{\lambda}_0), \boldsymbol{x}_0)$        ▷ with random restart
2: for $i = 1$ to $N$ do
3:     $\vec{\lambda}_i \sim \pi(\vec{\lambda}_{\min}, \vec{\lambda}_{\max})$
4:     $\boldsymbol{x}_i^* = OPT(F(\boldsymbol{x}, \vec{\lambda}_i), \nabla_x F(\boldsymbol{x}, \vec{\lambda}_0), \boldsymbol{x}^*)$
5: end for
6: Save $\{(\boldsymbol{x}_i, \vec{\lambda}_i)\}_{i=1}^N$
7: Compute $\boldsymbol{z}_i^* = \frac{\boldsymbol{x}_i^* - \bar{\boldsymbol{x}}}{\boldsymbol{\sigma_x}}, i = 1, ..., N$
8: $\mathcal{L}(\boldsymbol{w}) = \sum_{i=1}^N \mathrm{dist}(g(\boldsymbol{w}, \vec{\lambda}_i), \boldsymbol{z}_i^*)$        ▷ with random restart
9: $\boldsymbol{w}^* = OPT(\mathcal{L}, \nabla_w \mathcal{L}, \boldsymbol{w})$
```

4.2.6 Direct-training method with back propagation: SOMA BP

An adaptive trajectory is a solution that depends on a specific parameter of the physical system, which the optimizer does not control, although it can make use of it. Usually, for many robotics applications, the system does not have an analytical model, thereby preventing direct-learning strategies. However, for quantum dynamics, we show how we can use the model to more directly train the function approximator. We refer to this approach as SOMA with back propagation (SOMA BP). A sketch of the algorithm is provided in Fig. 4.2 (b).

Prime examples of adaptive trajectories are the analytical pulses in Sec. 4.2.2. For

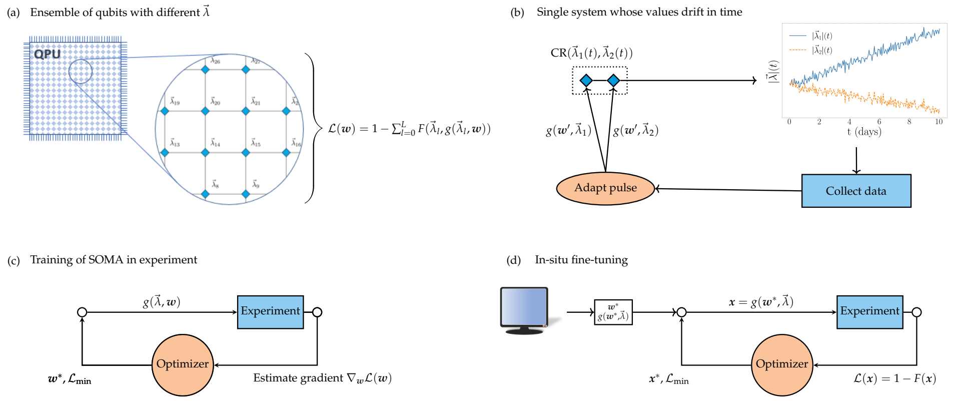

(a) Ensemble of qubits with different $\vec{\lambda}$
QPU
$\vec{\lambda}_{26}$
$\vec{\lambda}_{27}$
$\vec{\lambda}_{19}$
$\vec{\lambda}_{20}$
$\vec{\lambda}_{21}$
$\vec{\lambda}_{13}$
$\vec{\lambda}_{14}$
$\vec{\lambda}_{15}$
$\vec{\lambda}_{16}$
$\vec{\lambda}_{8}$
$\vec{\lambda}_{9}$
$\mathcal{L}(\boldsymbol{w}) = 1 - \sum_{l=0}^{L} F(\vec{\lambda}_l, g(\vec{\lambda}_l, \boldsymbol{w}))$
(b) Single system whose values drift in time
$\mathrm{CR}(\vec{\lambda}_1(t), \vec{\lambda}_2(t))$
$g(\boldsymbol{w}', \vec{\lambda}_1)$
$g(\boldsymbol{w}', \vec{\lambda}_2)$
Adapt pulse
Collect data
$|\vec{\lambda}_1|(t)$
$|\vec{\lambda}_2|(t)$
$|\vec{\lambda}|(t)$
t (days)
(c) Training of SOMA in experiment
$g(\vec{\lambda}, \boldsymbol{w})$
Experiment
Optimizer
$\boldsymbol{w}^*, \mathcal{L}_{\min}$
Estimate gradient $\nabla_{\boldsymbol{w}}\mathcal{L}(\boldsymbol{w})$
(d) In-situ fine-tuning
$\boldsymbol{w}^*$
$g(\boldsymbol{w}^*, \vec{\lambda})$
$\boldsymbol{x} = g(\boldsymbol{w}^*, \vec{\lambda})$
Experiment
Optimizer
$\boldsymbol{x}^*, \mathcal{L}_{\min}$
$\mathcal{L}(\boldsymbol{x}) = 1 - F(\boldsymbol{x})$

Figure 4.3: Four diagrams describing two possible use cases of SOMA and two possible experimental implementations of SOMA BP, respectively. (a) We consider a chip with several qubits, each one with its own Hamiltonian parameter $\vec{\lambda}$. We assume parameter variations $\delta\vec{\lambda}$ to be so large that robust pulses are generally ineffective. (b) In the second use case, we consider a single system, the parameters of which vary with time. Pulses are trained on the average cost function over ensembles of qubits. (c) The approximator is trained directly on the experimental setting by using a gradient estimator, in this case a policy gradient [SMS&al99], which usually requires large numbers of samples to be drawn from the system. (d) The regressor is first trained first in open-loop simulation and then used as an ansatz for a closed-loop optimization, which leaves the neural-network parameter untouched but modifies the output amplitude parameter (or, alternatively and if possible, the input parameters) to maximize the fidelity for specific experimental configurations. The optimization routine can be freely chosen among gradient-free algorithms, such as Nelder-Mead in Ref. [CCM11]. This method can be a viable option if the experimental setting can be simulated with sufficient precision.

instance, by using a frequency-dependent solution, DRAG eliminates the leakage inside a qutrit. Moreover, this solution parametrized by the function approximator, g, directly depends on the physical system values and can therefore be tuned if these are shifted. The cost function for an ensemble of L quantum-optimal-control (QOC) problems defined by problem parameters $\vec{\lambda}_1, ..., \vec{\lambda}_L$ is given by:

$$\mathcal{L}(\boldsymbol{w}) = 1 - \frac{1}{L} \sum_{l=1}^{L} F(g(\boldsymbol{w}, \vec{\lambda}_l), \vec{\lambda}_l) \tag{4.20}$$

where F, as before, is the figure of merit, e.g., the overlap fidelity of the operation with the target quantum gate [Nie02].
By parametrizing the solution in terms of a neural network that depends on the gate parameters, gradient-based optimization algorithms can be used to train the network directly off the above cost function. In essence, the usual back propagation of neural networks matches naturally with gradient-descent optimal-control methods such as GRAPE [KRK&al05]. Thus, while optimizing in the fidelity landscape of the controls, our algorithm is able to simultaneously train the network to adapt to the extraneous system and gate parameters. This method can also be used in combination with a more standard robust-GRAPE approach. In this case, those parameters the calibration and control of which proves difficult, can be excluded from the network input.

Algorithm 2 SOMA BP

```
    Input $\boldsymbol{w}, \vec{\lambda}_0, \vec{\lambda}_{\min}, \vec{\lambda}_{\max}$, optimizer OPT
    Output $\boldsymbol{w}^*$
1: for $i = 1$ to $L$ do
2:     $\vec{\lambda}_i \sim \pi(\vec{\lambda}_{\min}, \vec{\lambda}_{\max})$
3:     $x_i^* = \mathrm{OPT}(F(x, \lambda_i), x^*)$
4: end for
5: $\mathcal{L}(\boldsymbol{w}) = 1 - \frac{1}{L} \sum_{i=1}^{L} F(g(\boldsymbol{w}, \vec{\lambda}_i), \vec{\lambda}_i)$
6: $\boldsymbol{w}^* = \mathrm{OPT}(\mathcal{L}, \nabla_w \mathcal{L}, \boldsymbol{w})$          ▷ with random restart
```

4.2.7 Experimental adaptation

Direct experimental application of SOMA is possible both for model-based and model-free implementations. For these purposes, one must have access to a controlled distribution of $\vec{\lambda}$ values, corresponding to gate parameters, pulse parameters, and system parameters. For *in situ* optimization of the gates, one further requires access to an experimental cost function to ascertain (with low noise and bias) the merit of the pulse sequence. For model-free control learning, the system parameters should still be indexed in some way, e.g., by performing characterization, by using a proxy such as other known characteristics, or by externally tuning parameters (e.g., the qubit frequency via magnetic or Stark shifts). For model-based approaches, one can vary the parameters $\vec{\lambda}$ in software to map the solution space of the continuous gate sets. Thus, in both cases, provided that there is known variation in some parameters, then one can index them, e.g., discretely in space or (slowly varying) continuously in time, as shown in Fig. 4.3

(a) and (b) respectively.
Whichever parameters for $\vec{\lambda}$ one chooses for the experiment, the task then becomes to learn the neural-network weights for the maximal performance on the relevant device and gate defined uniquely by $\vec{\lambda}$. Two different approaches are shown in Fig. 4.3 (c) and (d). When an accurate model for the generators of the dynamics is known, then Fig. 4.3 (d) is a natural choice whereby offline (i.e., open-loop) optimization of the simulated gates is first brought up, and only once the solution class has been learned *in situ* (i.e., closed-loop) is control learning performed. In this secondary step, one can reoptimize either over the space of solutions (fine tuning optimal $\vec{\lambda}^*$) or over the space of physical controls (fine tuning optimal $\boldsymbol{x}^*$).
Closed-loop optimization directly on the experiment can be performed a number of ways, including numerical and parameter-shift approximations of the gradient, Nelder-Mead [NM65], or evolutionary algorithms [WSG&al14]. In the latter case, for example, Monte-Carlo gradient sampling can be used to estimate the update direction [MRF&al20]:

$$\nabla_{\boldsymbol{x}} F(\boldsymbol{x}) = \frac{1}{\tilde{N}} \sum_{i=1}^{\tilde{N}} F(\boldsymbol{x} + \sigma \boldsymbol{\epsilon}_i) \boldsymbol{\epsilon}_i \tag{4.21}$$

where $\epsilon_i \sim \mathcal{N}(\mathbf{0}, \mathbb{I}_Q)$, $i = 1, .., \tilde{N}$ is a normally distributed stochastic variable sampled $\tilde{N}$ times, $\boldsymbol{x}$ is the network output representing the pulse and σ is the standard deviation of the sampled pulses. This method is often referred to as natural evolution strategy [WSG&al14]. Fig. 4.3 (c) shows how the experimental gradient of the cost function can be then back propagated via the optimizer (similarly to the GRAPE implementation above) in order to update the network weights $\boldsymbol{w}^*$ directly, when the different $\vec{\lambda}$ can be sampled simultaneously.

4.3 Results

We test our methods and compare to previous optimal-control theoretic approaches. For this purpose, we train our solution networks to learn how to perform continuous gate sets for both single- and two-qubit operations. As a figure of merit of the QOC problems, we choose the gate fidelity defined in Eq. (4.9). While the gradients of the fidelity with respect to the pulse parameters x can be computed using GRAPE [KRK&al05; MGM&al11] (see Section 4.2.3 and Appendix 4.7.1), in the context of these simulations the gradient can also be obtained through automatic differentiation [BFH&al18; Ral06]. To simulate the quantum system, we use a second-order Magnus propagator as derived in Refs. [BCO&al09; DM22], and which is compatible with analytical or automatic differentiation [DM22]. For the optimization of all the parameters, we employ the algorithm L-BFGS-B [LN89].

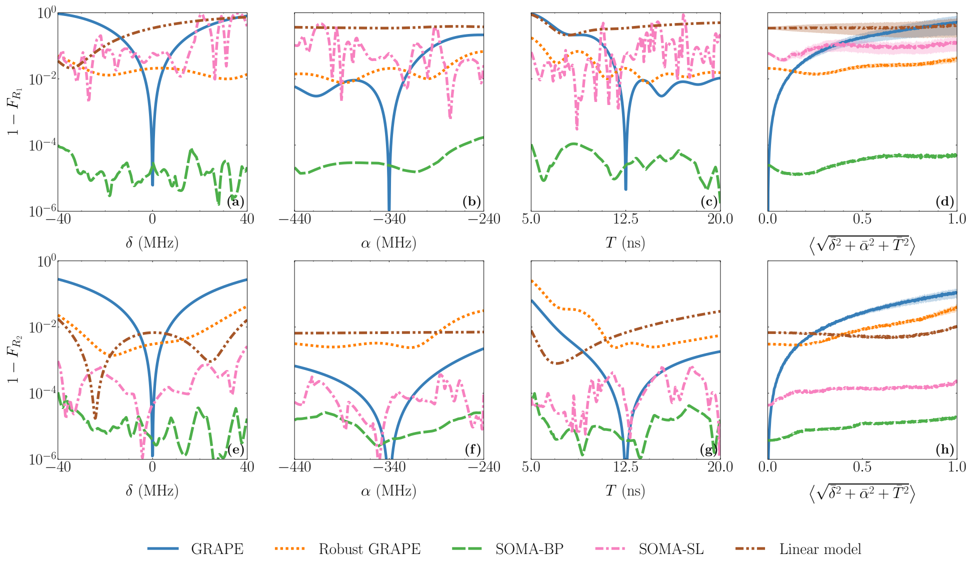


Figure 4.4: The infidelity of pulses predicted by the different optimization methods for the $R_1(\theta = \pi/2)$ gate (first row) and the $R_2(\theta = \pi/2)$ gate (second row) as a function of three different quantum control problem parameters: (a), (e) the detuning δ between the qubit frequency and the driving frequency; (b), (f) the nonlinearity α of the qutrit; and (c), (g) the total gate duration T. (d) (h) THE average performance (1000 test samples) of the algorithms as a function of the radial distance from the center $\vec{\lambda}_0$ of the parameter space. The terms $\bar{\delta}, \bar{\alpha}, \bar{T}$ indicate that the problem parameters are renormalized to the space $[0, 1]^D$ – see Appendix 4.7.2. The shaded regions around each plot line show the standard deviation of the corresponding infidelities. A more detailed discussion about the standard deviation can be found in Appendix 4.7.2. Both gates are optimized with four Fourier components for each one of the two control fields – see Eq. (4.28) – using the Hamiltonian in Eq. (4.29). The range of each parameter is given in Tab. 4.1.

4.3.1 Single-qubit gates

Single-qubit gates are the fundamental building blocks of quantum circuits. Their most general form – e.g., as they are implemented in the IBMQ compiler [IBM22] – is given by:

$$R(\theta_1, \theta_2, \theta_3) = \begin{bmatrix} \cos(\theta_1/2) & -e^{i\theta_2}\sin(\theta_1/2) \\ e^{i\theta_3}\sin(\theta_1/2) & e^{i(\theta_3+\theta_2)}\cos(\theta_1/2) \end{bmatrix}. \tag{4.22}$$

By choosing the vector parameter $\boldsymbol{\theta}$ appropriately, one can construct arbitrary single-qubit unitaries. Therefore, in any optimal quantum control problem, this triple can be considered as a vector of problem parameters, since they define the entire class of QOC

problems, the goal of which is the optimization of arbitrary single-qubit gates.
In the following section we consider an ensemble of QOC problems defined by parameters of the unitary target gate, Hamiltonian parameters, parameters of the control fields, and the evolution time T. To simplify the problem, we consider a target gate of type

$$R_1(\theta) = \begin{bmatrix} \cos(\theta/2) & \sin(\theta/2) \\ \sin(\theta/2) & -\cos(\theta/2) \end{bmatrix} \tag{4.23}$$

by setting $\theta_1 = \theta$, $\theta_2 = \pi$ and $\theta_3 = 0$ in Eq. (4.22). For $\theta = \frac{\pi}{2}$, the gate is the H gate, whereas for $\theta = \pi$ it produces the X gate.
The second family of unitaries that we consider can be obtained by setting $\theta_1 = \pi$ and $\theta_2 = \theta_3 = \theta - \pi$ in Eq. (4.22). The resulting family of gates,

$$R_2(\theta) = \begin{bmatrix} 0 & e^{i(\theta-\pi)} \\ e^{-i(\theta-\pi)} & 0 \end{bmatrix} \tag{4.24}$$

generates, among other unitaries, the X gate for $\theta = \pi$ and the Y gate for $\theta = \frac{\pi}{2}$. We would like to point out that a combination of single-qubit unitary gates as in Eq. (4.24) and Eq. (4.23) is sufficient to generate arbitrary single-qubit unitaries [NC10].
For single-qubit simulations, we consider the Hamiltonian of a superconducting transmon qubit [KSB&al20]. This system can be effectively reduced to a qutrit Hamiltonian [KSB&al20; KGM&al09], where the $|0\rangle$ and $|1\rangle$ levels provide the computational subspace and the $|2\rangle$ level represents the leakage. The drift Hamiltonian for our system [MGR&al09] reads

$$H_d = \omega_d \hat{n} + \alpha \hat{\Pi}_2 \tag{4.25}$$

$$\hat{\Pi}_j = |j\rangle \langle j| \tag{4.26}$$

$$\hat{n} = \sum_j j \hat{\Pi}_j, \tag{4.27}$$

where ω_d is the qubit frequency and α is the anharmonicity. Furthermore, we consider a control Hamiltonian of type:

$$H_c(t) = \Omega(t) e^{i\phi + i\omega t} \hat{\sigma}_+ + \Omega(t)^* e^{-i\phi - i\omega t} \hat{\sigma}_-, \tag{4.28}$$

where ω is the driving frequency and ϕ represents a time-independent phase shift between the raising $\hat{\sigma}_+$ and the lowering $\hat{\sigma}_-$ operators, which can be related to the rotating-wave approximation (RWA) [CDG98; MGM&al11] and which is, in this model, the only problem parameter influencing the control fields.
Computing the RWA with detuning δ allows us to rewrite the drift Hamiltonian as:

$$H_d = \delta \hat{\Pi}_1 + (\alpha - 2\delta) \hat{\Pi}_2, \tag{4.29}$$

$$H_c(t) = u_1(t) \hat{X}(\phi) + u_2(t) \hat{Y}(\phi) \tag{4.30}$$

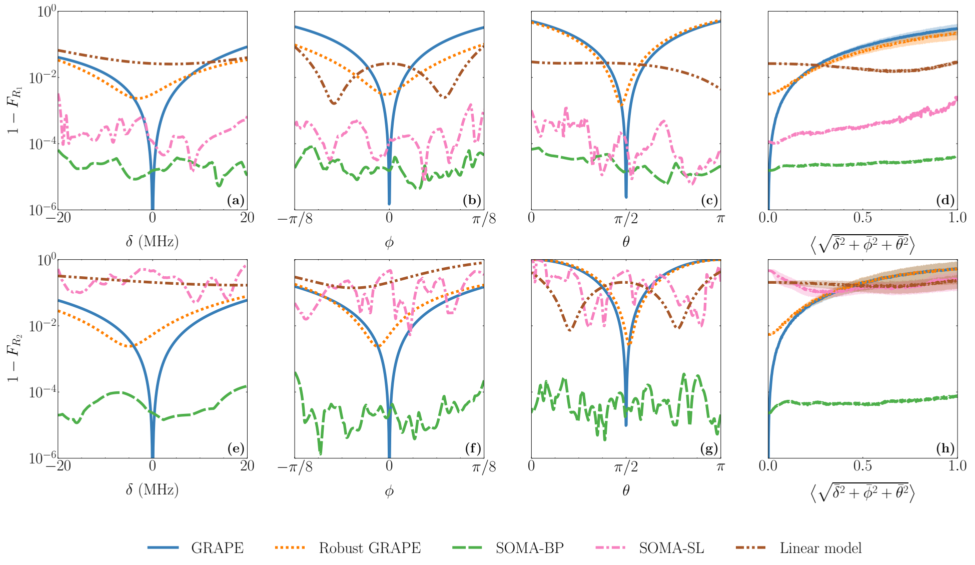


Figure 4.5: The infidelity of pulses predicted by the different optimization methods for the $R_1(\theta)$ gate (first row) and the $R_2(\theta)$ gate (second row) as a function of three different quantum control problem parameters: (a), (e) the detuning δ between the qubit frequency and the driving frequency; (b), (f) the phase error ϕ between the σ_+ and the σ_- terms; and (c), (g) the angle θ parametrizing the target gate. (d), (h) The average performance (1000 test samples) of the algorithms as a function of the radial distance from the center $\vec{\lambda}_0$ of the parameter space. The terms $\bar{\delta}, \bar{\phi}, \bar{\theta}$ indicate that the problem parameters are renormalized to the space $[0,1]^D$ (see Appendix 4.7.2). The shaded regions around each plot line show the standard deviation of the corresponding infidelities. A more detailed discussion about the standard deviation can be found in Appendix 4.7.2. Both gates are optimized with four Fourier components for each one of the two control fields for $T = 10$ ns using the Hamiltonian in Eq. (4.29).

where $\delta = \omega_d - \omega$, $\hat{X}(\phi) = e^{i\phi}\hat{\sigma}_+ + e^{-i\phi}\hat{\sigma}_-$, and $i\hat{Y}(\phi) = e^{i\phi}\hat{\sigma}_+ - e^{-i\phi}\hat{\sigma}_-$. For the control fields, we employ a Fourier ansatz:

$$u_j(t) = \sum_{k=1}^{K} x_{kj} \sin\left(\frac{k\pi t}{T}\right),\ j = 1,2 \tag{4.31}$$

with K Fourier modes. For the QOC simulations, we set the central values $\delta_0 = 0$ GHz and $\alpha_0 = -0.34$ GHz [TMW16]. The parameter vector of the QOC-problem class is given by

$$\vec{\lambda} = (\delta, \alpha, \phi, \theta, T)^T . \tag{4.32}$$

For all the single-qubit gate simulations, we consider a multidimensional rectangle centered at $\vec{\lambda}_0 = (\delta_0, \alpha_0, \phi_0, \theta_0, T_0)^T$ and with upper bounds defined by $\pm\vec{\lambda}_{\max}$. We consider four methods, which allow us to optimize multiple systems simultaneously: standard GRAPE; robust GRAPE, which uses the average GRAPE gradient over an ensemble of QOC-problems in Eq. (4.15); a supervised training method, which we refer to as SOMA SL (Algorithm 1), using both linear and nonlinear models and where GRAPE solutions are first generated and then approximated via Eq. (4.18); and the unsupervised method, which trains a neural-network pulse directly on the fidelity of an ensemble of QOC problems using back propagation on Eq. (4.20). The latter is referred to as SOMA BP (Algorithm 2).
In the following section, we consider a QOC problem with $N = 500$ time steps of a Magnus-type time integrator, which approximates the unitary temporal evolution of the quantum system [DM22]. Our pulses are parametrized as in Eq. (4.28) by $K = 4$ Fourier components according to Eq. (4.31) for each one of the two control fields $\hat{X}$ and $\hat{Y}$. The control fields are multiplied with a scaling factor equal to $1/A_0$ to ensure that the pulse amplitudes do not exceed the typical experimental maximal value of 1 GHz.
The results are divided into four blocks of four plots each (Fig. 4.4, Fig. 4.5), representing models trained on two different sets of problem parameters, the ranges of which are given in Tab. 4.1 and in Tab. 4.2. Here we limit the number of different parameters to three, although larger numbers are possible depending on the parameter range taken into account, the specifics of the physical system, the size of the neural network, and the number of systems sampled (see Appendix 4.7.2). The first three plots in each row show the infidelity as a function of one varying parameter, while the other parameters are kept at their original values given by $\vec{\lambda}_0$. The fourth plot shows the average performance of the different methods as a function of the radial distance from the initial QOC problem $\vec{\lambda}_0$. This last plot ensures that the performance of the methods is stable when different parameters are changed simultaneously across the entire range of the parameter space.
In Fig. 4.4, the problem parameters δ (the qubit-drive detuning), α (the nonlinearity of the qutrit system), and T (the gate evolution time) and the target gates $R_1(\theta)$ defined by Eq. (4.23) – first row – and $R_2(\theta)$ defined by Eq. (4.24) – second row – are considered. The blue continuous line shows the GRAPE solution for the $\vec{\lambda}_0$ parameter, whose fidelity shows an exponential decay as a function of the distance from the center of the parameter space. The orange dashed line shows the performance of robust GRAPE and the brown dashed line the performance of SOMA SL with a linear model, both with infidelities around 10^{-2}. The pink dashed line shows the performance of SOMA SL with a neural network and the green dashed line the performance of SOMA BP. We observe that for the second gate $R_2(\theta)$, both methods are able to deliver an average infidelity below 10^{-4}, whereas for the first one $R_1(\theta)$, SOMA BP clearly outperforms SOMA SL.
In Fig. 4.5, the problem parameters are as follows: drive frequency detuning δ, the phase mismatch in the control fields ϕ, and the gate angle θ, together with the target gates

	δ(MHz)	α(MHz)	ϕ	θ	$1/A_0$	T(ns)
Center	0	−340	0	$\frac{\pi}{2}$	0.01	10
Maximum	40	−240	0	$\frac{\pi}{2}$	0.01	20
Minimum	−40	−440	0	$\frac{\pi}{2}$	0.01	5

Table 4.1: The parameter range for the optimization of the single-qubit gates $R_1(\theta)$ and $R_2(\theta)$ as given in Fig. 4.4. For this simulation, we do not vary the angle parameters of the gate, but, rather, the physical parameters of the qutrit, such as δ and α and the evolution time T.

	δ(MHz)	α(MHz)	ϕ	θ	$1/A_0$	T(ns)
Center	0	−340	0	$\frac{\pi}{2}$	0.01	10
Maximum	20	−340	$\frac{\pi}{8}$	π	0.01	10
Minimum	−20	−340	$-\frac{\pi}{8}$	0	0.01	10

Table 4.2: The parameter range for the optimization of the single-qubit gates $R_1(\theta)$ and $R_2(\theta)$ as given in Fig. 4.5. In this case, we not only vary the detuning but also the angles of the gates and the phase factor ϕ.

$R_1(\theta)$ defined by Eq. (4.23) – first row – and $R_2(\theta)$ definied by Eq. (4.24) – second row. We note the same general trends as in Fig. 4.4. In particular, we observe that although both SOMA methods are closer in terms of performance for $R_1(\theta)$ with a little advantage for SOMA BP, for gate $R_2(\theta)$ SOMA BP clearly outperforms SOMA SL. We can again note that SOMA BP mostly outperforms all the other methods, providing pulses that are stable over a large range of parameters. For simulations with single-qubit gates, we use a two-layered network with 256 components per layer and eight neurons in output – the output space of the approximator has an output dimension $Q = 2K$.

4.3.2 CR gate with leakage

The cross-resonance (CR) gate is a two-qubit gate activated by microwave fields, which drive one of the qubits (target) at the frequency of the other (control). The gate is implemented in the context of quantum computing with superconducting qubits [Par06; RD10; KKL&al18; MM22]. The gate gives rise to a ZX-type interaction [Par06], which can then be used to generate different two-qubit entangling unitaries.
The CR gate can be embedded in a higher-dimensional system, e.g., where two qutrits are capacitively coupled together. In this case, the target gate is a two-qubit gate, but the unitary generated by the Hamiltonian evolution is a two-qutrit gate, thus the fidelity is only computed with respect to the computational subspace – the CR gate is constructed using the $|0\rangle$ and $|1\rangle$ levels. The Hamiltonian of a two-transmon system

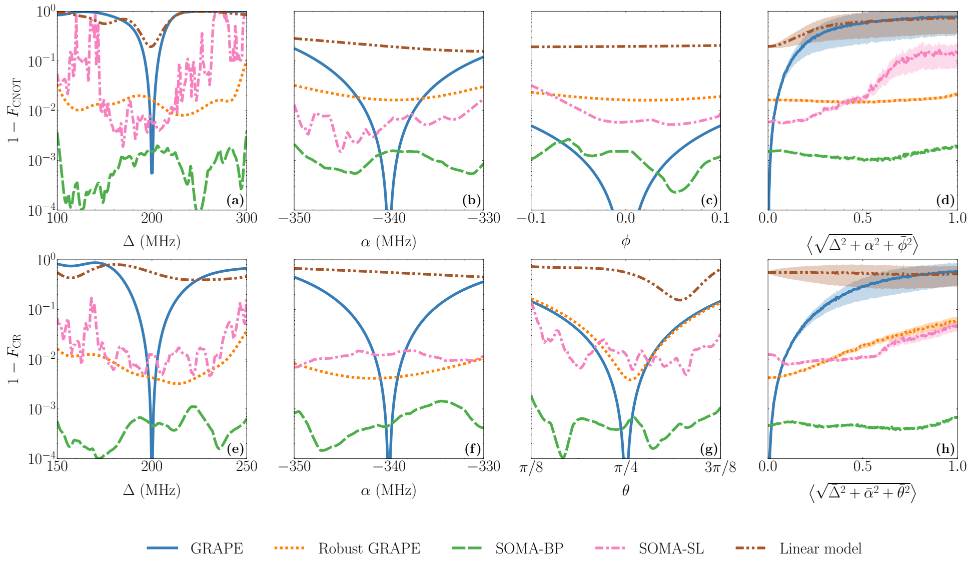


Figure 4.6: The infidelity of pulses predicted by the different optimization methods for the direct controlled-NOT (CNOT) gate (first row) and the $CR(\theta)$ gate (second row) as a function of different quantum control problem parameters: (a), (e) the frequency difference Δ between the two qubits; (b), (f) the nonlinearity α of the qutrits; (c) the phase error ϕ between the control fields $\hat{\sigma}_+$ and $\hat{\sigma}_-$; (g) the angle θ parametrizing the CR continuous family of gates given by Eq. (4.38). Plots (d) and (h) show the average performance (1000 test samples) of the algorithms as a function of the radial distance from the center $\vec{\lambda}_0$ of the parameter space. The terms $\bar{\Delta}, \bar{\alpha}, \bar{\phi}, \bar{\theta}$ indicate that the problem parameters are renormalized to the space $[0,1]^D$ – see Appendix 4.7.2. The shaded regions around each plot line show the standard deviation of the corresponding infidelities. A more detailed discussion about the standard deviation can be found in Appendix 4.7.2. Both gates are optimized with 30 Fourier components for each one of the four control fields and for total gate duration $T = 90$ ns.

reads [RD10]

$$\begin{aligned} H(t) &= H_d + H_c(t) \\ H_d &= \sum_{j=1}^{2} \left(\omega_j \hat{b}_j^\dagger \hat{b}_j + \alpha_j \hat{b}_j \hat{b}_j^\dagger \hat{b}_j^\dagger \hat{b}_j\right) + J\left(\hat{b}_1 \hat{b}_2^\dagger + \hat{b}_1^\dagger \hat{b}_2\right) \\ H_c(t) &= \sum_{j=1}^{2} \Omega(t) e^{i\omega_j t + i\phi} \hat{b}_j^\dagger + \Omega(t)^* e^{-i\omega_j t - i\phi} \hat{b}_j, \end{aligned} \tag{4.33}$$

where J is the coupling strength of the transmon-transmon interaction, b_j is the lowering operator on the jth transmon, Ω is the driving field, and a Duffing-oscillator approximation is performed [KGM&al09]. A further standard RWA allows us to simplify the problem, introducing at the same time the notation from Eq. (4.25), such that

$$H_d = \Delta \hat{n}_1 + \alpha(\hat{\Pi}_2^{(1)} + \hat{\Pi}_2^{(2)}) + J\left(\hat{b}_1 \hat{b}_2^\dagger + \hat{b}_1^\dagger \hat{b}_2\right) \tag{4.34}$$

$$H_c(t) = \sum_{j=1}^{2} \left(u_1^{(j)}(t) \left(e^{i\phi}\hat{b}_j^\dagger + e^{-i\phi}\hat{b}_j\right) + u_2^{(j)}(t) \left(e^{i\phi}\hat{b}_j^\dagger - e^{-i\phi}\hat{b}_j\right) \right), \tag{4.35}$$

where $\Delta = \omega_1 - \omega_2$. As control operators, we use the projectors $\hat{X}_j(\phi) = e^{-i\phi}\hat{b}_j + e^{i\phi}\hat{b}_j^\dagger$ and $i\hat{Y}_j(\phi) = e^{i\phi}\hat{b}_j^\dagger - e^{-i\phi}\hat{b}_j$ both on the control qutrit ($j = 1$) and the target qutrit ($j = 2$), as in Eq. (4.28), such that

$$H_c(t) = \sum_{j=1}^{2} u_1^{(j)}(t)\hat{X}_j(\phi) + u_2^{(j)}(t)\hat{Y}_j(\phi) \tag{4.36}$$

where $u_j^{(1)}(t)$, $u_j^{(1)}(t)$ are the control fields on the jth qutrit (for the $\hat{X}_j$ and the $\hat{Y}_j$ operators respectively). Note that all the fields operate at frequencies near-resonant to the target qubit. The Hamiltonian parameters are centered at values $\Delta_0 = 0.2$ GHz, $\alpha_0 = -0.34$ GHz, $J_0 = 0.01$ GHz. For each control field, we use a Fourier parametrization [CCM11; KKL&al18]:

$$\forall j = 1, 2,\ \forall i = 1, 2:\ u_i^{(j)}(t) = \sum_{k=1}^{K} x_k^{i,j} \sin\left(\frac{k\pi t}{T}\right) \tag{4.37}$$

We further assume, as in the single qubit case – see Eq. (4.28), that the control fields are influenced by a phase factor ϕ, which then counts as a QOC-problem parameter in the two-qubit simulations.

For two-qubit gates with leakage we present two different simulations, one for the CNOT gate alone and one for a family of CR-like gates with the following parametrization:

$$\mathrm{CR}(\theta) = \exp[i\theta(Z \otimes X)] \tag{4.38}$$

.

For the two-qutrit gates we use a two-layered neural network with 500 neurons per layer and 120 neurons in output. Here, the output of the approximator has a dimensionality $Q = 4K$, where $K = 30$ is the number of Fourier frequencies for each one of the four control fields – see Eq. (4.37). The time evolution is given by 5000 Magnus steps.

	Δ(MHz)	α(MHz)	J(MHz)	ϕ	A_0	T(ns)
Center	200	−340	10	0	0.05	90
Maximum	300	−330	10	0.1	0.05	90
Minimum	100	−350	10	−0.1	0.05	90

Table 4.3: The parameter range for the optimization of the two-qubit CNOT gate. The range of values Δ, α and ϕ, as well as the evolution time T, are determined by considering typical experimental settings of state-of-the-art superconducting quantum circuits [MM22; MMM20; HK21]. For the value of ϕ, we assume a small phase error influencing the $\hat{X}$ and $\hat{Y}$ control fields both on the target and the control qubits.

	Δ(MHz)	α(MHz)	J(MHz)	ϕ	θ	A_0	T(ns)
Center	200	−340	10	0	$\frac{\pi}{4}$	0.05	90
Maximum	250	−330	10	0	$\frac{\pi}{8}$	0.05	90
Minimum	150	−350	10	0	$\frac{3\pi}{8}$	0.05	90

Table 4.4: The parameter space for the optimization of the two-qubit CR gate as given in Eq. 4.38. Compared to the simulation of the CNOT gate, given in Tab. 4.3, the range of the target-control frequency detuning Δ is reduced but the continuous parameter θ is also considered.

The results of the pulse class learning are shown in Fig. 4.6. The evolution time is chosen as $T = 90$ ns, an improvement of about a factor of 2 over typical experimental durations. The first row of plots shows the results for the CNOT gate, where variations of the parameters Δ (the frequency difference between the control and target qubits), α (the nonlinearity of the qutrits), and ϕ (the phase term of the control fields) are considered. The second line shows the results for the CR(θ) gate as defined in Eq. (4.38). The maximal range for each parameter for each one of the two gates is described in Tab. 4.3 and Tab. 4.4 respectively. We observe that, for both gates, the neural-network pulse trained with SOMA BP outperforms the other algorithms. For the CNOT gate, it can produce pulses that are robust over a very large detuning range ($|\Delta_{\text{max}} - \Delta_{\text{min}}| =$ 200 MHz) by training on just 100 different systems sampled from a uniform distribution. For the CR gate, we choose a range of 100 MHz, while also taking into account the dependence on the angle θ. It is possible that a very large amount of samples or long training times could improve the performance of SOMA SL further; however as the size of the Hilbert space increases, the optimization of large sample numbers on classical machines can become computationally too time consuming.

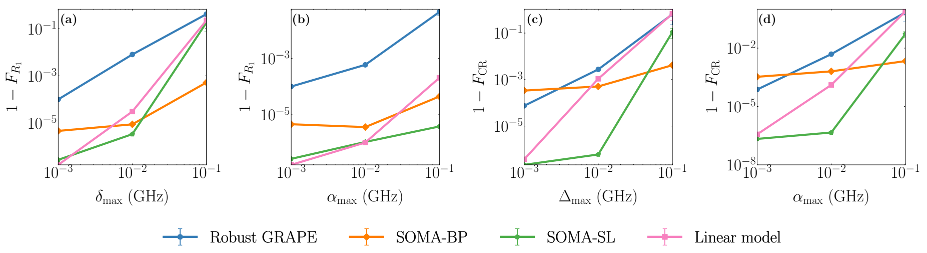


Figure 4.7: A comparison of four different generalized optimization methods as a function of the parameter space size, when only one parameter range is varied, while all the others are kept constant at 10^{-3} GHz: (a), (b) the results for the single-qubit gate $R_1(\theta)$; (c) and (d) the results for the CR(θ) gate. Both axes are on a logarithmic scale and represent the average infidelity over 1000 test samples. The values $\delta_{\text{max}}, \alpha_{\text{max}}$, and Δ_{max} on the horizontal axes represent the maximum range of the corresponding problem parameter δ, α, Δ. The maximum range defines the parameter space over which the QOC problems are sampled. We observe how the neural network trained with the supervised algorithm outputs pulses with higher average fidelity than the other methods but it fails nonetheless when the detuning variation in the two different systems is increased up to the order of 100 MHz. In this case, the model trained with SOMA BP, however, still outputs high-fidelity pulses.

4.4 Performance analysis

In this section, we compare the different meta-optimization methods as the size of the QOC problem-parameter space increases. In particular, we vary the order of magnitude of the laser-frequency detuning δ, the qubit-frequency detuning Δ for the two-qubit gate, and the nonlinearity α for both the single-qubit gate and the CR gate. For the sake of this analysis, we only vary the maximal range of one single parameter at a time, while the other problem parameters have a fixed maximal range of 10^{-3} GHz. As hyperparameters for the different algorithms, we use the same values that have proved to be effective in the previous simulations. For SOMA SL, we use up to 10000 sample problems optimized with GRAPE, whereas for SOMA BP we optimize the average fidelity with 500 system samples for the single-qubit gate and 100 systems for the two-qubit gate.
The results are shown in Fig. 4.7. We observe that although supervised training using the minima produced by GRAPE shows lower infidelities for small variations of the detuning (up to 10 MHz), it fails when this value is increased to 100 MHz, whereas the neural network trained with back propagation of the fidelity still produces valid optima of the QOC problem. In particular, we observe a crossing between 10 and 100 MHz for both types of detuning (δ and Δ) and for the nonlinearity α in the two-qutrit case, where the performance of the supervised method (SOMA SL, green line) worsens dramatically, whereas SOMA BP (orange line) is able to keep the fidelity at high, experimentally valid values. In general, we observed that SOMA SL significantly outperforms SOMA BP for small parameter variations, where the precision of the non-linear regression is high

enough to reproduce the pulse variations perfectly (see also Sec. 4.7). As a consequence, SOMA SL can be a useful tool to achieve adaptive robustness against small parameter variations, where it clearly surpasses SOMA BP, whereas the latter shines when the parameter variations and the number of Fourier components are comparatively larger. Furthermore, SOMA BP is capable of handling large variations of multiple parameters at the same time, as shown in Fig. 4.5. In all cases, at least one neural-network approach outperforms robust GRAPE and the linear regression model significantly.
We argue that sampling large amounts of problem parameters could actually lead to a better performance of SOMA SL for *in-situ* physical systems or numerical simulations where sampling proves fast and efficient. However, as we show in Fig. 4.8, it seems that for SOMA SL (green line) there exist systems where its improvement as a function of the sample size is limited, which makes SOMA BP (orange line) a better option, if its implementation is possible. Likewise robust GRAPE (blue line) and the linear model (pink line) gain limited benefit from increasing sample sizes. In particular for SOMA SL, we can often observe behaviors such as branching and outliers in the training data set, which probably contribute to a loss in the quality of the approximations. One may try to increase the precision of the model by using loss functions that are sensitive to outliers, such as the Huber loss [Hub64], or to restrict the use of SOMA SL to systems where limited parameter drifting does not prevent the regressor from learning a high-quality representation of the solution space.
We also study whether varying the chosen samples during training (i.e., minibatching) can affect the performance of our algorithms; in particular for robust GRAPE and SOMA BP. Computing the gradient over a batch of samples gives rise to a batched version of the algorithms. More specifically, a batched version of robust GRAPE, called bGRAPE, has been studied in Ref. [WDD&al19], where impressive robustness is achieved by combining batches of problem parameters with momentum-based stochastic gradient descent. This is, of course, a different algorithm than L-BFGS-B, i.e., the optimization algorithm employed in all simulations discussed in this paper, and it does not similarly guarantee near-quadratic convergence [LN89]. For SOMA SL, computation of the gradient based on the MSE loss over a batch of samples leads to standard neural-network training with stochastic gradient descent. For the systems we consider, we do not observe an improvement of bGRAPE over robust GRAPE (called sGRAPE in Ref. [WDD&al19]) neither with L-BFGS-B nor with ADAM [KB15]. We believe that this is due to the use of Fourier components, which allow for more controllability of the quantum system [MGM&al11], and the use of curvature information granted by both algorithms. Nonetheless, exploration of the effect of varying samples during training remains an interesting perspective worthy of further studies and commitment, both in the context of adaptive and robust control.

In the last part of the analysis, we discuss the scaling of the network approximations as we increase the output dimension, the number of qubits and the input dimension. Since we aim at controlling single- and two-qubit gates, the number of expected controls only scales linearly with the number of qubits; and since the weak coupling between qubits drops off roughly exponentially with distance, we also do not expect the search complexity in state space to increase dramatically. In simulations, we expect both algo-

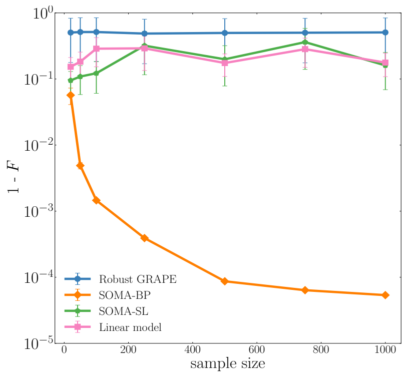


Figure 4.8: An example of the average performance (1000 test samples) of robust GRAPE, SOMA SL, SOMA BP and the linear model as a function of the number of (training) systems sampled for the single-qubit $R_2(\theta)$ gate with the same parameters as in Fig. 4.5. We observe here how in this case the average infidelity in SOMA BP decreases as a function of the sample size, whereas the other methods show little to no improvement.

rithms to behave similarly to GRAPE (or a different gradient-based control algorithm, if this is implemented) as the dimension of the Hilbert space increases. This is due to the underlying time evolution, which in both cases is given by the Trotterization or, as in our case, the Magnus expansion. The approximation of the quantum propagator affects the gradient-based optimization for single QOC problems, which generates the target data for SOMA SL, but also acts as an activation function for the neural network [WDD&al19] in SOMA BP.
As a consequence, we do expect the state space and the equations of motion to scale exponentially with the number of qubits, which is a general problem for all control and compilation tasks. Just as in the general case, we expect a combination of informed state-space parametrizations (such as tensor networks and sparse algebras) and of quantum-aided optimization (as in Sec. 4.2) largely to address this important problem. Moreover, by considering increasing numbers of qubits, the number of pulse parameters and control fields increases consequently. However control fields acting on different qubits usually commute, which can dramatically decrease correlations between the pulse components. Therefore, one does not need, in general, to have a single neural network outputting all pulse parameters at once, but, rather, several different neural networks, one for each group of control fields, which commute between each other. For SOMA SL, we need to generate a large data set of QOC-problem solutions by means of a standard quantum control algorithm, which may be slow for many-qubit systems. However, this task can be easily parallelized, since the different optimizations are independent of each other. In this case, the main obstacle is not represented by the nonlinear regression over the data, but, rather, by the quality of the data generated by the quantum control algorithm for each solution in the parameter space. Since the problem is high dimensional, the solution

space will probably exhibit structures such as branching and outliers that are difficult to include in the nonlinear regression. A possible option here is to employ algorithms for data reduction and clustering, in order to obtain a high-quality representation of the solution space.
As for SOMA BP, one may distinguish between simulation and experimental implementation. In the former case, the evaluation of the infidelity and its gradient as a function of the time evolution represents the main bottleneck – see Tab. 4.5 in Appendix 4.7.2 – together with vanishing gradients, which are also a well-known problem for other NISQ use cases and represent one of the main obstacles to any experimental application. Vanishing gradients could also be tackled with alternatives to back propagation (see, e.g., Ref.[LZF&al14]). Finally, for large output spaces and very deep networks, GPU training and stochastic gradient-descent algorithms may provide useful speedup, as it is the standard in deep learning.

4.5 Conclusion

In this work, we show how to engineer solutions of problems in quantum optimal control that depend on problem parameters located outside the optimization routine. This includes physical system parameters, other external parameters such as the pulse time or bandwidth, and gate parameters such as rotation angles.
We therefore propose two methods to learn large classes of quantum gate-synthesis problems, SOMA SL and SOMA BP. We show through experimentally relevant examples that these methods prove able to learn adaptive solutions to generalized QOC problems. The output gates have fidelities that remain very high over the entire continuous parametrization of the gate sets, for typically large ranges as would be encountered experimentally. These continuous gate sets provide the opportunity to be used as computational primitives in compilation tasks, in NISQ variational algorithms, and for entire arrays of qubits rather than individually optimized ones.

4.6 Data and Code

Data and code are stored on FZJ servers and are available on request. Please contact the authors at franz3105@gmail.com or f.motzoi@fz-juelich.de.

Acknowledgements

We thank Michael Schilling, Boxi Li, Martino Calzavara and Markus Schmitt for useful discussions. We acknowledge support from the German Ministry for Education and Research, under the QSolid project, Grant No. 13N16149, the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy – Cluster of Excellence Matter and Light for Quantum Computing (ML4Q) EXC 2004/1 – 390534769, the Jülich Supercomputing Center, and from AIDAS: AI, Data Analytics and Scalable Simulations, which is a Joint Virtual Laboratory gathering the Forschungszentrum Jülich (FZJ) and the French Alternative Energies and Atomic Energy Commission (CEA). The simulations were realized using the JAX [BFH&al18] and FLAX [HLO&al20] libraries.

4.7 Appendix

4.7.1 Gradients of the fidelity

GRAPE [KRK&al05; dSG&al11] is the standard open-loop gradient-based method for QOC solutions. It can be implemented together with a Magnus-based propagator – see [DM22]. The gradient can then be used in combination with a gradient-based optimization algorithm, e.g., the L-BFGS-B algorithm [BNS94; LN89]. Variations of GRAPE exist which exploit the advantages of parametrizing the pulse according to a specific set of basis functions (e.g. Fourier basis) [MGM&al11] or compute the higher order derivatives of the fidelity with respect to the pulse [dSG&al11; DMJ&al20]. In this paper, GRAPE is always used in combination with L-BFGS-B, which uses fast Hessian estimation [Vir&al20].
For a given set of parametrized functions $s_k : t \mapsto s_k(t),\ k = 1, ..., K$ describing the time dynamics of the control fields $u_j(t)$ with control parameters x_{kj} – see Eq. (4.28) and Eq. (4.37) –, and which can be time-sliced in values $s_{ki} = s_k(t_i), i = 0, ..., N_{\text{evo}}$, the gradient of the cost function with respect to the control parameters can be computed using the chain rule [MGM&al11]:

$$\frac{\partial F}{\partial x_{kj}} = \frac{\partial u_{ij}}{\partial x_{kj}} \frac{\partial F}{\partial u_{ij}} \tag{4.39}$$

where the k index runs over the number of basis functions components, the i index over the time-slice and the j index over the different control operators. In a similar way, this can be applied to a neural-network parametrization $g : \mathbb{R}^D \mapsto \mathbb{R}^Q, \vec{\lambda} \longrightarrow g(\vec{\lambda})$ of the GRAPE pulse, mapping a given number of meta-parameters to the pulse space, the original formula can be rewritten to output the gradients of the fidelity with respect to the neural-network parameters $(w_{ml},\ b_l)$:

$$\frac{\partial F}{\partial w_{ml}} = \frac{\partial x_{kj}}{\partial w_{ml}} \frac{\partial u_{ij}}{\partial x_{kj}} \frac{\partial F}{\partial u_{ij}} \tag{4.40}$$

$$\frac{\partial F}{\partial b_l} = \frac{\partial x_{kj}}{\partial b_l} \frac{\partial u_{ij}}{\partial x_{kj}} \frac{\partial F}{\partial u_{ij}} \tag{4.41}$$

with the same indexing as Eq. (4.39). In this case, the neural-network output values, which give the coefficients of the time-dependent basis functions, i.e., the terms

$x_{ik} = g(\vec{\lambda})_{ik}$ depend on the QOC problem parameters $\vec{\lambda}$.

4.7.2 Implementation details

Comparison of SOMA methods		
Algorithm parameter	SOMA SL	SOMA BP
QOC cost function evaluations	$n_{\text{fev}} \cdot N$	$n_{\text{fev}} \cdot L \cdot N_{\text{guesses}}$
Computation time	$t_{\text{GRAPE}} \cdot N + t_{\text{REG}}$	$t_{\text{BPROP}} \cdot L$
Number of samples	N	L

Table 4.5: Comparison of SOMA SL and SOMA BP in terms of their performance features. n_{fev} and t_{GRAPE} are here the maximal number of function evaluations and computation time required by L-BFGS-B with GRAPE to solve a single quantum control problem. t_{REG} is the time required for the neural-network regression. t_{BPROP} is the time required by the neural network back propagation for one sample.

For every physical system considered – single qubit gate with leakage and two-qubit gate with leakage, each one with different target gates and system parameters – we first define an interval for each parameter. Values are sampled from a uniform distribution over the hyper-volume defined by the intervals of parameters. The system together with the interval of parameters defines a family of QOC problems, which we analyze with one of 3 methods: Robust control with GRAPE, which simply seeks for the best pulse for a set of different systems, SOMA SL, which first solves a sample of systems and then performs a regression over the sampled values – for the sake of completeness we consider here both a linear and a non-linear approach – and SOMA BP, where the network is trained directly on the average infidelity of an ensemble of systems with back propagation.
In the case of Robust GRAPE, we sample L systems and run the optimization with random restart, i.e., we run the optimization $N_{\text{guesses}} = 5$ times with different conditions and then choose the pulse with the smallest average infidelity.
For SOMA SL, we sample up to $N = 10000$ points within the parameter space. For each one of these, we optimize the corresponding QOC problem with GRAPE – this can be performed with any proper optimal quantum control method – and then train the model to map the corresponding parameter to the optimal pulses.
As for SOMA BP, we sample L systems and run the neural-network training with random restart. The number of total samples required by the regression is usually larger than the one needed by the other two methods, the corresponding single-system optimization is nonetheless much faster and can be run in parallel. Therefore we set $N = 10000$ for each simulation, whereas for the sake of comparison and due to the similarity of the other two methods always use the same number of samples (either $L = 500$ for the single qubit case or $L = 100$ for the two-qubit case). Both SOMA methods employ two-layered neural networks with 256 neurons per layer for the single qubit system and 500 neurons

per layer for the two-qubit system. The linear model performs multi-linear regression on the same data used by SOMA SL.
For the random restart of SOMA methods, we run the optimization $N_{\text{guesses}} = 5$ times with different initial conditions and then test the quality of the predictions on a test set. Afterwards we choose the model with the lowest average test infidelity. For both algorithms, we use the L-BFGS-B algorithm. Since the training of SOMA SL, which uses a MSE loss, is much faster than SOMA BP, we do not limit its maximal number of iterations. For SOMA BP, however, this is limited to 6000 for the single qubit gates and to 7000 for the two-qubit gates. Here, the input parameters for both SOMA SL and SOMA BP should be re-scaled to lie e.g., within the range [0,1]. For SOMA BP, this can be avoided in certain cases, as long as the size of the pulses remains large enough. For details about the performance of the algorithms, see Tab. 4.5.
For each system, we evaluate the performance of the neural-network pulses on the entire parameter space. In order to visualize the performance of the algorithms considered in a one-dimensional plot, we consider the normalized parameter space, where all parameters axes are re-scaled to the interval [0,1].

$$f : \mathbb{R}^D \mapsto [0,1]^D, \vec{\lambda} \longrightarrow \frac{\vec{\lambda} - \vec{\lambda}_{\text{min}}}{|\vec{\lambda}_{\text{max}} - \vec{\lambda}_{\text{min}}|} \tag{4.42}$$

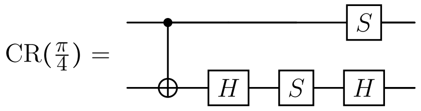
By then considering D-dimensional spheres of radius $r \in [0,1]^D$, we sample $N_s = 1000$ systems on the surfaces of such spheres and compute the average infidelity over these systems. By doing so, we can evaluate the performance over the parameter space, thereby ensuring that our methods are effective for every combination of problem parameter values. The result is then averaged over N_s samples, i.e., we plot the mean and its standard deviation. The latter is pictured as a shady region.
We also want to consider how the standard deviation of the average infidelity behaves when the different algorithms are tested against a batch of quantum systems sampled according to the parameter space. In particular, the standard deviation of the infidelity for N_s test systems is bounded by the average performance of the algorithms:

$$\sigma(\boldsymbol{w})^2 = \frac{1}{L} \sum_{i=1}^{N_s} F(\boldsymbol{w}, \vec{\lambda}_i)^2 - F_{\text{test}}^2 \leq 1 - F_{\text{test}}^2 \tag{4.43}$$

where F_{test} is the average fidelity of the trained pulses on the N_s test systems. For SOMA BP, assuming that over-fitting is negligible, we have $\sigma(\boldsymbol{w})^2 \leq 1 - (1 - \mathcal{L}(\boldsymbol{w}))^2$. which means that the fidelity of the algorithms considered is guaranteed not to drop significantly below the average performance showed in Fig. 4.4, Fig. 4.5 and Fig. 4.6.

4.7.3 Fidelity of discrete gates and their performance

In this section we briefly illustrate how analog gates are expected to outperform equivalent circuit decomposition with sequences of traditional discrete gates. In particular, we first consider the standard universal set of quantum gates $S = \{\text{CNOT}, T, S, H\}$ used in Fig. 4.1. We searched for decompositions to determine the correct compilation sequence for the aforementioned gates. In particular, we consider the CR gate defined in Eq. (4.38) for an angle of $\theta = \frac{\pi}{4}$. The decomposition is given by the following circuit:

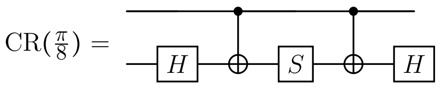


This decomposition consists of 1 entangling gate (the CNOT) and 4 local unitary operations and the representation is exact. Note that CNOT has to be implemented efficiently on the chosen quantum computing platform, which probably anyway requires quantum control. An entangling gate in the context of superconducting circuits usually requires $T \sim 150$ ns [KWS&al21]. This is a little bit slower than the CR gate presented here but shows the equivalency of these gate sets. Nonethless, SOMA also allows learning both the CR and CNOT as part of the same continuous gate set, thereby saving the additional cost of single qubit gates (which take typically at least $T \sim 15$ ns each).

	Number of gates				
Angle θ	N_{CNOT}	N_T	N_S	N_H	highest fidelity
$\frac{\pi}{\sqrt{2}}$	1	2	4	1	0.9819
$\frac{\pi}{\sqrt{3}}$	2	3	1	2	0.9778
$\frac{\pi}{\sqrt{5}}$	0	2	3	2	0.9728
$\frac{\pi}{\sqrt{7}}$	2	3	0	2	0.9999

Table 4.6: Results of searching for CR gate decompositions for four different angles: $\frac{\pi}{\sqrt{2}}, \frac{\pi}{\sqrt{3}}, \frac{\pi}{\sqrt{5}}, \frac{\pi}{\sqrt{7}}$, obtained with the help of exhaustive search (up to 10 circuit layers) and stochastic descent (up to 20 circuit layers). The universal gate set is S. We observe here that while for $\frac{\pi}{\sqrt{7}}$ we can find a suitable decomposition with high fidelity, this is not the case for the remaining angle values. Moreover, two of the best decompositions contain two CNOT gates, which again lead us to much longer gate evolution time.

Where the advantage really shows up is for a gate angle that does not belong to the same entanglement class as CNOT. It is clear here, how continuously parametrized gates and consequently SOMA can be beneficial to quantum compilation. A CR gate with an angle $\theta = \frac{\pi}{8}$ also requires 5 gates to be implemented, as shown by the following circuit,



but its circuit contains two CNOTs. Since the CR($\frac{\pi}{8}$) can take roughly half the time of a CR($\frac{\pi}{4}$), this implies the CNOT-based circuit can be as much as 7 times slower than the continuously parametrized gate.

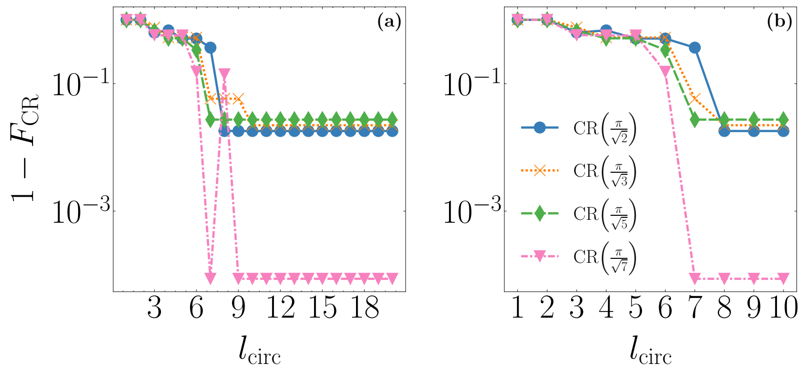


Figure 4.9: Infidelities reached by applying stochastic descent search (a) and brute-force (exhaustive) search (b) on the CR the gates defined by the angles in Tab. 4.6 in order to decompose them in discrete circuits. As we can observe, both algorithms provide us with the same results. However, they cannot find a circuit representation with $F > 0.98$ for the first three gates, thus indicating that circuits decomposing these gates with a higher fidelity are longer, more error-prone, and harder to discover.

For other angles, it becomes increasingly difficult to find good circuits, with only partial approximations being possible at reasonably short depth. In order to test the quality of the approximation, we search for optimal discrete circuits representing a quantum circuit according to a given parametrization. We use exhaustive search [Nie00] of all possible circuits (up to circuit depth 10) and stochastic descent, a special type of structured random search – see RL approach in [BDS&al18] – (up to circuit depth 20) to search for optimal decompositions of discrete gates and try to reproduce the chosen circuit with increasing number of unitaries. The results are given in Tab. 4.6 and Fig. 4.9 for the CR gate with angles $\frac{\pi}{\sqrt{2}}, \frac{\pi}{\sqrt{3}}, \frac{\pi}{\sqrt{5}}, \frac{\pi}{\sqrt{7}}$. We see that although the fidelity of the discrete gates increases with the size of the quantum circuit, in three cases it cannot reach the value of F=0.99 for circuits of depth smaller than 20. In the case of $\theta = \frac{\pi}{\sqrt{7}}$, a valid decomposition with fidelity F=0.9999 is found. The search is performed exhaustively for $l_{\mathrm{circ}} < 10$ and then using stochastic descent for $l_{\mathrm{circ}} > 10$. Moreover, the decomposition of $\mathrm{CR}(\frac{\pi}{\sqrt{7}})$ contains 2 CNOT gates, which again implies a slowdown of the gate execution time. Other examples for gates with superior analog performance can be seen, e.g., Ref. [Fox&al20], with superior performance especially expected for variational circuits.

5 Hybrid discrete-continuous compilation of trapped-ion quantum circuits with deep reinforcement learning

Disclaimer: A modified version of this chapter was published in Ref. [PSJ&al24]. FP, MS and SJ developed the theoretical framework of the paper. The PS-LSTM algorithm was developed by FP and SJ during FP's master project [Pre20]. Afterward, FP and MS developed the work further to adapt it to significantly more complicated scenarios than the ones considered in the previous work. In particular, MS developed the gate decomposition – see Appendix 5.8.1. FP developed the code for fast variational gates and ran all the simulations. LMT suggested the use of the curriculum. The paper was written by FP with inputs from all authors. HJB and FM supervised the project.

Shortening quantum circuits is crucial to reducing the destructive effect of environmental decoherence and enabling useful algorithms. Here, we demonstrate an improvement in such compilation tasks via a combination of using hybrid discrete-continuous optimization across a continuous gate set, and architecture-tailored implementation. The continuous parameters are discovered with a gradient-based optimization algorithm, while in tandem the optimal gate orderings are learned via a deep reinforcement learning algorithm, based on projective simulation. To test this approach, we introduce a framework to simulate collective gates in trapped-ion systems efficiently on a classical device. The algorithm proves able to significantly reduce the size of relevant quantum circuits for trapped-ion computing. Furthermore, we show that our framework can also be applied to an experimental setup whose goal is to reproduce an unknown unitary process.

5.1 Introduction

The last decade has seen significant progress in the development of quantum computing architectures [Aru&al19]. While scalable fault-tolerant quantum computers are still out of reach in the near future, noisy, intermediate-scale quantum (NISQ) computers may already offer some benefits over classical ones for specific computational tasks [Pre18; Bra&al23]. In particular, variational algorithms [Per&al14; Cer&al21b], where most of the operations depend on several continuous parameters, have emerged as a suitable

class of methods that could potentially achieve quantum speed-up on NISQ devices.
Implementing a high-level quantum algorithm both on fault-tolerant and NISQ devices requires adequate methods to compile it in the set of universal quantum gates available to the hardware. While several frameworks for compilation of quantum circuit-based algorithms on physical platforms are being developed [KMO&al23; YIL&al21; Qis23], common available approaches, such as heuristic and automated search [VDO&al19; WHB20], in many case cannot output an optimal circuit for a specific target operation [Mar&al22]. In digital quantum computers, compilers implement a general quantum circuit through a discrete set of universal quantum gates. In real physical quantum devices, however, and more generally in analog computation, an additional layer of complexity is present, due to the necessity of optimizing continuous gate parameters to reproduce target unitaries. These parameters may depend, e.g., on the specific Hamiltonian employed in the quantum computing platform, such as the XY-Hamiltonian or the Mølmer–Sørensen interaction for trapped ions [SM99; MS99], the cross resonance interaction for IBM quantum computers [Mal21] or the Fermi-Hubbard model for neutral atoms [DW16]. As a result, when taking into account physical parameters, variational algorithms, and generally continuous gate sets [PCM22], one must supplement the circuit compilation task with a subsequent optimization of the parameters defining the individual constituent gates. For the optimization of continuous parameters, we have several options available, e.g., gradient-based algorithms [Dao&al22], evolutionary algorithms [Sim13] and direct search [NM65]. For the compilation of the circuit gate structure, a standard approach is given by methods based on the Solovay-Kitaev algorithm [DN06], different circuit factorization strategies [VW04; WKW&al16], graph path traversal algorithms, such as the A^* algorithm [DST&al20], semi-definite programming and and various machine learning methods, including reinforcement learning [Mar&al22]. More specifically, deep reinforcement learning has been recently successfully implemented for the optimization of discrete quantum circuits [Mor&al21; FNM&al21; Ost&al21; SEL&al22; BSK&al21; ZZZ&al20; YLB21].
In reinforcement learning (RL) [SB18], an agent learns to maximize a properly engineered reward signal by interacting with an environment, which encodes the optimization task to solve. RL has already been applied to solve various challenging tasks, including, e.g., surpassing human performance in certain classes of games [Sil&al17] or in complex, computationally expensive problems such as protein folding [Jum&al21]. Projective Simulation (PS) is a physics-inspired framework for intelligent agents which has also been applied to solve RL tasks in quantum physics [BD12; Mel&al17; MPK&al18; WMD&al20; Dal&al20; NDD&al19] and biology [RMB19]. This model can naturally be extended to a deep RL model [JTN&al21; Pre20] and has found applications in representation learning [Nau&al22; Eva&al22].
In this work, we propose a unified approach to optimizing the placement and parameter optimization of the gates in the circuit. We argue that such an approach is both relevant to traditional compilation of a more expressive, continuous gateset [PCM22], as well as to variational circuits where the experimental cost-function optimization may be simultaneously performed over discrete and continuous degrees of freedom. We use a RL agent, based on the PS framework, for the combinatorial optimization and a gradient-

based optimizer for the continuous optimization of the gate angles. In particular, we extend the method proposed in [Ost&al21] for the optimization of variational circuits in quantum chemistry and [SEL&al22] for state preparation to the case of unitary compilation. We consider the framework where an agent, that has control over an experimental platform, interacts with a black-box unitary process and attempts to simulate it. The task of the RL agent is to optimize the position of the gates on the circuit, whereas the gradient-based optimizer finds the optimal set of continuous parameters that minimize a given cost function. The reward function for the RL agent is constructed based on the results of the continuous optimization.
We test the learning algorithm first on standard unitaries, such as Toffoli gates, and then consider the task of quantum process simulation. The latter can be conceived as an experimental black-box unitary approximation strategy that allows the agent to reconstruct an unknown unitary process by compiling a proper quantum circuit.
Independently of our circuit compilation results, we propose a method to speed-up the simulation of quantum circuits based on an efficient representation of trapped-ion gates replacing standard matrix exponentiation. This method allows us to obtain analytic expressions for the ion-trap gates and their gradients and also to simulate the given gate set on other quantum computing platforms, such as superconducting quantum circuits [KTL&al21] or neutral atoms [Saf16].
The paper is organized as follows: In Section 5.2 we introduce the quantum circuit framework for trapped-ions. In Section 5.3.1 we present our method to compute fast analytic ion gates in simulation. In Section 5.3.2 we discuss continuous optimization methods and strategies to compute the gradients of the cost function both in simulation and on a real quantum device. In Section 5.3.3 we introduce PS and its extensions in the context of RL methods. In Section 5.4 we introduce the problem of circuit synthesis and our hybrid RL-continuous optimization method. In Section 5.5 we discuss the results of applying the proposed method to the compilation of (black-box) unitaries.

5.2 Problem setting

In this work, we consider a specific set of gates, normally implemented in trapped-ion quantum circuits, which is based on global Mølmer–Sørensen (MS) gates, equatorial rotations acting on the entire register of qubits, as well as local polar rotations. Trapped ions are among the most promising platforms for quantum computing hardware [BCM&al19]. They exhibit impressive coherence times even in absence of dynamical decoupling and spin echo techniques [HAB&al14] and have been shown to allow for high-fidelity quantum gates [BHL&al16; EWP&al19]. In a trapped-ion quantum computer, ions – usually are confined in a Paul trap [Pau90] using a varying electromagnetic field. The ions are addressed individually by a system of lasers aligned externally to the trap. The interaction of the laser field with the ion motional and electronic degrees of freedom allows for entangling operations. The laser pulses can be engineered to define the following

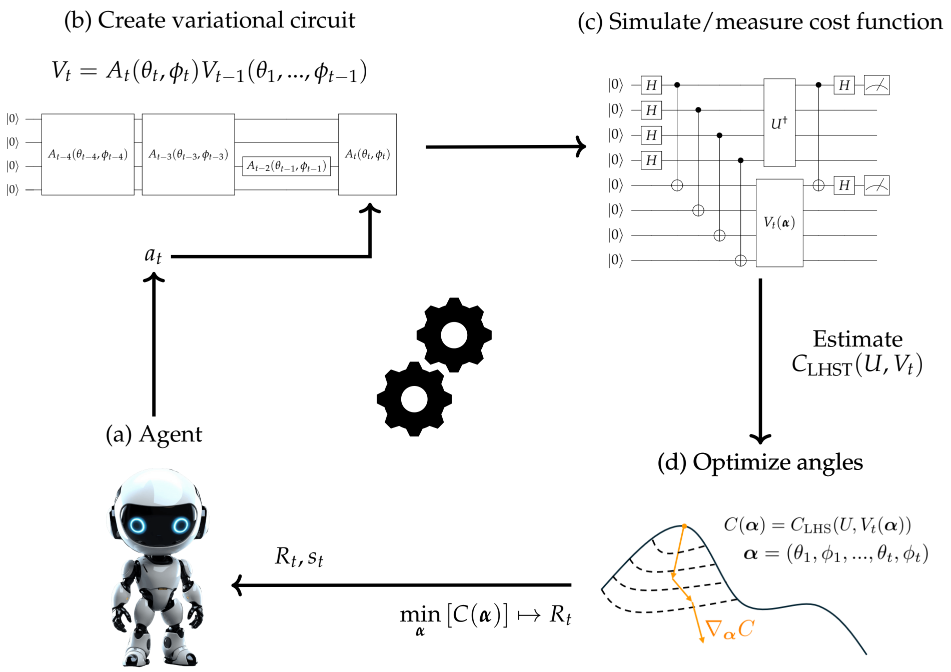

(b) Create variational circuit
$V_t = A_t(\theta_t, \phi_t) V_{t-1}(\theta_1, ..., \phi_{t-1})$
$|0\rangle$
$A_{t-4}(\theta_{t-4}, \phi_{t-4})$
$A_{t-3}(\theta_{t-3}, \phi_{t-3})$
$A_{t-2}(\theta_{t-1}, \phi_{t-1})$
$A_t(\theta_t, \phi_t)$
a_t
(c) Simulate/measure cost function
H
$U^\dagger$
$V_t(\boldsymbol{\alpha})$
Estimate $C_{\mathrm{LHST}}(U, V_t)$
(a) Agent
(d) Optimize angles
$C(\boldsymbol{\alpha}) = C_{\mathrm{LHS}}(U, V_t(\boldsymbol{\alpha}))$
$\boldsymbol{\alpha} = (\theta_1, \phi_1, ..., \theta_t, \phi_t)$
$\nabla_{\boldsymbol{\alpha}} C$
R_t, s_t
$\min_{\boldsymbol{\alpha}} [C(\boldsymbol{\alpha})] \mapsto R_t$

Figure 5.1: General scheme of the proposed hybrid training loop, where the RL agent (a) learns to optimize a (variational) quantum circuit. By choosing an action $a_t \in \{1, 2, ..., n+2\}$ the gate G_{a_t} is placed on the circuit (b), where $G_1 = \mathrm{MS}, G_2 = C_{xy}, G_3 = Z_1, ..., G_{n+2} = Z_n$ correspond to the gate set introduced in Eqs. (5.1)-(5.3). At each RL time step t, the circuit is used to compute a cost function $C(\boldsymbol{\alpha})$ based on the fidelity – see Eq. (5.4) – either through classical simulation or the Hilbert-Schmidt test (c) – see Eq. (5.13) – experimentally. The continuous optimizer (d) then outputs a guess for the optimal parameters $\boldsymbol{\alpha}^* = \mathrm{argmin}_{\boldsymbol{\alpha}} [C(\boldsymbol{\alpha})]$ that minimize the cost function for a specific circuit V_t. The minimal value of the cost function is used to assign a reward to the PS agent – see Eq.(5.29) –, thus closing the RL training loop.

universal set of quantum gates [MMN&al16; GMT&al12; MRR&al14]:

$$\mathrm{MS}(\theta, \phi) = \exp\left\{-i\frac{\theta}{4}(S_x \cos\phi + S_y \sin\phi)^2\right\} \tag{5.1}$$

$$C_{xy}(\theta, \phi) = \exp\left\{-i\frac{\theta}{2}(S_x \cos\phi + S_y \sin\phi)\right\} \tag{5.2}$$

$$Z_j(\theta) = \exp\left\{-i\frac{\theta}{2}\sigma_z^{(j)}\right\}, \tag{5.3}$$

where the first gate is a global MS gate [SM99; MS99], the second gate is a rotation of the entire register of qubits in the equatorial plane of the Bloch sphere, and the third gate represents a single-qubit, and therefore local, σ_z rotation acting on qubit j. The operators $S_x = \sum_{i=1}^{n} \sigma_x^{(i)}$, $S_y = \sum_{i=1}^{n} \sigma_y^{(i)}$, $S_z = \sum_{i=1}^{n} \sigma_z^{(i)}$ are given by the Pauli operators acting on qubit i. These gates can be easily generalized to the qudit case [RMP&al22]. For the optimization of unitaries, the gate overlap fidelity is a standard figure of merit [Nie02]:

$$F(U,V) = \frac{1}{d^2}\left|\mathrm{tr}\left\{VU^\dagger\right\}\right|^2, \tag{5.4}$$

where U and V are unitaries (one of them is the goal of the optimization) and d is the dimension of the Hilbert space – for n-qubit systems $d = 2^n$. Commonly, synthesising quantum circuits to reproduce an arbitrary unitary U requires having access to quantum computing hardware that implements a universal gate set and running the optimization of the continuous parameters.
If the circuit synthesis is performed on a classical computer, it is also necessary to simulate the gate set efficiently. Simulating and optimizing n-qubit collective gates such as those given in Eq. (5.1) and Eq. (5.2) is generally considered more challenging than with just two-qubit entangling gates and single qubit rotations [MMN&al16]. A standard strategy is to progressively increase the number of entangling MS gates on the circuit, accompanied by a suitable number of single and multi-qubit rotations, and to progressively optimize the gate parameters until an acceptable threshold of the figure of merit is reached. This is a viable strategy to obtain one solution for a quantum compilation problem on a trapped-ion device; it is however sub-optimal with respect to the number of gates required. More efficient solutions exist, but the optimization landscape may be difficult to navigate for various algorithms [DMS22]. In particular, as the optimization landscape with respect to the gate angles is particularly vast and depends on the arrangement of the gates on the circuit, it is often necessary to search through several combinations of gate sequences. Random or automated search can be implemented to reduce the number of gates present on the circuit by arbitrarily trying several configurations [Mor&al21; MMN&al16].

5.3 Methods

In the following section, we discuss our approach to address the three relevant aspects that characterize a circuit synthesis task: the efficient computer-aided simulation of the relevant trapped-ion gates, the optimization of the continuous gate parameters and the optimal arrangement of the gates on the circuit. In addition, we address the possible implementation of our hybrid RL-gradient based optimization method on real quantum devices.

5.3.1 Dynamics via fast exponentiation

In simulation, direct computation of the gates in Eqs. (5.1)-(5.2) is commonly done via direct matrix exponentiation of a Hamiltonian, which is generally slow for large matrices, since it requires several matrix multiplications, each one with a practical complexity of $O(d^{2.8})$ [Str69]– d is the matrix dimension –. Instead, here we show how to significantly reduce the per-iteration computational cost of the matrix exponentiation by factorizing it with respect to the rotation angle θ and to the phase angle ϕ. This is achieved via a spectral decomposition where the phase-independent part can then be cached before the optimization.

Mathematically, the factorization of the matrix exponential for a MS or C_{xy} gate can be written via spectral decomposition as

$$\begin{aligned} U_H(\theta,\phi) &= \exp\{iH(\theta,\phi)\} = \\ &= V(\theta,\phi)\left(\sum_{l=0}^{d-1} e^{-i\lambda_l(\theta,\phi)}\,|l\rangle\langle l|\right)V^\dagger(\theta,\phi), \end{aligned} \tag{5.5}$$

where $H(\theta,\phi)$ is a θ- and ϕ-dependent Hamiltonian – see exponents in Eqs. (5.1)-(5.2) –, V is the matrix of eigenvectors with respective eigenvalues λ_l.

Regrouping in terms of degenerate eigenvalues, and considering the case where the set of eigenvalues are ϕ-independent, i.e., $\lambda_l = \lambda_l(\theta)$, while a ϕ-dependent unitary is applied to the Hamiltonian, i.e., $H(\theta,\phi) = V(\phi)H(\theta)V^\dagger(\phi)$. As a consequence, the eigenvectors matrix $V = V(\phi)$ is independent of θ, and we can rewrite

$$U_H(\theta,\phi) = \sum_{k=0}^{n_\lambda} V(\phi)\left(e^{-i\lambda_k(\theta)}\sum_{l_k}|l_k\rangle\langle l_k|\right)V^\dagger(\phi) \tag{5.6}$$

$$= P(\phi)\odot\sum_{k=0}^{n_\lambda} e^{-i\lambda_k(\theta)}D_l, \tag{5.7}$$

where n_λ is the number of distinct eigenvalues, $|l_k\rangle$ are the respective eigenvectors, $D_k = V(0)\sum_{l_k}|l_k\rangle\langle l_k|V^\dagger(0)$ and $P(\phi)$ is a matrix of phase components. In our case, the columns of $P(\phi)$ are all equal and are given by the vector $\boldsymbol{p}(\phi) = \begin{pmatrix}1\\ e^{i\phi}\end{pmatrix}^{\otimes n}$, while $\odot$ represents element-wise (Hadamard) matrix multiplication. The C_{xy} and MS gates have few unique eigenvalues with $\lambda_k = (2k-n)\theta,\ 0\le k\le n_\lambda = n$ and $\lambda_k = (2k-n)^2\theta,\ 0\le k\le n_\lambda = \lceil n+\frac{1}{2}\rceil$, respectively, making the computation with cached D_k particularly efficient. A detailed derivation, together with a discussion of the computational speedup of this representation, is available in Appendix 5.8.1.

5.3.2 Continuous gradient-based optimization

We consider the optimization of an observable with respect to the continuous parameters. Although single problems can be optimized effectively by implementing an appropriate discretization method, in general this approach can lead to a sub-optimal solution, since it introduces discontinuities in the search space. Instead, we want to consider the dependency of the fidelity from continuous parameters and calculate its gradient.

5.3.2.1 Cost function based on known target unitaries.

We consider a quantum circuit composed of a sequence of continuous gates with angle parameters $\boldsymbol{\alpha} = (\boldsymbol{\alpha}_1, ..., \boldsymbol{\alpha}_L)$ and where each gate is composed of multiple rotation angles $\boldsymbol{\alpha}_1 \in \mathbb{R}^{M_1}, \ldots, \boldsymbol{\alpha}_L \in \mathbb{R}^{M_L}$ and $M_1, ..., M_L \geq 1$ and $M = \sum_{m=1}^{L} M_m$. The circuit is then given by

$$V(\boldsymbol{\alpha}) = \prod_{l=1}^{L} V_l(\boldsymbol{\alpha}_l), \tag{5.8}$$

where $V_l(\boldsymbol{\alpha}_l)$ is an arbitrary parametric unitary with parameters $\boldsymbol{\alpha}_l$.
The gradient with respect to a figure of merit, here the average gate fidelity – see Eq. (5.4) –, can also be directly computed and are given by

$$\nabla_{\boldsymbol{\alpha}_l} F = \frac{2}{d^2} \operatorname{Re}\Big\{\operatorname{tr}\Big(\nabla_{\boldsymbol{\alpha}_l} V(\boldsymbol{\alpha}) U^\dagger\Big) \operatorname{tr}\Big(V(\boldsymbol{\alpha})^\dagger U\Big)\Big\}, \tag{5.9}$$

with

$$\nabla_{\boldsymbol{\alpha}_l} V(\boldsymbol{\alpha}) = V_1(\boldsymbol{\alpha}_{l1}) \cdots \nabla_{\boldsymbol{\alpha}_l} V_l(\boldsymbol{\alpha}_l) \cdots V_L(\boldsymbol{\alpha}_L), \tag{5.10}$$

for $1 \leq l \leq L$. The naive element-wise computation of the gradient components uses $M \cdot L$ matrix multiplications, as the matrix in Eq. (5.9) needs to be evaluated M times and is given by the product of L unitaries. However, the gradient can be computed recursively, a method which is often referred to as GRAPE [KRK&al05], by storing the values of the intermediate unitaries $W_l = W_{l-1} V_l^\dagger(\boldsymbol{\alpha}_l), W_0 = I_d$ for $l = 1, ..., L$ in the product

$$\nabla_{\boldsymbol{\alpha}_l} F = \frac{2}{d^2} \operatorname{Re}\Big\{\operatorname{tr}\Big(W_{l-1} \nabla_{\boldsymbol{\alpha}_l} V_l(\boldsymbol{\alpha}_l) W_l^\dagger V U^\dagger\Big) \operatorname{tr}\Big(V^\dagger U\Big)\Big\}. \tag{5.11}$$

The gradient computation method given in Eq. (5.11) computes first the intermediate unitaries, for which we need L matrix multiplications and uses them to evaluate the gradient, which compared to Eq. (5.9) needs only $4M + L + 1$ matrix multiplications and therefore scales linearly with the number of gates and gate parameters in the circuit.
Eq. (5.11) allows us to compute the gradient of any cost function that resembles the structure of Eq. (5.4) efficiently in numerical simulations. This can be further sped up by similarly analytically computing the Hessian matrix of the cost function [DMJ&al20] or by using GPU or TPU architectures [MHB&al22]. Eq. (5.11) can also be useful in an experimental setting when the target dynamics, such as a desired state or unitary evolution, are known a priori.

5.3.2.2 Cost function based on black-box access to a target unitary.

In many NISQ-relevant algorithms, the structure of the quantum circuit need not be prescriptive. Instead, we optimize a general circuit ansatz variationally to minimize some performance cost function. In this setting, we once again may optimize both the discrete and continuous degrees of freedom of the circuit to minimize the cost function. One can conceive of situations where the target circuit unitary U may be given as a black-box process [CN97; BH96], that is where any decomposition of the circuit is unknown and we do not have a classical description of the unitary entries, or a circuit to compute them. In this situation, the cost function computing the distance between the black-box target U and a given parametric quantum circuit $V(\boldsymbol{\alpha})$ needs to be estimated.
A possible way to compute distances between a unitary implemented on a quantum computer and a black-box unitary implemented by a different quantum system is the Hilbert-Schmidt test (HST) [KLP&al19], a generalization of the SWAP test for state preparation, which evaluates the gate fidelity of Eq. (5.4) on a quantum circuit. A related cost function for black-box quantum compilation is given by the local Hilbert-Schmidt (LHS) test – see Fig. 5.1, panel (c) –, which has been shown to be easier to optimize and less sensitive to Barren plateaus [McC&al18; Cer&al21a]. Observe that the cost function $C_{\text{LHS}}(V,U)$ is bound from above and below by the average gate fidelity given in Eq. (5.4),

$$C_{\text{LHS}}(V,U) < C_{\text{HST}}(V,U) < nC_{\text{LHS}}(V,U), \tag{5.12}$$

which implies that minimizing $C_{\text{LHS}}(V,U)$ also minimizes $C_{\text{HST}}(V,U)$. Other cost functions have been proposed which are based on so-called incoherent learning, i.e., where the quantum computer and the black-box quantum system do not need to interact coherently [Jer&al23b].
For gradient-based optimization, we need to differentiate the cost functions C_{HST} or C_{LHS} with respect to continuous gate parameters. Unfortunately, the gradient method given in Eq. (5.11) requires access to the intermediate unitaries, which are generally not available in real-world scenarios. Moreover, we do not generally have direct access to the gradient of the unitary operations. As a consequence, we need to use the cost function itself to estimate its gradient with respect to the continuous parameters, which proves slower. In fact, computing the gradient in Eq. (5.9) with the method of finite differences requires $2M$ evaluations of the test circuit, where M is the total number of parameters. However, finite-difference methods applied to quantum circuits tend to have small signal-to-noise ratios and therefore require large numbers of shots [KE21], especially when compared to sampling strategies that estimate the gradient via trigonometric interpolation [WIW&al22; BWK22].

Let us now consider the specific case of trapped ions with the gate set introduced in Section 5.2. There are three different types of gates (Z, MS and C_{xy}), with four types of parameters shifts: the three rotation angles of the Z gate, the MS gate and the C_{xy} gate, respectively, and the phase term ϕ of the MS and C_{xy} gates, which is the same for both unitaries (see also Appendix 5.8.2).

In the following, we consider a cost function on the Hilbert Schmidt test, but the results can be applied to the local test as well. In the case of the Z gate, only one parameter θ is present, whereas for the other two gates we have two possible parameter-shifts. The experimental quantum cost function C_{HST} comparing the parameterized circuit $V(\boldsymbol{\alpha})$ in Eq. (5.8) with a target unitary U is given by

$$C_{HST}(\boldsymbol{\alpha}) = 1 - \mathrm{Tr}\Big\{H(U \otimes V(\boldsymbol{\alpha})^*)\rho(U^\dagger \otimes V(\boldsymbol{\alpha})^T)\Big\}, \tag{5.13}$$

where ρ and H are projectors defined in Appendix 5.8.2 and Ref. [KLP&al19]. We consider here only the shift with respect to one gate at a time, which we name $V_1(\boldsymbol{\alpha}_1)$, while the remaining gates in the circuit, i.e., $V_2(\boldsymbol{\alpha}_2), ..., V_L(\boldsymbol{\alpha}_L)$, are fixed. This procedure can be repeated for each gate parameter. We first consider the case where $V_1(\boldsymbol{\alpha}_1 = \theta) = Z(\theta)$. The exact derivative of $C_{\mathrm{HST}}(\theta)$ is given by:

$$\frac{\partial}{\partial\theta} C_{\mathrm{HST}}(\theta, Z) = C_{\mathrm{HST}}(\theta + \frac{\pi}{2}, Z) - C_{\mathrm{HST}}(\theta - \frac{\pi}{2}, Z) \tag{5.14}$$

For $V_1(\boldsymbol{\alpha}_1 = (\theta, \phi)^T) = C_{xy}(\theta, \phi)$ and following Ref. [WIW&al22], the parameter derivative with respect to θ is given by

$$\frac{\partial}{\partial\theta} C_{\mathrm{HST}}(\theta, \phi, C_{xy})\Big|_{\theta=0} = \tag{5.15}$$
$$\sum_{l=1}^{2n} \frac{(-1)^{l-1}}{2\sin\left(\frac{2l-1}{2n}\pi\right)} C_{\mathrm{HST}}\left(\theta + \frac{2l-1}{2n}\pi, \phi, C_{xy}\right),$$

whereas for $V_1(\boldsymbol{\alpha}_1 = (\theta, \phi)^T) = \mathrm{MS}(\theta, \phi)$, the parameter-shift rule with respect to θ becomes,

$$\frac{\partial}{\partial\theta} C_{\mathrm{HST}}(\theta, \phi, \mathrm{MS})\Big|_{\theta=0} = \tag{5.16}$$
$$\sum_{l=1}^{2\lceil\frac{n}{2}\rceil} \frac{(-1)^{l-1}}{2\sin\left(\frac{2l-1}{\lceil\frac{n}{2}+1\rceil}\pi\right)} C_{\mathrm{HST}}\left(\theta + \frac{2l-1}{\lceil\frac{n}{2}+1\rceil}\pi, \phi, \mathrm{MS}\right)\delta_i^{\mathrm{floor}}.$$

A more simplified rule can be derived for the parameter ϕ, which is valid for both the C_{xy} and the MS gate

$$\frac{\partial}{\partial\phi} C_{\mathrm{HST}}(\theta, \phi) = C_{\mathrm{HST}}(\theta, \phi + \frac{\pi}{4}) - C_{\mathrm{HST}}(\theta, \phi - \frac{\pi}{4}) + \tag{5.17}$$
$$+ \Big(\frac{1}{\sqrt{2}} - \frac{1}{2}\Big) C_{\mathrm{HST}}(\theta, \phi - \frac{\pi}{2}) + \Big(\frac{1}{\sqrt{2}} + \frac{1}{2}\Big) C_{\mathrm{HST}}(\theta, \phi + \frac{\pi}{2}).$$

Using the expressions given by Eq. (5.14) for the Z gate, by Eq. (5.15) and Eq. (5.17) for the C_{xy} gate and Eq. (5.16) and Eq. (5.17) for the MS gate, we can compute any gradient of cost functions estimated in experiments on quantum devices that use cost functions such as Eq. (5.13). For more details about the parameter-shift rules of the C_{xy} and MS gates, and further possible simplifications, see Appendix 5.8.2.

5.3.3 Combinatorial Optimization

In this section, we consider the problem of determining an optimal arrangement of the gates on the quantum circuit. This problem arises from the necessity of minimizing the depth of a quantum circuit to reduce the total circuit execution time, thereby reducing decoherence. And while the circuit can be compiled in layers [OGB21], it is difficult to determine the optimal size due to the large number of possible optimal parameter configurations that produce the same circuit. Therefore, it is necessary to search through various combinations of gate arrangements to determine a minimal one. This task, which partially falls in the realm of combinatorial optimization [MSI&al20], is particularly suitable for reinforcement learning algorithms [Mor&al21].

5.3.3.1 Reinforcement Learning with Projective Simulation

Reinforcement learning describes a class of algorithms which use an agent-environment interaction model to maximize a reward function. In particular, the agent can be represented by a parametrized model, called policy, that outputs action signals on the environment upon receiving observations as inputs. For each action or sequence thereof, the environment returns observations and reward signals. In gradient-based methods, the policy parameters are updated in the direction that maximizes the discounted future expected return [SB18]. Based upon the different tasks considered, a vast range of algorithms and methods have been developed to tackle various environments [Li17].
In the following section, we consider the PS architecture – see Fig. 5.2 (a). Projective Simulations (PS) is a framework for agency and decision making that has also found applications as a RL agent [Boy&al20; MPK&al18]. In that context, a PS agent interacts with an environment by performing actions sampled from an action space A, whereas the environment provides the agent with perceptual inputs, which reside in a percept space S, and reward signals. Its central feature is represented by a so-called episodic and compositional memory (ECM) – see Fig. 5.2 (a) –, a graphical model consisting of a network, or weighted graph, where the vertices are clips and represent, e.g., remembered percepts or remembered actions, but also more general states of the agent's memory. In this framework, the agent creates a clip inside the ECM each time it receives a previously unknown input from the environment, or each time it creates a new action clip or a more abstract clip, e.g., through action composition. This makes it is possible to create ECM networks with complex graph structures, allowing for generalization capabilities [Mel&al17; Eva&al22]. Each input triggers a random walk between the nodes of the ECM that is governed by the edge weights of the graph. We assume in the following that the ECMs created are two-layered networks, with one layer representing percepts clips and the other action clips and where a percept at time t of the RL interaction is

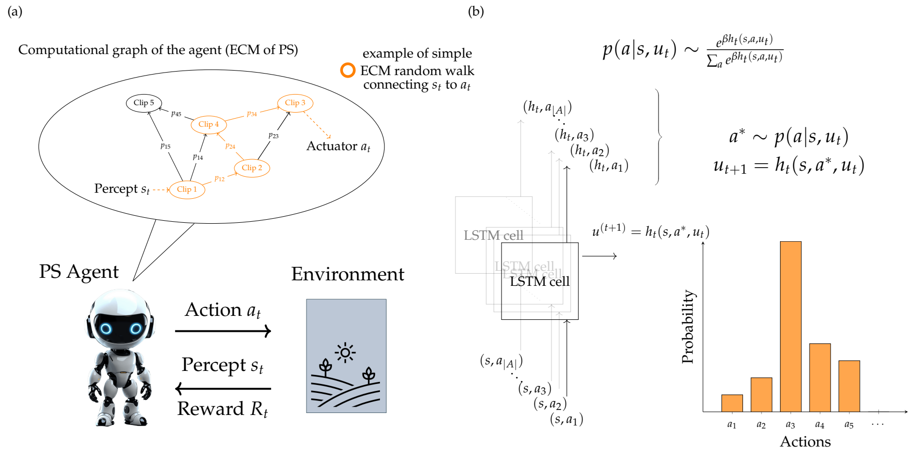

(a)
Computational graph of the agent (ECM of PS)
example of simple ECM random walk connecting s_t to a_t
Clip 1
Clip 2
Clip 3
Clip 4
Clip 5
Percept s_t
Actuator a_t
PS Agent
Environment
Action a_t
Percept s_t
Reward R_t
(b)
$p(a|s,u_t) \sim \frac{e^{\beta h_t(s,a,u_t)}}{\sum_a e^{\beta h_t(s,a,u_t)}}$
$a^* \sim p(a|s,u_t)$
$u_{t+1} = h_t(s,a^*,u_t)$
$u^{(t+1)} = h_t(s,a^*,u_t)$
LSTM cell
Probability
Actions

Figure 5.2: (a) Schematic representation of a PS-environment interaction: the agent is equipped with a memory structure that allows it to process the information input by the environment (in the case of PS, the episodic and compositional memory, ECM [BD12; Mel&al17]). Every perceptual input from the environment triggers a sequence of transitions – showed in orange – within the internal computational graph of the agent and governed by transition probabilities p_{ij}. The sequence of transitions connects a percept clip s_t, which in our case corresponds to the sequence of actions $a_1, \dots, a_{t-1}$ used to generate the circuit V_t using the corresponding gates, to an action clip corresponding to a_t. (b) Policy parametrization of the PS-LSTM algorithm. In this implementation, the agent receives a sequential input at time t and constructs a policy by outputting weights $h_t(s,a,u_t)$ corresponding to each action a, the current percept s and the hidden state of the LSTM network u_t. Afterward, an action a^* is sampled from the policy constructed this way and the weight corresponding to this action is propagated further in the hidden state of the LSTM network, as given in Eq. (5.23). The weights are then reset when the episode terminates.

connected directly to all action clips. The edge weights are initialized uniformly:

$$\forall a \in A, \forall s \in S : h_0(s,a) = 1. \tag{5.18}$$

We define the probability distribution[1] over percepts $s \in S$ and actions $a \in A$ as

$$p(a|s) = \frac{e^{\beta h_t(s,a)}}{\sum_a e^{\beta h_t(s,a)}}, \tag{5.19}$$

[1]In standard PS, the probability is defined without the exponential factors.

for a edge weight $h_t(s,a)$ connecting s to a. The PS algorithm optimizes the policy, similar to other RL frameworks, through a reward signal, which is provided by the environment to the agent. At each RL time-step t, the update rule is given by

$$\begin{aligned} h_{t+1}(s,a) = h_t(s,a) - \gamma(h_t(s,a) - 1) + \\ + g_t(s,a) R_t, \end{aligned} \tag{5.20}$$

where R_t is the reward at time step t, γ is a damping coefficient that regularizes the result and reduces potential instances of trapping in local minima, $g_t(s,a)$ are so-called glow values [Mau&al15], which are initially set to zero

$$\forall a \in A, \forall s \in S : g_0(s,a) = 0. \tag{5.21}$$

They are set (or reset) to $g_t(s,a) = 1$ at time t if the corresponding edge is traversed and decay in value for each time step where they are not used according to the rule

$$g_{t+1}(s,a) = (1-\eta) g_t(s,a), \tag{5.22}$$

where $0 \leq \eta \leq 1$. The glow mechanism helps distribute the rewards along the entire chain of state-action transitions traversed by the agent in an episode. We see that the edge weights that are rewarded positively grow, thereby enhancing the probability that the same action is chosen again in the future upon receiving a similar percept.
PS has been successfully extended to include more powerful computational structures, such as deep feedforward energy-based networks [JTN&al21; Nau&al22] and recurrent networks [Pre20]. The aforementioned architectures enable the PS agent to update the policy using function approximators. This allows the agent, as it is the case for deep Q-learning [Mni&al15], to construct more powerful representations of the percept- and action-spaces and achieve high performances in environments with, e.g., continuous parametric percepts and actions without the need of discretization strategies [SB18]. We focus here on the Long-Short Memory Network (LSTM) architecture [Hoc91; HS97], a recurrent neural network that is equipped with an internal memory architecture that helps it learn long-range correlations in a sequential input. In this case, the action sampled at time t also depends on the LSTM internal state u_t, i.e.,

$$a^* \sim p(a|s,u_t) = \frac{e^{\beta h_t(s,a,u_t)}}{\sum\limits_a e^{\beta h_t(s,a,u_t)}} \tag{5.23}$$

with $s \in S$ and $a, a^* \in A$ and the u-value is updated w.r.t. the sampled action as follows:

$$u_{t+1} = h_t(s, a^*, u_t). \tag{5.24}$$

The term $u_t = u_t(s,a)$ also depends on percepts and actions, but as we see in Eq. (5.24), only the u-value corresponding to the action sampled by the agent at time t are propagated as information to the next RL time step, as shown in Fig. 5.2 (b). This value is the one that carries relevant information about the correlations in the sequential structure of the percepts. We would like to stress that the presence of an additional non-linear term in the update rule in Eq. (5.23) induces a different type of ECM structure, where the term $u_t(s,a)$ represents an additional edge weight. The update of the reward mechanism can be generalized starting from the standard PS reward update mechanism to fit the training of neural network-based policies, in analogy with the case of Q-learning [JTN&al21; Pre20].

5.4 Circuit compilation

Quantum compilation is the general task of reproducing a general operation $\mathcal{M} \in \mathbb{C}^{4^n \times 4^n}$ on a n-qubit quantum processor. In particular, we consider the compilation of unitary operations $U \in U(n)$.

5.4.0.1 Layer-based compilation and heuristic search

A general approach for gate synthesis in trapped-ion circuits is discussed in Ref. [MMN&al16]. This approach is based on the universality of two-qubit entangling gates for quantum computation [DiV95]. As a consequence, it is possible to compile a quantum algorithm in growing layers, i.e., by iteratively placing entangling MS gates on the circuit followed in each case by a collective rotation of the following type

$$\begin{aligned}
&\mathcal{R}(\theta_k, \phi_k, \ldots, \theta_{k+n+1}, \phi_{k+1}) = \\
&= C_{xy}(\theta_k, \phi_k) \prod_{i=1}^{n} Z_i(\theta_{k+i}) C_{xy}(\theta_{k+n+1}, \phi_{k+1}),
\end{aligned} \tag{5.25}$$

where the gates C_{xy} and Z are defined in Eqs. (5.2)-(5.3). For each layer of gates present on the circuit, we have one MS gate and one general rotation $\mathcal{R}$ – see Eq. (5.25) –, which consists of n local Z gates and two C_{xy} gates.

Algorithm 3 Exhaustive search with layer-based compilation

```
    Input Target unitary $U$, threshold $\epsilon$, cost function $C$.
    Output $V^*, \boldsymbol{\alpha}^*$.
 1: ▷ Construct layers:
 2: for $L = 1$ to $L_{\text{MS}}$ do
 3:     ▷ Add MS gate and rotations:
 4:     $V(\boldsymbol{\alpha}) = \prod_{l=1}^{L} V_l(\boldsymbol{\alpha}_l)$
 5:     ▷ All angles but the ones of MS gates:
 6:     $\boldsymbol{\alpha} = (\theta_1, \phi_1, ..., \theta_{(n+2)L}, \phi_{2L})^T$
 7:     $\tilde{K} = (n+2)(L+1)$
 8:     ▷ Loop over numbers of rotations:
 9:     for $k = 1$ to $\tilde{K}$ do
10:         ▷ Loop over combinations of indices:
11:         for $j = 1$ to $\binom{\tilde{K}}{k}$ do
12:             ▷ Set angles with chosen indices to zero:
13:             $\alpha_{\sigma(j,k)} = 0$
14:             $\tilde{\boldsymbol{\alpha}}^k = (..., \alpha_{\sigma(j,k)} = 0..., \alpha_{\sigma(j,k)} = 0, ...)^T$
15:             ▷ Cost function according to Eq. (5.27):
16:             $C(\tilde{\boldsymbol{\alpha}}^k) = 1 - F(V, U)$
17:             $C^* = \min(C(\tilde{\boldsymbol{\alpha}}^k), \nabla_{\boldsymbol{\alpha}^k} C(\tilde{\boldsymbol{\alpha}}^k))$
18:             if $C^* \leq \epsilon$ then
19:                 $\boldsymbol{\alpha}^* = (\tilde{\boldsymbol{\alpha}}^k)^*$
20:                 $V^* = V$
21:                 break
22:             else
23:                 continue
24:             end if
25:         end for
26:     end for
27: end for
```

The unitary added to the circuit at each layer l is:

$$V_l(\boldsymbol{\alpha_l}) = \text{MS}(\theta_l^{\text{MS}}, \phi_l^{\text{MS}})\ \mathcal{R}(\theta_l, \phi_l, ..., \theta_{l+n+1}, \phi_{l+1}). \tag{5.26}$$

The cost function

$$C(\boldsymbol{\alpha}) = 1 - \frac{1}{d^2}\left|\mathrm{tr}\left\{\prod_{l=1}^{L} V(\boldsymbol{\alpha}_l)U^\dagger\right\}\right|^2, \tag{5.27}$$

based on the gate fidelity in Eq. (5.4), has to be minimized with respect to the angle parameters $\boldsymbol{\alpha}$. After obtaining the optimal parameters $\boldsymbol{\alpha}^*$, if the desired error threshold is reached, the algorithm terminates, otherwise a new layer of gates is placed on the circuit, up to a maximum number of layers L_{MS}. By running this procedure iteratively with different angle parameter sets, we can search through the optimization landscape to find different gate decompositions (see Algorithm 3 and Ref. [MMN&al16]).

Algorithm 4 PS-based compilation

Input Target unitary U, Action set $G_1 = \mathrm{MS}, G_2 = C_{xy}, G_3 = Z_1, \ldots, G_{n+2} = Z_n$, threshold function ϵ_t, cost function C

Output $V^*, \boldsymbol{\alpha}^*$

```
 1: for $e = 1$ to $E_{\max}$ do
 2:     $s_1 = (0, \ldots, 0), V_1 = I_d$
 3:     for $t = 1$ to $L_{\max}$ do
 4:         ▷ Sample action according to Eq. (5.19):
 5:         $a_t \sim \pi(a|s_t)$
 6:         $V_{t+1}(\boldsymbol{\alpha}_t) = G_{a_t} V_t(\boldsymbol{\alpha}_{t-1})$
 7:         $s_{t+1} = (a_1, \ldots, a_t, 0, \ldots, 0)$
 8:         $\boldsymbol{\alpha} = (\boldsymbol{\alpha}_1, \ldots, \boldsymbol{\alpha}_t) \sim 2\pi \cdot \mathcal{N}(\mathbf{0}_{\boldsymbol{\alpha}}, \boldsymbol{I}_{\boldsymbol{\alpha}})$
 9:         ▷ According to Eq. (5.27):
10:         $\boldsymbol{\alpha}^* = \mathrm{argmin}(C(\boldsymbol{\alpha}), \nabla_{\boldsymbol{\alpha}} C(\boldsymbol{\alpha}))$
11:         $R_t = \begin{cases} 2 & \text{if } \epsilon_{\min} \le C(\boldsymbol{\alpha}^*) \le \epsilon_t \\ 10 & \text{if } C(\boldsymbol{\alpha}^*) \le \epsilon_{\min} \\ 0 & \text{otherwise} \end{cases}$
12:         if $C^* \le \epsilon_t$ then
13:             $V_* = V_{t+1}$
14:             break
15:         end if
16:         Update rule for all $h_{t+1}(s_t, a_t)$ (see Eq. (5.20))
17:     end for
18:     Update threshold $\epsilon_t$ according to Eq. (5.30)
19: end for
```

However, while this method can be applied to circuits with a small number of layers and qubits, its use becomes impractical as these two parameters grow. In fact, the number of total combinations to analyze for a given number of MS layers L_{MS} and a number of qubits n is

$$N_{\text{combinations}} = \frac{2^{(n+2)(L_{\mathrm{MS}}+1)} - 1}{1 - \frac{1}{4}^{(n+2)}} - L_{\mathrm{MS}}^2. \tag{5.28}$$

For a 3-qubit circuit with L layers of entangling gates, the number of possible circuits is $N_{\text{combinations}} = 1049591$. For a 4-qubit circuit with 5 entangling gates the number of combinations is approximately $N_{\text{combinations}} \sim 2^{36}$. We see that already for 4-qubit gates, a brute force approach is unfeasible. Moreover, due to the large search space, even approaches based on random search are not a viable option, especially if the number of entangling gates is large.
In the following section, we suggest a different approach to the optimization scheme where the position of the discrete parameters is modified by a RL agent equipped with a curriculum scheme, whereas the continuous parameters are optimized at each iteration and using a pre-defined heuristic for angle initialization. This can offer benefits in several situations where the number of gates is large enough to make the use of automated search prohibitive but not so large to make the problem completely intractable from the point of view of continuous parameter optimization.

5.4.0.2 Reinforcement learning-based compilation

In the following, we discuss the implementation of RL-based compilation. In this system, the agent acts on the environment, which represents a quantum circuit, by choosing a gate from, e.g., the gate set of Section 5.2 and placing it on the circuit. As an observation, the agent receives information about the environment internal state. This can vary depending on the task to be solved: in Ref. [Mor&al21], e.g., the input is a single-qubit unitary and the task is to construct a pre-trained optimal compiler that constructs any unitary with minimal average number of actions. In Refs. [Pre20; BDS&al18], the agent receives an encoded representation of the quantum circuit in terms of gates and qubits. The circuit-based input has the advantage of scaling linearly with the number of qubits for n-qubit entangling gates and its sequential structure makes it suitable for recurrent or autoregressive policies [YLB21; SEL&al22]. The perceptual input of the PS-LSTM agent is given by the current gate on the circuit. The action of the PS-LSTM agent is placing one further gate on the circuit. A representation of the interaction between the agent and the quantum circuit environment is given in Fig. 5.1: at each time step, the agent – Fig. 5.1 (a) – can place one or more gates on the circuit. Then the circuit – Fig. 5.1 (b) – is mapped to the corresponding unitary function, which depends on parameters $\boldsymbol{\alpha}$. The gradient-based optimizer – Fig. 5.1 (d) –, i.e., the L-BFGS-B algorithm [LN89], is given the task of finding the optimal set of parameters to maximize the fidelity – Fig. 5.1 (c) – between the current circuit and a target unitary process. The reward function and the percept for the next interaction step are constructed based on the result of the

optimization. The algorithm is shown in Algorithm 4 for the standard PS method and can be easily generalized to a framework with deep networks [JTN&al21; Pre20].
Furthermore, one may design different types of RL environments based on the way the agent places gates on the circuit. We develop here two different approaches for two different types of RL environments: In the first environment, we just mimic the structure of layer-based compilation, in which we keep the entangling gates fixed and then let the agent vary the rotations between them. This assumes that the RL interaction is defined by the number of gates placed between each entangling layer and the total amount of entangling layers considered. This architecture has the advantage of restricting the search space of the RL agent, but it allows for less exploration of the cost function landscape.
The second architecture allows the agent to place any available gate, entangling or non-entangling, on the circuit and as such does not restrict the action of the agent on the quantum circuit, with one single exception: if the agent places the same type of gate twice in a row on the circuit, these two gates are merged together in one single gate. This is needed to avoid the agent getting stuck in a loop where it keeps performing the same operation over and over again, without any meaningful exploration of the optimization landscape from the perspective of the continuous optimizer. We employ this architecture in our simulations.
In our implementation, the action a_t chosen at time t is the index of certain gate in the gate set, whereas the corresponding percept s_t is given by the concatenation of previously chosen actions $(a_1, ..., a_{t-1})$. Thus, the action space is given by the gate indices $A = \{1, 2, ..., n+2\}$ and the percept space by the Cartesian product of $L_{\max}$ action spaces: $S = A^{\times L_{\max}}$. The PS-LSTM algorithm, however, can also be given just the action at time $t-1$ as percept, since the LSTM memory can automatically capture the correlations between the elements in the sequence. Here, we adopt a RL training procedure with a curriculum scheme as described in Ref. [Ost&al21]. This allows the agent to sufficiently explore the solution space and gradually adapt the solution. More specifically, for each task of quantum circuit optimization, we define a curriculum strategy where we reward the agent at time step t each time it finds a sequence with achieved minimal infidelity $C(\boldsymbol{\alpha}^*)$ lower than a chosen moving threshold ϵ_t and a fixed threshold $\epsilon_{\min} = 10^{-2}$:

$$R_t = \begin{cases} 2 & \text{if } \epsilon_{\min} < C(\boldsymbol{\alpha}^*) < \epsilon_t \\ 10 & \text{if } C(\boldsymbol{\alpha}^*) < \epsilon_{\min} \\ 0 & \text{otherwise} \end{cases} . \tag{5.29}$$

The episode terminates when the reward the cost function minimum in a given time step falls below the threshold ϵ_t or when the maximal length of the circuit per episode, $L_{\max}$, is reached. The RL training terminates upon reaching the maximum number of episodes $E_{\max}$. The reward scheme helps to progressively increase the fidelity throughout training without allowing for too long circuits.
The threshold is then lowered as episodes progress based on previous rewards obtained by the agent. In our implementation, we lower the threshold when it has been surpassed

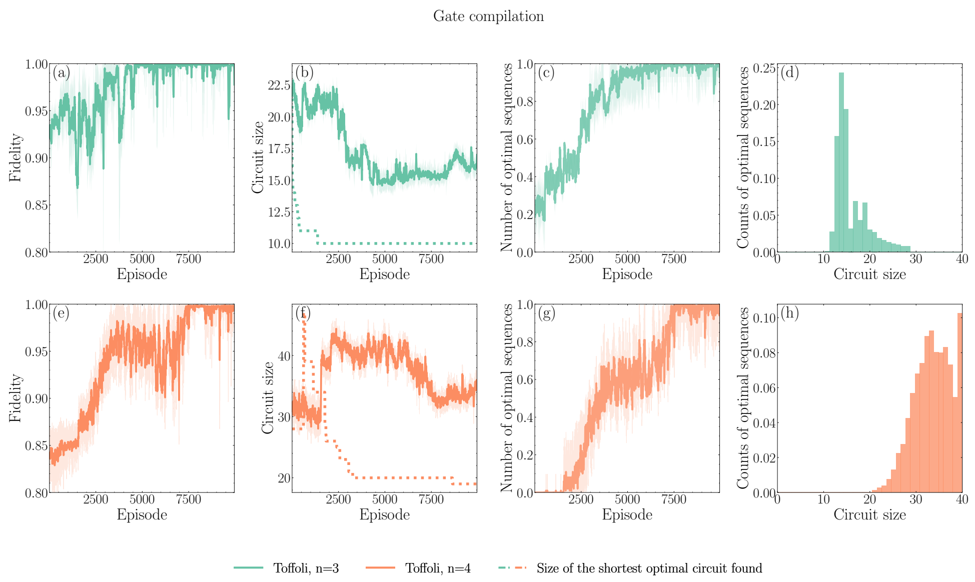


Figure 5.3: Quantum circuit optimization of different gates using the gate set defined in Eqs. (5.1)-(5.3). (a), (e) show the average reward; (b), (f) the average circuit size (thick line) and the size of the best circuit found so far by all agents (dotted line); (c), (g) the number of optimal sequences per episode; (d), (h) are histograms which show the distribution of the optimal gate sequences over the circuit size. All lines refer to simulations of the 3-qubit Toffoli gate, first compiled on a 3-qubit (green line, average over 10 agents and 20 episodes) and then on a 4-qubit circuit (orange line, average over 5 agents and 20 episodes). Due to the n-qubit interaction of the trapped-ion gates, the optimal sequence that generates the gate – which we define as the shortest sequence whose infidelity falls below $\epsilon_{\text{min}} = 10^{-2}$ – increases in size from 10 to 19 gates. Shaded regions in the plots represent the corresponding standard deviations. While the average fidelity increases to reach the maximum for both circuits, we see that the average circuit length appears to be higher than the shortest circuit length. There are possible explanations for this: the shortest sequences can be harder to optimize, so there is a higher chance that the optimizer fails at outputting the cost function minimum for a given circuit structure and the curriculum scheme, that modifies the problem online during the training. Overall, we observe that the size of the optimal circuit decreases as training progresses, thus validating the successful optimization of the policy.

by the agent at least 500 times using the following scheme:

$$\epsilon_{t+1} = \begin{cases} \epsilon_{\text{min}} + \frac{1}{2}(\epsilon_{\text{min}} - \epsilon_t) & \text{if } \epsilon_{\text{min}} \leq \epsilon_t \leq 1 \\ \epsilon_{\text{min}} & \text{if } \epsilon_t \leq \epsilon_{\text{min}} \\ 1 & \text{otherwise.} \end{cases} \tag{5.30}$$

5.5 Results

We test our algorithm on two relevant tasks of circuit optimization: Standard gate compilation, which can be useful in particular for experimental applications of frequently used gates (Toffoli, etc.) and the simulation of black-box unitary processes with quantum circuits. The latter framework is particularly interesting for quantum simulation and offers the possibility of implementing our algorithm directly on an experimental setting where a black-box unitary process has to be simulated by a quantum circuit with available gates.

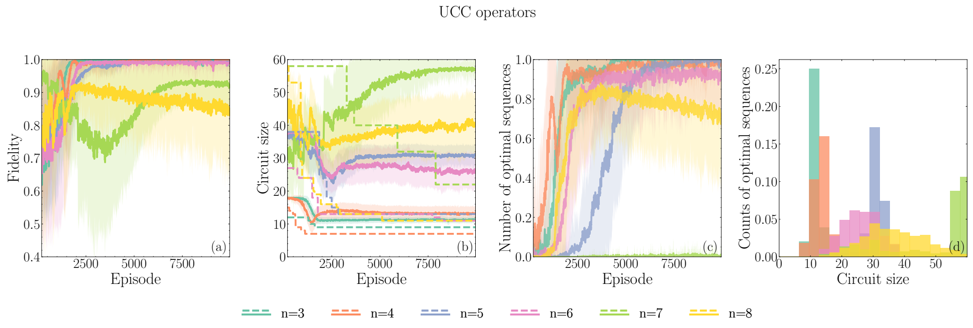


Figure 5.4: (a) Fidelity; (b) circuit size (thick lines) and size of the best circuit found by all agents (dotted lines); (c) number of optimal sequences per episode and (d) histogram of the optimal sequences – i.e., whose infidelity falls below $\epsilon_{\text{min}} = 10^{-2}$ – based on the circuit size for the UCC operators defined in Eq. (5.31) for $\beta = \frac{\pi}{2}$ and for 3, 4, 5, 6, 7, 8 qubits. The learning curves show the average over 20 agents, after which an average over a time window of 20 episodes is performed. We observe how the average maximal fidelity reached by the agents decreases as a function of the number of qubits, whereas the optimal circuit size increases, which means that a longer circuit is necessary to synthesize the desired operator. Moreover, the fraction of optimal sequences found by the agent is also decreasing for higher numbers of qubits, as optimal policies become more and more sparse and thus harder to discover for the PS-LSTM agent. The hardest task for the PS-LSTM agent proves to be 7-qubit UCC operator, where the learning curve of the agents fails to converge, while, e.g., for the 8-qubit UCC unitary the agent can find sequences with very high fidelities ($1 - F < 10^{-5}$), even though the convergence is sub-optimal. We also observe that for each learning curve there is a dip at some point in the training: this is most likely due to the curriculum scheme, which modifies the reward threshold during training and therefore influences the size of the optimal circuits found by the agent. However, when considering the ensemble of simulations, we see that the agents are capable of finding shorter and shorter optimal circuits. Shaded regions in the plots represent the corresponding standard deviations.

5.5.1 Example 1: gate compilation

As a first application of the method presented above, we consider the problem of compiling a gate on an ion-trap quantum processor using the gate set given in Eqs. (5.1)-(5.3). For this class of tasks, we choose two gate compilation problems which can be of interest for typical applications, before passing to a more general framework which considers black-box unitaries. First, we consider a standard quantum computing gate, the 3-qubit Toffoli gate [Tof80; MKH&al09] – see Fig. 5.3 – for a 3-qubit (green line, upper four plots) and a 4-qubit circuit (orange line, lower four plots). The dashed lines in Fig. 5.3 show the optimal solutions found by the agent. In the 3-qubit case, this solution matches agrees with the results given in Ref. [MMN&al16]. Fig. 5.3 (a), (e) show the fidelity of the sequences produced and optimized by the agents as the learning progresses, (b), (f) show the average circuit size and the size of the shortest circuit found by the agent, (c), (g) show the average number of optimal sequences, i.e., with fidelity higher than 0.99, found in each episode (which in the best-case scenario should converge to one per episode) and (d), (h) show histograms representing the number of sequences for different bins of circuit size. We observe from the first and second plot in each row of Fig. 5.3 that the agents are capable of maximizing the fidelity and at the same time of reducing the average circuit size in both cases. Moreover, while the average circuit size is larger than the size of the best circuit, most likely due to the presence of the curriculum and the influence of local minima on the angle optimization, we also see that the agents find shorter and shorter optimal circuits as the training progresses, hinting that the at least one agent in the ensemble is indeed learning to optimize the circuit correctly. Moreover, the third plot in each row shows us that the agents find an increasing number of high-fidelity sequences during training. The minimal sequences found by the agents are located in the leftmost tail of the distribution.
The 3-qubit Toffoli gate implemented on a 4-qubit circuit could have in principle a relatively long generating quantum circuit in the chosen gate set, since the 4-qubit gate set also affects the qubits left unchanged by the Toffoli gate. We also observe, in our simulations, that sequences generating this gate are sparse in the action space, which could make it generally difficult to produce this gate on a register of n qubits without the help of MS and C_{xy} gates acting only on subspaces of the register. We test whether our method can discover a circuit of reasonable size.
For large numbers of qubits and large circuit sizes, we observe that although the algorithm can find shorter circuits, it is still impaired by the computational overhead of simulating such circuits exactly. In this case, the gate decomposition method introduced in Section 5.3.1 provides a useful speedup for RL simulations, especially if compiled on GPU architectures.

5.5.2 Example 2: General quantum process simulation of a Hamiltonian model

As a second example, we compile parametric-type operators that play a relevant role in quantum chemistry [TB06; BM07], i.e., UCC-type operators. These are part of a more general class of many-body operators [MKW17]:

$$U(\beta) = \exp\left(i\beta \left(\prod_{i=1}^{n} (\sigma_x^{(i)} - i\sigma_y^{(i)}) + \text{h.c.} \right) \right). \tag{5.31}$$

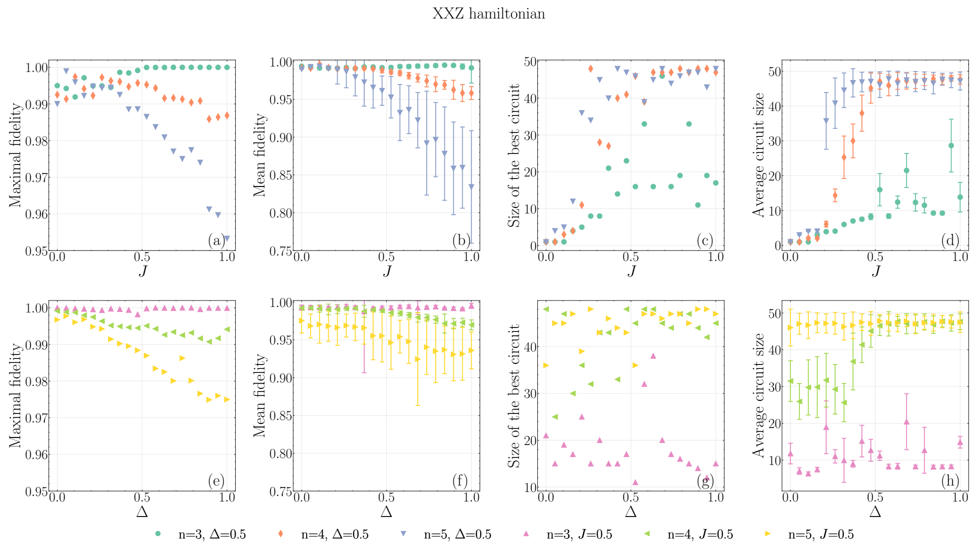


Figure 5.5: These plots summarize the result of simulating the XXZ-model unitary with our approach for different parameter configurations for $n = 3, 4, 5$ qubits (3 agents for each parameter sample). (a) shows the maximal value of the fidelity found by the agents, (b) the mean fidelity, (c) the size of the shortest circuit associated with the fidelity in (a) and (d) the average circuit size, as a function of the coupling J of the XXZ model for two different configuration of the transverse coupling $\Delta = 0.5$ with varying J and and $J = 0.5$ with varying Δ (dark green, orange and blue and pink, light green and yellow, respectively). The evolution time was fixed to $\tau = 0.25$ and the maximum size of the circuit to 50. The average fidelity is calculated over the last 200 episodes. Vertical bars show the standard error for the mean values of fidelity and circuit size. We see here that the circuit size increases dramatically as the parameters Δ and J increase. We also see from the number of outliers in (c) and (g), that for certain values of the parameters it is hard for the agents to exactly retrieve the optimal circuit.

The class of operators defined by Eq. (5.31) resembles the collective rotations used in trapped-ion gate sets given in Eqs. (5.1)-(5.3), they are however sparser. Due to their importance, they represent our first candidate for process simulation on the quantum circuit. The results for the simulation using PS-LSTM of several such operators – $n = 3, 4, 5, 6, 7, 8$ – and for $\beta = \frac{\pi}{2}$ is given in Fig. 5.4. We observe that the agent is able to increase the fidelity up to the optimal value. We also see that the size of the optimal

circuit generating the corresponding operators increases with the number of qubits. Furthermore, due to the growing number of local rotations available to the agent and the sparseness of the reward, it becomes increasingly difficult to find and reward optimal sequences. This also shows the benefit of implementing curriculum strategies in such hybrid discrete-continuous optimization problems, where the landscape both for the RL agent and for the numeric optimizer become increasingly sparse. We also note that the generation of 7 and 8-qubit UCC operators proves more challenging. In particular, for the 7-qubit UCC unitary, the agent is able to raise the fidelity to values close to $F = 0.99$, but it cannot significantly increase the number of circuits with $F > 0.99$ over the course of the training. In the case of the 8-qubit UCC operator, the agent instead proves capable of discovering sequences with very high fidelity, i.e $1 - F < 10^{-5}$, although the average fidelity worsens slightly towards the end of the training. We also see that an ensemble of agents can successfully reduce the size of the best circuit, even in those cases where the average performance worsens during training.
We would like now to consider a (black-box) unitary $U(\tau) = e^{-iH\tau}$, where τ is the evolution time described by a Hamiltonian of the following type:

$$\begin{aligned} H = -2h \sum_{i=1}^{n} \sigma_z^{(i)} - J \sum_{i=1}^{n-1} \Big(\sigma_x^{(i)} \otimes \sigma_x^{(i+1)} + \sigma_y^{(i)} \otimes \sigma_y^{(i+1)} \\ + \Delta \Big(\sigma_z^{(i)} \otimes \sigma_z^{(i+1)} - \frac{1}{4} I \Big) \Big), \end{aligned} \tag{5.32}$$

which is usually referred to as the XXZ model [Tak99]. We want to analyze how the optimal quantum circuit found by the RL agent changes when we vary the Hamiltonian parameters. For the XXZ model, this parameter variation represents for instance the transition from a XX model when $\Delta = 0$ to a XXX model for $\Delta = 1$, or from ZZ interaction for $J = 0$ to a XXZ model for $J > 0$. In fact, we observe in different parameter regions different behaviours of the circuit structure. Results of several RL runs for two different configurations, one with fix coupling $J = 0.5$ and varying Δ and another one with fix $\Delta = 0.5$ and varying J of the XXZ model unitary are shown in Fig. 5.5 for 3, 4 and 5 qubits. Plots (a), (e) show the maximal fidelity found by the agents for each run, plots (b), (f) the mean fidelity, plots (c), (g) the number of gates in the shortest circuit among those with the highest fidelity and plots (d), (h) the average circuit size, all represented in their functional dependence from the coupling J and transverse coupling Δ of the XXZ model for two different configuration of the transverse coupling $\Delta = 0.5$ with varying J and and $J = 0.5$ with varying Δ (dark green, orange and blue and pink, light green and yellow, respectively). We also see how rapidly the circuit size grows in the presence of entanglement, for example from $J = 0$ to $J = 1$, whereas the increase in average circuit size seems less pronounced as we vary Δ. We observe that the optimal circuit size can also decrease, for example when $\Delta = 1$ and $J = 0.5$. We also see some instabilities and high variance in both the average circuit size and the size of the best circuit discovered, though it is hard to determine whether the agent fails at finding a shorter circuit for a specific parameter or the problem becomes suddenly harder to represent with the given gate set due to the parameter variation.

5.6 Conclusion

In this work we construct a framework to learn both classically simulated and black-box unitaries on a quantum circuit using RL and unconstrained optimization. Our simulation is specifically tailored to an ion-trap architecture based on collective gates and local rotations. We demonstrate the synthesis of optimal circuits for Toffoli gates and UCC operators for varying numbers of qubits. As instances of black-box unitaries, we also consider Hamiltonian simulations of the XXZ model. More specifically, we study the convergence and the quality of the solutions found by agent and optimizer as we modify relevant parameters of the underlying black-box unitary, such as the coupling in the XXZ Hamiltonian. After testing the algorithm on different unitary process simulation scenarios, we observe that the optimization of circuits is generally possible even for large numbers of gates, it is however difficult to foresee how sparse the cost function minima can be as we increase the number of qubits and reduce the sparsity in the corresponding Hamiltonian. Possible improvements include combining the RL search with a graph traversal algorithm to have a more efficient exploration of the discrete action space [Dal&al20; ZZZ&al20]. We expect that this approach can be applied to different architectures beyond trapped ions, and more carefully engineered reward functions can be developed to further enhance the discovery of optimal circuits. Our unified optimization, combining circuit compilation and unitary synthesis, may find use both in classical pre-optimization and in experimental circuit learning, be it for in-situ (variational) algorithms or module-compilation tasks.

5.7 Data and Code availability

The code and the data are available at the following link: https://github.com/franz3105/RL_Ion_gates.

Acknowledgement

We would like to thank Markus Schmitt, Robert Zeier and Marco Canteri for the useful discussions. FP, MS and FM acknowledge support from the German Federal Ministry of Education and Research (BMBF) within the framework programme "Quantum technologies – from basic research to market" (Project QSolid, Grant No. 13N16149), the European Union's Horizon Programme (HORIZON-CL4-2021-DIGITALEMERGING-02-10) Grant Agreement 101080085 QCFD, by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy – Cluster of Excellence Matter and Light for Quantum Computing (ML4Q) EXC 2004/1 – 390534, and from the Jülich Supercomputing Center through the JUWELS and JURECA cluster. SJ acknowledges the Austrian Academy of Sciences as a recipient of the DOC

Fellowship. This research was funded in whole or in part by the Austrian Science Fund (FWF) through the DK-ALM W1259-N27 ([Grant DOI: 10.55776/W1259] and the SFB BeyondC F7102 [Grant DOI: 10.55776/F71]. For open access purposes, the authors have applied a CC BY public copyright license to any author-accepted manuscript version arising from this submission. This work was also co-funded by the European Union (ERC, QuantAI, Project No. 101055129). Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Research Council. Neither the European Union nor the granting authority can be held responsible for them. Simulations were realized using the libraries NUMBA [LPS15], JAX [BFH&al18], PYTORCH [PGM&al19] and SCIPY [Vir&al20]. The robot picture in Fig. 5.1 and Fig. 5.2 was generated by the authors using Midjourney.

5.8 Appendix

5.8.1 Fast analytic ion gates

In this section, we derive the representations for the MS and C_{xy} gates based on their spectral decomposition and we show how these can be further simplified. We also provide descriptions of the gate gradients based on the optimized representations. The n-qubit XY-Rotation and MS Hamiltonians are given by

$$\hat{H}_{\mathrm{XY}}(n,\phi) = S_x\cos(\phi) + S_y\sin(\phi), \tag{5.33}$$

$$\hat{H}_{\mathrm{MS}}(n,\phi) = \hat{H}_{\mathrm{XY}}(n,\phi)^2 = \left(S_x\cos(\phi) + S_y\sin(\phi)\right)^2, \tag{5.34}$$

We will first focus on solving the XY-rotation gate and then generalize our solution to the MS gate, using the fact that the MS-Hamiltonian is the square of the XY-Hamiltonian. A representation of the XY unitary in terms of its real and imaginary entries for $n = 8$, $\theta = \frac{3}{4}\pi$ and $\phi = 2\pi$ is given in Fig. 5.6.

5.8.1.1 General Approach - Constructing the XY-rotation Gate

Due to the lack of multi-qubit interaction terms in the XY-rotation Hamiltonian, the eigenvectors and eigenvalues of the n-qubit case can be constructed from the single qubit closed-form solution, allowing us to factorize the evolution into ϕ dependent terms and θ dependent terms (θ can be understood as the time evolution parameter, using a standard notation of the literature of trapped-ion quantum computing). This allows us to calculate the associated unitaries from element-wise operations. We decompose the Hamiltonian into

$$\hat{H}_{\mathrm{XY}}(n,\phi) = \hat{V}_n(\phi)\Lambda_{\mathrm{XY}}\hat{V}_n^{\dagger}(\phi), \tag{5.35}$$

with the ϕ dependent $\hat{V}_n$ and the diagonal matrix of the eigenvalues Λ_{XY}. For the unitaries the eigenvalues then capture the θ dependence via $\exp\left(-i\Lambda_{\mathrm{XY}}\frac{\theta}{2}\right)$.

5.8.1.1.1 Eigenvalues The eigenvalues are constructed from the single qubit case using the following Kronecker sum notation $\hat{A}^{\oplus n} = \sum_{i=1}^{n} \mathbb{I}^{\otimes i-1} \otimes \hat{A} \otimes \mathbb{I}^{\otimes n-i}$,

$$\Lambda_{\mathrm{XY}} = \operatorname{diag}\left(2\mathbf{b}_n - n\right), \quad \text{with} \quad \mathbf{b}_n = \begin{pmatrix} 0 \\ 1 \end{pmatrix}^{\oplus n}. \tag{5.36}$$

We find that the i-th eigenvalue can be constructed from the binary Hamming weight vector $\mathbf{b}_n$, because the i-th component of the binary Hamming weight $b_n(i)$ corresponds to the number of qubits with $|1\rangle$ at index i of the Hilbert space.

5.8.1.1.2 Eigenvectors The eigenvector matrices $\hat{V}_n(\phi)$ are also constructed from the single qubit case, but using the tensorproduct,

$$\hat{V}_n(\phi) = \frac{1}{\sqrt{2^n}} \begin{pmatrix} 1 & 1 \\ -e^{i\phi} & e^{i\phi} \end{pmatrix}^{\otimes n} = \hat{P}_n(\phi) \odot \frac{1}{\sqrt{2^n}} \underbrace{(\hat{\mathbb{I}}_2 + i\sigma_y)^{\otimes n}}_{=\hat{S}_n}. \tag{5.37}$$

We furthermore decompose them into an elementwise product ($\odot$) of a sign $\hat{S}_n$ and phase $\hat{P}_n(\phi)$ component. The phase components $\hat{P}_n(\phi)$ are column-independent $\hat{P}_n(\phi) = [\mathbf{p}_n(\phi), \mathbf{p}_n(\phi), \cdots, \mathbf{p}_n(\phi)]$, where the vectors themselves can be regarded as the element wise complex exponential v_ϕ of the binary Hamming weight vector $\mathbf{b}_n$

$$\mathbf{p}_n(\phi) = \begin{pmatrix} 1 \\ e^{i\phi} \end{pmatrix}^{\otimes n} = \exp(i\phi\mathbf{b}_n) = v_\phi(\mathbf{b}_n). \tag{5.38}$$

5.8.1.1.3 Calculating the Unitary We can now use the column independence of the phase components $\hat{P}_n(\phi)$, to commute it with the eigenvalues. This allows us to separate the phase component further. For the unitary of the XY-gate we then find

$$\begin{aligned} C_{xy}(\phi,\theta) &= \exp\left(-i\hat{H}_{\mathrm{XY}}(\phi)\theta\right) = \hat{V}_n(\phi)\exp(-i\Lambda_{\mathrm{XY}})\theta)\hat{V}_n^\dagger(\phi) \\ &= \frac{1}{2^n} \underbrace{\hat{P}_n(\phi)\hat{P}_n^\dagger(\phi)}_{=v_\phi(\hat{C})} \hat{S}_n \exp(-i\theta/2\Lambda_{\mathrm{XY}})\hat{S}_n^\dagger. \end{aligned} \tag{5.39}$$

We substitute the phase term product with the previously defined element wise complex exponential of $\hat{C} = \hat{P}_n - \hat{P}_n^T$. Furthermore we split the sign components into degeneracy subspaces one for each different eigenvalue in λ_{XY}. We decompose $\hat{S}_n = \sum_{i=0}^{n} \hat{S}_{\lambda_i}$, where $\hat{S}_{\lambda_i}$ retains the values of $\hat{S}_n$ for columns with binary Hamming weight i, but is zero for all other columns. From this we can then construct cached matrices $\hat{D}_{\lambda_i} = \hat{S}_{\lambda_i}\hat{S}_{\lambda_i}^\dagger$, so that we can rewrite

$$\hat{S}_n \exp(-i\theta/2\Lambda_{\mathrm{XY}})\hat{S}_n^\dagger = \sum_{i=0}^{n} e^{-i\lambda_{\mathrm{XY},i}\frac{\theta}{2}}\hat{D}_{\lambda_i}. \tag{5.40}$$

From this we conclude

$$C_{xy}(\phi,\theta) = \frac{1}{2^n} v_\phi(\hat{C}) \odot \sum_{i=0}^{n} \exp\left(-i\lambda_{\mathrm{XY,i}}\theta\right) \hat{D}_{\lambda_i}. \tag{5.41}$$

The element wise exponential $v_\phi(\hat{C})$ is constructed efficiently by reusing $2n+1$ phase values, because $C_{ij} \in [0, 1, \cdots, 2n]$. Each of the required computations require $\mathcal{O}\left((2^n)^2\right)$ operations, resulting in a total complexity of $\mathcal{O}\left((n+1)(2^n)^2\right)$.

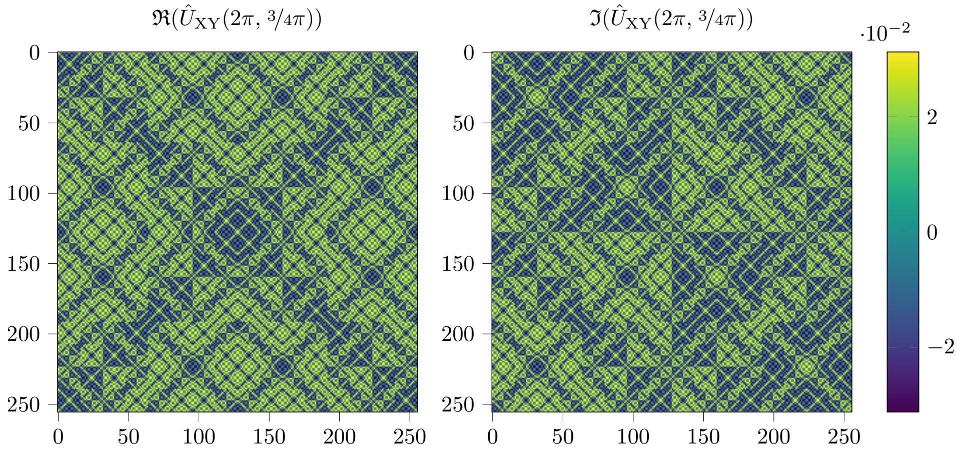


Figure 5.6: Real and imaginary components of the unitary matrix of the n-qubit XY-rotation gate with $\phi = 2\pi$ and $\theta = 3/4\pi$. The colormap describes the value range of the matrix elements.

5.8.1.2 Generalization to the MS gate

The approach used for C_{xy} gate can be generalized to the MS gate. Due to its Hamiltonian being the square of the aforementioned Hamiltonian, the two gates share their eigenvectors, while the eigenvalues of the C_{xy} gate are squared $\lambda_{\mathrm{XY,i}}^2 = \lambda_{\mathrm{MS,\,i}}$. This reduces the number of different eigenvalues from $n+1$ to $\lceil \frac{n}{2} \rceil + \mod(2+1, 2)$. The expansion in Eq. (5.41) is transformed into

$$\mathrm{MS}(\phi,\theta) = \frac{1}{2^n} v_\phi(\hat{C}) \odot \left(\sum_{i=0}^{\lceil \frac{n}{2} \rceil} \exp\left(-i\lambda_{\mathrm{MS,i}}\theta\right) \overbrace{\left(\hat{D}_{\lambda_i} + \hat{D}_{\lambda_{n-i}}\right)}^{=\tilde{D}_i} \delta_i^{\mathrm{floor}} \right). \tag{5.42}$$

where

$$\delta_i^{\mathrm{floor}} = \begin{cases} 1 & \text{if } 0 \le i \le \frac{n}{2} \\ 0 & \text{otherwise.} \end{cases} \tag{5.43}$$

As both expansions, Eq. (5.41) and Eq. (5.42), rely solely on operations local to the matrix elements, these operations are ideal for parallelisation. In Fig. 5.7 we show the improvement in terms of computation time (a) and speedup (b) of our method compared to standard matrix exponentiation as a function of the number of qubits : in (a) the orange line shows the computation time of the matrix exponential, whereas the green line shows our method implemented without caching the parameter-independent matrices, and the blue line our method again, but with cached matrices; (b) shows instead the ratio between the computation time of the cached method and standard matrix exponentiation. We see that we can reach a speedup between 50 and 250 for $n \le 12$, which proves particularly useful in our quantum-circuit simulations: In fact, these require large numbers of cost function evaluations, due to the presence of both the reinforcement learning agent and the gradient-based optimization.

5.8.1.3 Gradients and Hessians

The gradient of the C_{xy} gate can then be computed analytically via

$$\frac{\partial}{\partial\phi} C_{xy}(\phi,\theta) = i\hat{C}\frac{1}{2^n} v_\phi(\hat{C}) \odot \sum_{i=0}^{n} \exp\left(-i\lambda_{\mathrm{XY,i}}\theta\right) \hat{D}_{\lambda_i}, \tag{5.44}$$

$$\frac{\partial}{\partial\theta} C_{xy}(\phi,\theta) = \frac{1}{2^n} v_\phi(\hat{C}) \odot \sum_{i=0}^{n} -i\lambda_{\mathrm{XY,i}} \exp\left(-i\lambda_{\mathrm{XY,i}}\theta\right) \hat{D}_{\lambda_i}. \tag{5.45}$$

For the derivative by ϕ the sum can be reused from the previous gate construction in Eq. (5.41) further reducing computational demands. For the MS gate we find

$$\frac{\partial}{\partial\phi}\mathrm{MS}(\phi,\theta) = i\hat{C}\frac{1}{2^n} v_\phi(\hat{C}) \odot \left(\sum_{i=0}^{\lceil \frac{n}{2} \rceil} \exp\left(-i\lambda_{\mathrm{MS,i}}\theta\right) \overbrace{\left(\hat{D}_{\lambda_i} + \hat{D}_{\lambda_{n-i}}\right)}^{=\tilde{D}_i} \delta_i^{\mathrm{floor}} \right) \tag{5.46}$$

$$\frac{\partial}{\partial\theta}\mathrm{MS}(\phi,\theta) = \frac{1}{2^n} v_\phi(\hat{C}) \odot \left(\sum_{i=0}^{\lceil \frac{n}{2} \rceil} -i\lambda_{\mathrm{MS,i}} \exp\left(-i\lambda_{\mathrm{MS,i}}\theta\right) \overbrace{\left(\hat{D}_{\lambda_i} + \hat{D}_{\lambda_{n-i}}\right)}^{=\tilde{D}_i} \delta_i^{\mathrm{floor}} \right). \tag{5.47}$$

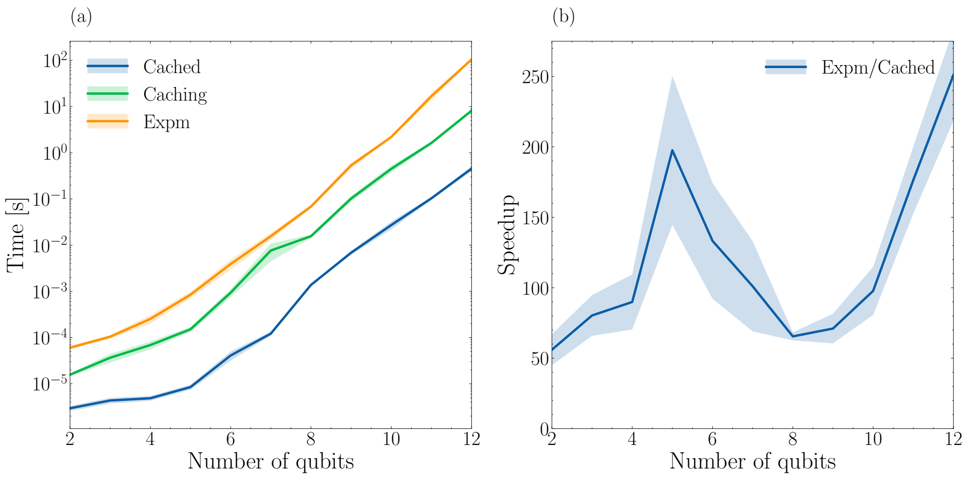


Figure 5.7: (a) Walltime for MS gate construction with standard matrix exponentiation and with the analytical approach of this paper as a function of the number of qubits. The orange line corresponds to the function *scipy.expm*, the blue and green lines represent the analytical approach during caching and after caching, respectively; (b) speedup of the new approach, once the parameter-indedependent matrices have been cached, as a function of the number of qubits. Here we see that the speedup is not constant as the number of qubits changes, but we have a maximum speedup around 5 qubits, which then decreases until we reach 8 qubits, and then increases again. This effect is most likely due to an underlying slowdown in the underlying fundamental operations, e.g., due to the processor cache being filled up quickly, so that access to the RAM becomes necessary. Moreover, our method is implemented only using NUMBA, i.e., compiled PYTHON code, whereas the exponential matrix function of SCIPY profits from underlying time-critical routines written in C, C++ and Fortran. It seems, however, that the speedup grows steadily for values larger than 8 qubits.

Hessians can be derived direcly by applying the chain rule a second time in Eqs. (5.44)-(5.47).

5.8.2 Parameter-shift rules for ion gates

The method for gate computation introduced above allows to express the derivative of a quantum cost function in terms of so-called parameter-shifts rules. These have been studied in the context of variational quantum circuit, quantum control and quantum machine learning. The gates contained in the gate set $Z_1(\theta), ..., Z_n(\theta)$, $C_{xy}(\theta, \phi)$, $\mathrm{MS}(\theta, \phi)$. In general, expectation values with respect to parametrized quantum circuits will have the form

$$C(\boldsymbol{\alpha}) = \langle\psi| V(\boldsymbol{\alpha})\rho V^{\dagger}(\boldsymbol{\alpha}) |\psi\rangle\,, \tag{5.48}$$

for a given quantum state $|\psi\rangle$, where $V(\boldsymbol{\alpha})$ is a gate in the gateset defined in Section (5.2). The cost function depends on a specific target operator ρ.

We first show that the cost function in Eq. (5.48) is equivalent to the cost function of Eq. (5.13), as well as to the expectation value of the local and non-local Hilbert Schmidt circuit, which is given in Ref. [KLP&al19]:

$$C_{HST}(\boldsymbol{\alpha}) = 1 - \mathrm{Tr}\Big\{H(U \otimes V(\boldsymbol{\alpha})^{*})\rho(U^{\dagger} \otimes V(\boldsymbol{\alpha})^{T})\Big\}, \tag{5.49}$$

with $\rho = H = |\phi_{+}\rangle_{A,B}\langle\phi_{+}|_{A,B}$ for the Hilbert Schmidt test, where A and B are the subystems for unitary U and V and

$$|\phi_{+}\rangle_{A,B} = \frac{1}{\sqrt{d}}\sum_{\boldsymbol{j}} |\boldsymbol{j}_A\rangle \otimes |\boldsymbol{j}_B\rangle\,. \tag{5.50}$$

Proof. We show the equivalence for the Hilbert Schmidt test. The proof for the local Hilbert Schmidt test is analogous.
We see that

$$\mathrm{Tr}\Big\{H(UV^{\dagger} \otimes I)\rho(V \otimes U^{*})\Big\} = \frac{1}{d}\sum_{\boldsymbol{j}} \mathrm{Tr}\Big\{|\boldsymbol{j}_A\rangle\langle\boldsymbol{j}_A| UV^{\dagger} |\boldsymbol{j}_A\rangle\langle\boldsymbol{j}_A| VU^{\dagger}\Big\}\,\mathrm{Tr}\{|\boldsymbol{j}_B\rangle \otimes \langle\boldsymbol{j}_B|\},$$

which can be represented as a sum of squared amplitudes of with the same structure of (5.48). □

As a result, the cost functions considered here have to a common description, similar to Eq. (5.48). We derive and comment the corresponding parameter-shift rule for each one of these gates. We consider here the shift of one single parameter at a time, i.e., a scalar rotation angle θ. The gradient is constructed by shifting each parameter according to its own specific parameter-shift rule. The derivatives of the function can be written as:

$$\frac{\partial}{\partial\theta}C(\theta) = \langle\psi| \frac{\partial}{\partial\theta}V(\theta)UV^{\dagger}(\theta) + V(\theta)U\frac{\partial}{\partial\theta}V^{\dagger}(\theta) |\psi\rangle\,. \tag{5.51}$$

Assuming the gate has a generator $V(\theta) = e^{iG\theta}$, where G is a parameter-independent hermitian matrix, then we have:

$$\frac{\partial}{\partial\theta}C(\theta) = \langle\psi| V(\theta)i[G,U]V^{\dagger}(\theta) |\psi\rangle\,. \tag{5.52}$$

In the simplest case, we consider, e.g., $G = \sigma_z^{(i)}$ and obtain [LYP&al17]:

$$[\sigma_z^{(i)}, U] = e^{i\sigma_z^{(i)}\pi/2} U e^{-i\sigma_z^{(i)}\pi/2} - e^{-i\sigma_z^{(i)}\pi/2} U e^{i\sigma_z^{(i)}\pi/2}, \tag{5.53}$$

which leads to

$$\frac{\partial}{\partial\theta} C(\theta) = C\left(\theta + \frac{\pi}{2}\right) - C\left(\theta - \frac{\pi}{2}\right), \tag{5.54}$$

a parameter-shift rule which is valid for the local Z-rotations. For the other two gates, we have to differentiate with respect to both the parameters θ and ϕ. For θ, applying Eq. (15) in Ref. [WIW&al22], we have:

$$\frac{\partial}{\partial\theta} C(\theta,\phi)\bigg|_{\theta=0} = \sum_{l=1}^{2n} \frac{(-1)^{l-1}}{2\sin\left(\frac{2l-1}{2n}\pi\right)} C\left(\frac{2l-1}{2n}\pi, \phi\right). \tag{5.55}$$

By using the general decomposition of the C_{xy} gates, parameter-shift rules can be obtained directly by studying how the eigenvalues and eigenvectors of the unitaries vary as a function of the continuous parameters. Since the C_{xy} gate has $n+1$ distinct eigenvalues we will need at most $2(n+1)$ samples to calculate the derivative – see [WIW&al22]. We observe that the matrices $\hat{V}_n(\phi)$ in Eq. (5.37) can be decomposed further in elementary gates, such as:

$$\hat{V}_n(\phi) = (\mathcal{P}(\phi)HX)^{\otimes n} = \mathcal{P}(\phi)^{\otimes n} H^{\otimes n} X^{\otimes n}, \tag{5.56}$$

where

$$\mathcal{P}(\phi) = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\phi} \end{pmatrix}, H = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}, X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}. \tag{5.57}$$

Let us first consider the C_{xy} gate. The corresponding Hamiltonian has $n+1$ different eigenvalues with energy $\lambda_i = 2i - n$ and $c_k(\phi) = v_k(\phi)^\dagger U v_k(\phi)$ are the overlaps between the target operator U and the gate eigenvectors $v_k(\phi)$ of the gate. For the set of difference pairs we have $\lambda_i - \epsilon_j = 2(i-j) := 2l$. Then Eq. (5.55) becomes

$$\frac{\partial}{\partial\theta} C(\theta,\phi)\bigg|_{\theta=0} = \sum_{k=1}^{n}\sum_{l=1}^{2n} \frac{(-1)^{l-1}}{2\sin\left(\frac{2l-1}{2n}\pi\right)} c_k(\phi) e^{2ik(\frac{2l-1}{2n})}. \tag{5.58}$$

.

This expression can be simplified by evaluating the sum at $\theta = 0$ with respect to the index l.

$$\frac{\partial}{\partial\theta} C(\theta,\phi)\bigg|_{\theta=0} = \sum_{k=1}^{n} c_k(\phi) \frac{\csc\left(\frac{\pi}{2n}\right) e^{\frac{\pi i(k(4n+2)-n)}{n}} \left(e^{\frac{2\pi i(n+1)(n-2k)}{n}} - (-1)^{n+\frac{1}{n}}\right)}{2n\left((-1)^n e^{4\pi i k} + e^{2\pi i n}\right)}. \tag{5.59}$$

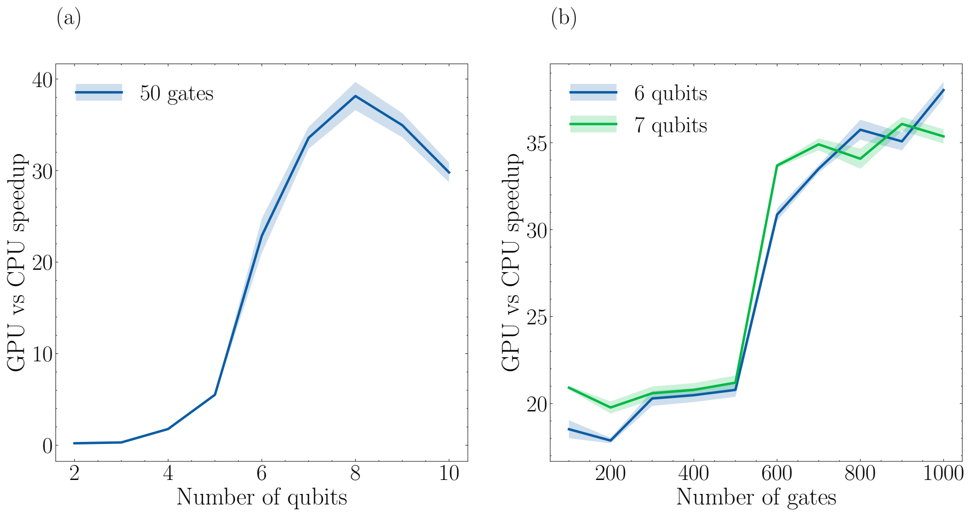


Figure 5.8: Average speedup (30 shots) required to compute the gradient for: (a) the same circuit (50 gates) with increasing numbers of qubits. (b) a 6-qubit (blue) and 7-qubit (green) gate set for increasing number of random gates on the circuit. Shaded regions show the standard deviation for each point (vertical bars are connected together to form a poligon). The orange line represents the gradient compiled with NUMBA on CPU (Intel(R) Xeon(R) Gold 6240 CPU @ 2.60GHz), the second one the gradient compiled with JAX on GPU (NVIDIA Tesla V100S-PCIE-32GB). We see a slowdown in the speedup as we move past 8 qubits: this is probably due to computational overheads that do not profit from the GPU parallel calculations. In fact, we see that the slowdown is present only as we increase the Hilbert space dimension and not as we increase the number of gates.

For the MS gate we obtain instead the following expression:

$$\frac{\partial}{\partial\theta}C(\theta,\phi)\Big|_{\theta=0} = \sum_{k=1}^{n} c_k(\phi)\left\{-\frac{\csc\left(\frac{\pi}{2n}\right)e^{8\pi ikn-\frac{\pi i(n-2k)^2}{n}}\left((-1)^{n+\frac{1}{n}}e^{\frac{2\pi i(n+1)(n-2k)^2}{n}}-1\right)}{2n\left((-1)^n e^{2\pi i(4k^2+n^2)}+e^{8\pi ikn}\right)}\right\}. \tag{5.60}$$

For the derivative of Eq. (5.48) with respect to ϕ, we just have to consider the gate $\mathcal{P}(\phi)$ in Eq. (5.56), whose single-qubit gate generator is given in Eq. (5.61). We observe that due to the simple structure of the gate, the derivative of the $C_{x,y}$ gate with respect to

ϕ is simply given by:

$$\mathcal{P}(\phi) = \exp\left\{\frac{i\phi}{2}(I_n - S_z)\right\} = \exp\left\{\frac{i\phi}{2}\right\}\exp\left\{-\frac{i\phi}{2}S_z\right\}, \tag{5.61}$$

where $S_z = \sum_{i=1}^{n}\sigma_z^{(i)}$ is the collective Z-operator acting on the entire qubit register. After inserting Eq. (5.61) into Eq. (5.48) for a C_{xy} gate, and using the representation in Eq. (5.35), we obtain

$$C(\theta,\phi) = \langle\psi|\left(\exp\left\{-\frac{i\phi}{2}S_z\right\}B_{\mathrm{XY}}(\theta)\exp\left\{\frac{i\phi}{2}S_z\right\}U\exp\left\{-\frac{i\phi}{2}S_z\right\}B_{\mathrm{XY}}(\theta)^\dagger\exp\left\{\frac{i\phi}{2}S_z\right\}\right)|\psi\rangle\,, \tag{5.62}$$

where $B_{\mathrm{XY}}(\theta) = H^{\otimes n}X^{\otimes n}\left(\sum_{l=0}^{n}\exp\{-i\lambda_l\theta\}\,|l\rangle\langle l|\right)X^{\otimes n}H^{\otimes n}$ – the same equation holds for the MS gate, one needs just to replace λ_{XY} with the corresponding eigenvalues λ_{MS}. Considering that $\mathcal{P}(\phi)$ has two eigenvalues, we can write

$$\exp\left\{-\frac{i\phi}{2}S_z\right\} = e^{i\phi}\Pi_z^{(1)} + e^{-i\phi}\Pi_z^{(2)}, \tag{5.63}$$

where $\Pi_z^{(1)}$ and $\Pi_z^{(2)}$ are the projectors on the respective degenerate eigenspaces. By using Eq. (5.63) in Eq. (5.62), we see that Eq. (5.62) has the following form

$$C(\theta,\phi) = b_0(\theta) + b_1(\theta)e^{-i\phi} + b_2(\theta)e^{i\phi} + b_3(\theta)e^{2i\phi} + b_4(\theta)e^{-2i\phi}, \tag{5.64}$$

where $b_i(\theta), i = 0,1,2,3,4$ are ϕ-independent coefficients. Thus, we can write the derivative of C with respect to ϕ as

$$\frac{\partial}{\partial\phi}C(\theta,\phi) = -ib_1(\theta)e^{-i\phi} + ib_2(\theta)e^{i\phi} + b_3(\theta)2ie^{2i\phi} - 2ib_4(\theta)e^{-2i\phi}, \tag{5.65}$$

which can be written as a linear combination of $C(\theta,\phi\pm\frac{\pi}{2})$ and $C(\theta,\phi\pm\frac{\pi}{4})$ cost functions:

$$\frac{\partial}{\partial\phi}C(\theta,\phi) = \tilde{C}(\theta,\phi+\frac{\pi}{4}) - \tilde{C}(\theta,\phi-\frac{\pi}{4}) + (\frac{1}{\sqrt{2}}-\frac{1}{2})\tilde{C}(\theta,\phi-\frac{\pi}{2}) + (\frac{1}{\sqrt{2}}+\frac{1}{2})\tilde{C}(\theta,\phi+\frac{\pi}{2}). \tag{5.66}$$

5.8.3 Algorithmic implementation details

Our framework consists of two main parts: The agent (we consider a PS-LSTM agent with LSTM cells and a linear layer, but any RL agent is a viable option), which should learn to construct a proper representation of the quantum circuit and the optimizer, which has to be equipped with a proper gradient function. The gradient function is constructed according to the standard GRAPE procedure and using the gate representations for the C_{xy} and the MS gates given in Appendix 5.8.1. We use and test two different versions of the gradient: The first one is compiled using NUMBA [LPS15], a

library for fast python code, the second one employs JAX to allow execution on GPU. A comparison of the two gradient functions is given in Fig. 5.8. We observe that the GPU-based function allows for a certain speed-up, in particular as the number of gates on the circuit increases. The main advantage of NUMBA lies in a faster and more straightforward implementation of the parallel optimization runs with different seeds.
At time t of the agent-environment interaction, the agent receives an input (percept) from the environment and outputs an action which is executed onto the environment. The agent-environment interaction has the following structure.

Action: The action of the agent corresponds to placing one of the available gates in the gate set (entangling or non-entangling) onto the quantum circuit. The gate is represented by an integer $a \in 1, ..., |A|$. The array of all the integers chosen by the agent up to time t forms the quantum circuit structure at time t.

Percept: As input to the RL agent, we generally use a one-hot encoding of the circuit. For a circuit of length L and $|A|$ different gates, the percept $s_t \in \{0,1\}^L \times \{0,1\}^{|A|}$ has entries equal to:

$$(s_t)_{i,j} = \begin{cases} 1 & \text{if } i,j = a,t \\ 0 & \text{otherwise.} \end{cases} \tag{5.67}$$

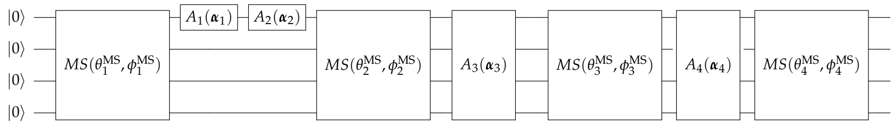
However, for the PS-LSTM agent (or other agents that are modified analogously), a more suitable input can also be used. Since the update of the internal state of the PS-LSTM agent follows the RL steps, the agent can just accept the input at time t as a percept, since previous information about past agent-environment interactions is still processed by the the internal state of the LSTM network. Therefore, the percept becomes the one-hot encoded vector

$$(s_t)_j = \begin{cases} 1 & \text{if } j = a \\ 0 & \text{otherwise.} \end{cases} \tag{5.68}$$

This percept only gives the agent partial information about the internal state of the environment, which turns the problem from an MDP (Markov Decision Process) into a so-called POMDP problem (Partially Observable Markov Decision Process) [Pre20]. Other agents than those derived by PS may use different inputs, based on their specific convergence properties in dealing with different environments. In general, if the agent can accept a recurrent or autoregressive network as a policy, the percept given in Eq. (5.68) may be used. It could be, however, that the algorithm needs to be modified appropriately to function with this type of percept. The first type of input has been used for the simulation of UCC operators, whereas the second one has been implemented in all the other simulations. In general, both percepts lead to similar results, but the second one should be preferred for the PS-LSTM architecture and most importantly does not scale with the size of the circuit, leading therefore to faster forward passages in the policy network.

Reward and curriculum: Throughout the development of this manuscript, several reward systems were tested. In general, we used a two-step reward that assigns a smaller

(a) Hybrid layer-based-RL environment with action space $A = \{C_{xy}, Z_1, ..., Z_n\}$ and fixed MS gates.


$|0\rangle$
$|0\rangle$
$|0\rangle$
$|0\rangle$
$MS(\theta_1^{\text{MS}}, \phi_1^{\text{MS}})$
$A_1(\boldsymbol{\alpha}_1)$
$A_2(\boldsymbol{\alpha}_2)$
$MS(\theta_2^{\text{MS}}, \phi_2^{\text{MS}})$
$A_3(\boldsymbol{\alpha}_3)$
$MS(\theta_3^{\text{MS}}, \phi_3^{\text{MS}})$
$A_4(\boldsymbol{\alpha}_4)$
$MS(\theta_4^{\text{MS}}, \phi_4^{\text{MS}})$

(b) Full RL environment with action space $A = \{\text{MS}, C_{xy}, Z_1, ..., Z_n\}$.

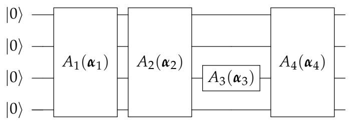

$|0\rangle$
$|0\rangle$
$|0\rangle$
$|0\rangle$
$A_1(\boldsymbol{\alpha}_1)$
$A_2(\boldsymbol{\alpha}_2)$
$A_3(\boldsymbol{\alpha}_3)$
$A_4(\boldsymbol{\alpha}_4)$

Figure 5.9: Quantum circuit representation of two possible RL environments for quantum circuit optimization. The first environment (a) resembles the structure of the layer-based compilation, that is, the entangling MS gates are fixed and the agent can place rotations on the circuit between the entangling layers. The second one (b) has no pre-defined circuit structure but rather leaves the agent completely free to place any gate withing the gate set on the quantum circuit, with only one additional simplification: two gates of the same type placed immediately one after the other are automatically merged to form one single gate. This is done to prevent the agent from getting stuck in loops, i.e., local minima, where it keeps choosing the same gate over and over again. In general, the first circuit reduces the size of the action space and therefore the possible shapes of the corresponding cost function, hence reducing exploration in favour of a more standardized search.

reward value when the cost function minimum falls below the actual curriculum threshold ϵ_t and a larger reward value when the cost function minimum falls below the global target threshold ϵ_{min}. Both curriculum thresholds can be adjusted depending on the gate synthesis problem to be tackled.
The episode terminates when the cost function minimum in a given time step falls below the threshold ϵ_t or when the maximal length of the circuit per episode, L_{max}, is reached. The RL training terminates upon reaching the maximum number of episodes E_{max}. The reward scheme helps to progressively increase the fidelity throughout training without allowing for too long circuits. The threshold is then lowered as episodes progress based on previous rewards obtained by the agent. In our implementation, we lower the threshold when it has been surpassed by the agent at least 500 times using

the following scheme:

$$\epsilon_{t+1} = \begin{cases} \epsilon_{\min} + \frac{1}{2}(\epsilon_{\min} - \epsilon_t) & \text{if } \epsilon_{\min} \le \epsilon_t \le 1 \\ \epsilon_{\min} & \text{if } \epsilon_t \le \epsilon_{\min} \\ 1 & \text{otherwise.} \end{cases} \tag{5.69}$$

Environments: We propose two different types of agent-circuit interaction. In the first type – see Fig. 5.9 (a) –, which is more similar to layer-based compilation, the agent only places rotations on the circuit, while the entangling gates are fixed in position. While the episode progresses, more entangling gates are added to the environment, until a maximum $L_{\max}$ of gates on the circuit is reached. The circuit is also simplified automatically by the environment, in such a way that if the agent places the same two gates on the circuit one after the other, they are reduced to one single gate. This is implemented in order to prevent the agent from getting trapped in local minima where it places the same gate over and over again on the circuit, therefore reducing the necessary training time. In the second type – see Fig. 5.9 (b) –, entangling gates are also given as a possible action on the environment. This second type of environment leaves more room for exploration, but it is also more challenging for the agent. The simulations presented in this work were realized by using only the environment that allows for completely free gate placement (so both the rotation gates as well as the entangling gates).

Agents: In order to test the effectiveness of our algorithm, we employ different agents for testing. We consider the standard state-of-the-art PPO algorithm [Bar21] and the two versions of PS with deep energy-based neural networks PS-DEBN and PS-LSTM. We observe that PPO performs slightly worse than PS-LSTM, but better than PS-FNN, but is probably due to the non-recurrent version of the PPO algorithm implemented. Due to the large action and percept space, we did not include standard PS in the comparisons, since this would require to store large h-matrices for each percept-action transition, leading to slow computation and eventually memory overflow. This algorithm can however be used in circuit synthesis problems with a small number of gates and qubits.

Optimizers: As an unconstrained optimizer, in this work we only consider the L-BFGS-B algorithm, as it is implemented in SCIPY [Vir&al20]. The number of iterations of the optimizers for each RL interaction is set to 100. Both optimizers can run a given number of optimization attempts in parallel with different seeds (a technique usually referred to as random restart), which should prevent the optimizer from getting trapped in local minima.

Simulations: In this work we consider three different groups of simulations: the synthesis of a 3-qubit Toffoli gate on a 3-qubit and 4-qubit circuit, the synthesis of UCC (Unitary Coupled Cluster) operators and the synthesis of the unitary of the XXZ Hamiltonian with varying Hamiltonian parameters. In the case of the XXZ Hamiltonian and the Toffoli gate, the simulation is realized on CPUs (72 Intel(R) Xeon(R) Gold

Gate type	A^* search	Dijkstra	Greedy	our method
MS gate	5	3	4	3
$C_{x,y}$ gate	3	4	94	4
Z gate	27	24	297	3
Depth	17	15	113	10
Runtime	17 s	63 s	161 s	1.6 h

Table 5.1: Result of the compilation of the 3-qubit Toffoli gate with BQSKit [YIL&al21] using the gate set given by Eqs. (5.1)-(5.3) and three of the given search heuristics: A^*, Dijkstra, and greedy. The threshold is set to $\epsilon = 10^{-2}$. We see that the compilation uses a considerable amount of Z gates and a larger amount of collective gates than the RL algorithm. We see, however, that a standard compiler can be significantly faster than a RL-based search with 10000 episodes and it is therefore possible to apply further pruning methods and iterative optimization loops on top of the main compilation algorithm.

6240 CPU @ 2.60GHz), whereas the simulation of the UCC operators is performed on GPUs (4 NVIDIA A100 Tensor Core GPU with 40 GB). This means, the library used to compute cost functions and gradients in order to optimize them is NUMBA for the CPU-based simulations and JAX for GPU-based ones. The reason is that NUMBA allows for faster parallelization with the JOBLIB library that proves difficult to achieve with JAX, thereby enabling us a better exploration of the cost function landscape for small numbers of qubits, whereas JAX is significantly faster for larger numbers of qubits. In particular, we set the number of optimization runs per RL iteration to 10 when we employ the NUMBA version, whereas we use only 1 when we employ the code using JAX. The curriculum threshold was kept to $\epsilon_{\text{min}} = 10^{-2}$. We used two-layered LSTM networks implementing the policy given in Eq. (5.23) and 128 neurons, a training batch size of 64, a learning rate of 0.01, a curriculum update window of 500. The agent is trained with a replay-memory upon finding a gate sequence with infidelity lower than the curriculum threshold, and also every 100 agent-environment interactions if the replay memory is large enough. The target network is updated every 50 agent-environment interactions. The number of iterations of the optimizer in the hybrid RL-continuous simulations is fixed to 100. The temperature parameter β of the softmax distibution that parametrized the PS-LSTM policy – see Eq. (5.19) – is annealed from $\beta = 10^{-3}$ to $\beta = 1$ according to a linear schedule based on the number of episodes, in order to force the agent, towards the end of the training, to consider shorter and shorter circuits, thereby reducing the exploration. Other parameters vary based on the simulation considered. The data and hyperparameters can be found in `https://github.com/franz3105/RL_Ion_gates`.

Comparison with BQSKit: BQSKit is a quantum circuit compilation library developed by the Berkeley National Laboratory [YIL&al21]. It supports compilation of both discrete and variational circuits with different algorithms, such as tree-search with BFS, qsearch, etc. and allows for the implementation of custom gate sets. For comparison, we implement the trapped-ion gate set used in this work – see also Eqs. (5.1)-(5.3)

– and run the compilation using the QSearch algorithm implemented in the aforementioned library, which is based on multiple tree traversal search algorithm such as the A^* algorithm [HNR68] or the Dijkstra algorithm [HN19]. The results of these runs for the Toffoli gate are given in Table 5.1: we see that the A^* and Dijkstra routines give similar solutions, whereas the greedy heuristic proves significantly worse overall. The compilation routine offered by BQSKit, while significantly faster than the RL method, is unable to converge to the same compact solution discovered by the PS-LSTM algorithm.

6 Optimal two-qubit gates in recurrence protocols of entanglement purification

Disclaimer: A modified version of this chapter was published in Ref. [PCT&al22]. FP and JZB developed the theoretical framework of the paper. FP developed the code and ran all the simulations. JZB wrote the introduction to the purification protocols with inputs from FP and MT. FP and JZB worked together on the plots. FP wrote the discussion of the results. TC provided useful guidance and revisions to the manuscript. All authors contributed significantly to the revision of the manuscript.

We propose and investigate a method to optimize recurrence entanglement purification protocols. The approach is based on a numerical search in the whole set of $SU(4)$ matrices with the aid of a quasi-Newton algorithm. Our method evaluates average concurrences where the probabilistic occurrence of mixed entangled states is also taken into account. We show for certain families of states that optimal protocols are not necessarily achieved by bilaterally applied controlled-NOT gates. As we discover several optimal solutions, the proposed method offers some flexibility in experimental implementations of entanglement purification protocols and interesting perspectives in quantum information processing.

6.1 Introduction

Entanglement is a key resource for several tasks in quantum information, quantum simulations [MCD&al21], computation [AL18], and communication [PAB&al20]. These tasks are based on the idea of creating networks of quantum systems, where the generation of maximally entangled sates between qubits in spatially separated nodes is essential. These networks, which may consist of distant or nearby nodes, have been thoroughly investigated for efficient processing and transfer of quantum information [ABB&al21]. However, due to interactions with an uncontrollable environment, mixed or non-maximally entangled states are produced. To protect quantum information and to guarantee a high performance of its processing, one can use quantum error correction [Per85; DMN13] or quantum teleportation in combination with entanglement purification [BBP&al96; DEJ&al96; BDS&al96]. Quantum error correction is characterized by the quantum capacity of the transmission channel, which can be compared with the yield of entanglement purification for both one- and two-way classical communication [BDS&al96]. The latter is the subject of this article. In particular, we consider a so-called recurrence protocol [DB07], an iterative approach, which operates in each purification step only on two identical copies of states.

In this paper, we discuss entanglement purification from the point of view of optimality. Recently, optimized entanglement purification has been investigated with the help of genetic algorithms [KAJ19], where the analytical and numerical studies are based on Werner states [Wer89]. In fact, also the first ever proposed protocols are based on either Werner [BBP&al96] or Bell diagonal states [DEJ&al96]. It has been shown that 4 or 12 local random $SU(2)$ transformations can convert any state into a Bell diagonal or Werner state, respectively [BDS&al96]. We have already argued that these random local unitary transformations and states obtained in consequence are only useful for grasping and understanding the complex task of entanglement purification because they may waste useful entanglement [TB16]. Therefore, here, we develop a method for general states and demonstrate it for several simple examples including also the Werner state. Nonetheless, our motivation also lies in the fact that an experimental implementation may not have enough control over the generated mixed entangled states and thus more general, adaptive, and optimal entanglement purification strategies have to be made available. In this general context, we assume that an experiment can still guarantee identical copies of states before the protocol takes place.

Our method is based on quasi-Monte Carlo numerical sampling of the states, which undergo the purification protocol, and then the concurrence [Woo98] of the output states is integrated over an *a priori* probability distribution function. This results in an average two-qubit gate-dependent cost function for the whole sample of states. We employ a quasi-Newton algorithm [LN89] to solve this non-linear optimization problem on the whole $SU(4)$ group.

We focus on increasing entanglement in each step, therefore, the obtained two-qubit gate is optimal in the sense that it achieves, on average, a higher increase of entanglement of an input family of states for a given purification step. This is beneficial in reducing the number of qubit pairs required to purify a single two-qubit state, as this number grows exponentially with the number of steps. We discuss the performance of our method for several examples and compare with protocols based on one bilateral application of controlled-NOT (CNOT) gates, the paradigmatic two-qubit operation used in the seminal purification protocols [BBP&al96; DEJ&al96].

The paper is organized as follows. In Sec. 6.2 we introduce our method and give some elementary examples to allow further acquaintance with the concept of the introduced cost function. In Sec. 6.3 we demonstrate our approach for one two-parameter and four one-parameter family of states. Numerical and analytical results are presented for concurrences and success probabilities. In Sec. 6.4 we summarize and draw our conclusions. Some details supporting the main text are collected in the Appendix 6.6.1.

6.2 Method

In this section, we describe how the recurrence protocol is optimized with respect to an input family of quantum states.

6.2.1 Entanglement purification protocol

Let us consider the product state of two-qubit pairs

$$\boldsymbol{\rho} = \rho^{A_1,B_1} \otimes \rho^{A_2,B_2}, \tag{6.1}$$

where qubit components of each pair are assumed to be spatially separated at nearby or distant locations. These locations are labeled by A and B. In an entanglement purification protocol, one performs local quantum operations, which may not involve just two-qubit gates [Ber17].
This is followed by measurements on one of the pairs at both locations. A classical communication between A and B results in a qubit pair with a higher degree of entanglement. Both pairs are assumed to start in the same state ρ and the degree of the entanglement is usually measured by the fidelity with respect to one of the Bell states

$$|\Psi^{\pm}\rangle = \frac{1}{\sqrt{2}}\left(|01\rangle \pm |10\rangle\right), |\Phi^{\pm}\rangle = \frac{1}{\sqrt{2}}\left(|00\rangle \pm |11\rangle\right). \tag{6.2}$$

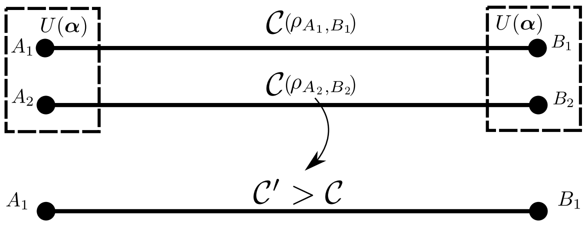
However, these states can be subject to local unitary transformations, which can cause some technical difficulties, when one uses fidelity, i.e. using fidelity as a cost function would force us to search also for additional local unitary operations in order to align the output state of the protocol with the Bell basis.
Furthermore, using a fidelity restricts the purification process to a particular basis. For instance, in Refs. [BBP&al96; DEJ&al96] a state can be purified only if it presents fidelity greater than 1/2 with respect to any Bell state in Eq. (6.2). In this setting a mixed state with fidelity close to one with respect to the maximally entangled state $|\Psi_{\mathrm{M}}\rangle = (|\Phi^-\rangle + i\,|\Phi^+\rangle + i\,|\Psi^-\rangle + |\Psi^+\rangle)/2$ would not be purifiable, as $|\Psi_{\mathrm{M}}\rangle$ has a fidelity of $1/4$ with respect to any Bell state. As we intend to analyze the entanglement purification in a very general setup, we require an entanglement measure that is invariant under local unitary transformations. Therefore, we turn to the concurrence as a measure of the attainability of a maximally entangled state [Woo98]:

$$\mathcal{C}(\rho) = \max\{0, \kappa_1 - \kappa_2 - \kappa_3 - \kappa_4\}. \tag{6.3}$$

Here $\kappa_1, \kappa_2, \kappa_3, \kappa_4$ are the square roots of the non-negative eigenvalues (enumerated from the largest one to the smallest one) of the non-Hermitian matrix :

$$\tilde{\rho} = \rho(\sigma_y \otimes \sigma_y)\rho^*(\sigma_y \otimes \sigma_y),$$

where the asterisk is the complex conjugation in the standard basis and σ_y is the Pauli matrix. It is worth noting that there are other possible entanglement measures, such as



$U(\boldsymbol{\alpha})$
$\mathcal{C}(\rho_{A_1,B_1})$
$U(\boldsymbol{\alpha})$
A_1
B_1
A_2
$\mathcal{C}(\rho_{A_2,B_2})$
B_2
A_1
$\mathcal{C}' > \mathcal{C}$
B_1

Figure 6.1: Schematic representation of bipartite entanglement purification, where the entanglement purification protocol trades two entangled qubit pairs for a qubit pair with a higher degree of entanglement, which is quantified by the concurrence $\mathcal{C}$.

the entanglement formation or the relative entropy of entanglement [PV07], but we do not consider them in this article, because the concurrence is, from a numerical point of view, a more tractable entanglement measure for two-qubit states.

In this paper, we examine and optimize a purification protocol having the following steps.

(i) Two unitary transformations $\rho \to U(\boldsymbol{\alpha})\rho U(\boldsymbol{\alpha})^\dagger$ are applied locally at A and B (see Fig. 6.1), where U is a general two-qubit unitary described by a parameter vector $\boldsymbol{\alpha}$. After the application of the quantum operation the four-qubit system attains the state

$$\boldsymbol{\rho}' = U_{A1,A2}(\boldsymbol{\alpha})U_{B1,B2}(\boldsymbol{\alpha})\boldsymbol{\rho}U^\dagger_{B1,B2}(\boldsymbol{\alpha})U^\dagger_{A1,A2}(\boldsymbol{\alpha}). \tag{6.4}$$

(ii) One of the pairs (A_2, B_2) is then locally measured in the standard basis. There are four possible states, i.e., four-dimensional vectors, in which one can find the measured pair:

$$\begin{aligned} |1\rangle &= |00\rangle_{A_2,B_2}, \quad |2\rangle = |01\rangle_{A_2,B_2}, \\ |3\rangle &= |10\rangle_{A_2,B_2}, \quad |4\rangle = |11\rangle_{A_2,B_2}. \end{aligned}$$

A successful measurement of one of the states $|i\rangle$ with $i \in \{1,2,3,4\}$ results in a state for the other qubit

$$\tilde{\rho}_i^{A_1,B_1} = \frac{\langle i|\boldsymbol{\rho}'|i\rangle}{\mathrm{Tr}\{|i\rangle\langle i|\boldsymbol{\rho}'\}} \tag{6.5}$$

with probability

$$p_i = \mathrm{Tr}\{|i\rangle\langle i|\boldsymbol{\rho}'\}.$$

(iii) Depending on the value of the measurement results, which are communicated between the two parties, the state with the largest concurrence and the related success probability are kept, whereas the others are discarded. The output of the protocol is the pair $(\mathcal{C}', P)$, with the concurrence value $\mathcal{C}'$ of the state obtained with probability P. If there are multiple maxima, e.g. , $\mathcal{C}(\tilde{\rho}_1^{A_1,B_1}) = \mathcal{C}(\tilde{\rho}_2^{A_1,B_1})$, then $\mathcal{C}' = \mathcal{C}(\tilde{\rho}_1^{A_1,B_1})$ and $P = p_1 + p_2$.

Given two copies of a state ρ with concurrence $\mathcal{C}$, it is straightforward to see that the pair $(\mathcal{C}', P)$ depends on the vector $\boldsymbol{\alpha}$, which we use as the optimization parameter.
Recurrence entanglement purification protocols might include symmetric and asymmetric single-qubit gates before and after the bilateral action of the two-qubit operations [BBP&al96; DEJ&al96]. These are important in an analytical approach to maintain the form of the state after each iteration, however, entanglement is not affected by this process. In this work we omit these specific single-qubit gates, as we are focused on increasing the value of the concurrence and not on the specific form of the state after each iteration step. We may instead consider general single-qubit unitaries as part of more general parametrizations. Furthermore, we stress again that our approach aims to increase the amount of entanglement between the qubits pairs regardless of the basis. This is an improvement concerning previous approaches based on seminal protocols [BBP&al96; DEJ&al96] where a fidelity greater than 1/2 with respect to a Bell state is needed for a working entanglement purification protocol.

6.2.2 Protocol optimization

In order to optimize the protocol described in Sec. 6.2.1, we first define an appropriate parametrization for the two-qubit unitary transformations used in Eq. (6.4). In principle, one should consider elements from $U(4)$, the group of 4×4 unitary matrices, which contains the subgroup $SU(4)$. However, $U(4)$ is the semi-direct product of $U(1)$ and $SU(4)$, where elements of $U(1)$ are rotations of the unit circle [Bak02].
Therefore, choosing elements from $SU(4)$ in Eq. (6.4) represents the most general unitary quantum operation involving two-qubit gates at locations A and B. Elements in $SU(4)$ can be parametrized as [TBS02]:

$$\begin{aligned} U(\boldsymbol{\alpha}) = {} & e^{i\sigma_3\alpha_1} e^{i\sigma_2\alpha_2} e^{i\sigma_3\alpha_3} e^{i\sigma_5\alpha_4} e^{i\sigma_3\alpha_5} e^{i\sigma_{10}\alpha_6} e^{i\sigma_3\alpha_7} e^{i\sigma_2\alpha_8} \\ & e^{i\sigma_3\alpha_9} e^{i\sigma_5\alpha_{10}} e^{i\sigma_3\alpha_{11}} e^{i\sigma_2\alpha_{12}} e^{i\sigma_3\alpha_{13}} e^{i\sigma_8\alpha_{14}} e^{i\sigma_{15}\alpha_{15}}, \end{aligned}$$

with $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \ldots, \alpha_{15})^T \in \mathbb{R}^{15}$ (T denotes the transposition), and [TBS02]:

$$\begin{aligned} & 0 \leqslant \alpha_1, \alpha_3, \alpha_5, \alpha_7, \alpha_9, \alpha_{11}, \alpha_{13} \leqslant \pi, \\ & 0 \leqslant \alpha_2, \alpha_4, \alpha_6, \alpha_8, \alpha_{10}, \alpha_{12} \leqslant \frac{\pi}{2} \\ & 0 \leqslant \alpha_{14} \leqslant \frac{\pi}{\sqrt{3}}, \quad 0 \leqslant \alpha_{15} \leqslant \frac{\pi}{\sqrt{6}}, \end{aligned} \tag{6.6}$$

where σ_i, $i = 1, ..., 15$ are a basis of the Lie group $SU(4)$ (see the Appendix 6.6.1). This is called the Euler angle parametrization of $SU(4)$, which is sufficient to represent every element of the Lie group. For example, the canonical parametrization e^{iH}, where H is a 4×4 self-adjoint matrix with trace zero, is not minimal, because after exponentiation we may have multiple wrappings around the great circles of the 7-sphere.
Our aim is to increase the concurrence, which is a non-linear function of the state ρ and the protocol's unitary matrix U. Furthermore, we have lower and upper bounds on $\boldsymbol{\alpha}$,

as given in Eq. (6.6). We then consider a cost function $f : \mathbb{R}^{15} \to \mathbb{R}$ as

$$f(\boldsymbol{\alpha}) = 1 - \mathcal{C}'(\boldsymbol{\alpha}) \tag{6.7}$$

and our optimization problem is to find a value $\boldsymbol{\alpha}^*$ that leads to a minimum $f(\boldsymbol{\alpha}^*)$. The gradient $\nabla f(\boldsymbol{\alpha})$ can be made available to us due to an automatic differentiation [BFH&al18], so we implement for the optimization a quasi-Newton algorithm the so-called limited memory Broyden-Fletcher-Goldfarb-Shanno (L-BFGS) method [LN89]. Here, we have a constrained optimization problem, because $\boldsymbol{\alpha}$ takes values in the hyper-rectangle defined by Eq. (6.6) and thus we use the L-BFGS approach of Ref. [BLN&al95]. As long as the state to be purified is given, the above approach yields an optimal $\boldsymbol{\alpha}^*$ and a concurrence $\mathcal{C}'$ as close as possible to one.

The vector $\boldsymbol{\alpha}^*$ gives the best two-qubit gate associated with this given state for a given purification step. However, our main aim is to provide the best quantum gates, which are able to purify most effectively certain classes of states and not only a fixed one. This is relevant in a scenario where the generation of distant entanglement is affected by altering noise, leading to slightly different types of states entering the protocol. In order to formulate this quantitatively we introduce a probability density function (PDF) $p(\boldsymbol{x})$, where the vector $\boldsymbol{x}$ defines uniquely the state ρ according to a parametrization. For example, in the case of a Werner state [Wer89]

$$\begin{aligned} \rho(x) &= x\,|\Psi^-\rangle\langle\Psi^-| + \frac{1-x}{3}\,|\Psi^+\rangle\langle\Psi^+| \\ &+ \frac{1-x}{3}\,|\Phi^-\rangle\langle\Phi^-| + \frac{1-x}{3}\,|\Phi^+\rangle\langle\Phi^+|, \end{aligned} \tag{6.8}$$

we have $x \in [0,1]$ and the PDF satisfies

$$\int_0^1 p(x)\,dx = 1.$$

In general, a two-qubit state can be described by 15 parameters, which have to fulfill some non-trivial conditions [Kim03; BK03; Gam16]. The choice of the PDF is not straightforward and the only guideline we have is that the support $\mathrm{supp}(p)$ consists of all $\boldsymbol{x}$, which define entangled states. This is motivated by the fact that separable states are not purifiable. In the case of the Werner states, the support of the PDF is the interval $(0.5, 1]$. Thus, the PDF may put more weight on states with a given concurrence.

Algorithm 5 Optimization of recurrence protocol

```
    Input $\boldsymbol{\rho}(\boldsymbol{x})$, optimizer OPT
    Output $\boldsymbol{\rho}(\boldsymbol{x})$
 1: for $j = 1$ to $M$ do
 2:     $\boldsymbol{x}_j \sim p(\boldsymbol{x}_j)$
 3:     $\boldsymbol{\rho}_j = \boldsymbol{\rho}(\boldsymbol{x}_j) \otimes \boldsymbol{\rho}(\boldsymbol{x}_j)$
 4: end for
 5: for $N = 1$ to $N_{\max}$ do                                  ▷ with random restart
 6:     $U_{AB}(\boldsymbol{\alpha}) = U_{A1,A2}(\boldsymbol{\alpha})U_{B1,B2}(\boldsymbol{\alpha})$
 7:     for $j = 1$ to $M$ do
 8:         $\boldsymbol{\sigma}^j(\boldsymbol{\alpha}) = U_{AB}(\boldsymbol{\alpha})\boldsymbol{\rho}_j U^\dagger_{AB}(\boldsymbol{\alpha})$
 9:         for $l = 1$ to $4$ do
10:             $\boldsymbol{\sigma}^{jl}_A(\boldsymbol{\alpha}) = \frac{\langle l|\boldsymbol{\sigma}^j(\boldsymbol{\alpha})|l\rangle}{\mathrm{Tr}\{|l\rangle\langle l|\boldsymbol{\sigma}^j(\boldsymbol{\alpha})\}}$
11:         end for
12:     end for
13:     $\mathcal{C}^l(\boldsymbol{\alpha}) = \frac{1}{M}\sum_{j=1}^{M}\mathcal{C}(\boldsymbol{\sigma}^{jl}_A)$
14:     $l' = \underset{l=1,2,3,4}{\mathrm{argmin}}\left[1 - \mathcal{C}^l(\boldsymbol{\alpha})\right]$
15:     $\bar{f}(\boldsymbol{\alpha}) = 1 - \mathcal{C}^{l'}(\boldsymbol{\alpha})$
16:     $\boldsymbol{\alpha}^* = \mathrm{OPT}(\bar{f}(\boldsymbol{\alpha}), \nabla_{\boldsymbol{\alpha}}\bar{f}(\boldsymbol{\alpha}))$
17:     for $j = 1$ to $M$ do
18:         $\boldsymbol{\rho}_j = \boldsymbol{\sigma}^{jl'}_A(\boldsymbol{\alpha}^*) \otimes \boldsymbol{\sigma}^{jl'}_A(\boldsymbol{\alpha}^*)$
19:     end for
20: end for
```

In this context, we have an output concurrence $\mathcal{C}'(\boldsymbol{\alpha}, \boldsymbol{x})$ depending on both the two-qubit gate and the input state. Therefore, we are going to use an average cost function

$$\bar{f}(\boldsymbol{\alpha}) = 1 - \int_{\mathrm{supp}(p)} \mathcal{C}'(\boldsymbol{\alpha}, \boldsymbol{x}) p(\boldsymbol{x})\, d\boldsymbol{x} \tag{6.9}$$

in the L-BFGS algorithm. The integral can be either solved numerically or approximated via a quasi-Monte Carlo approach by sampling $\boldsymbol{x}$ over its corresponding parameter distribution $p(\boldsymbol{x})$:

$$\bar{f}(\boldsymbol{\alpha}) = 1 - \underset{\boldsymbol{x}\sim p(\boldsymbol{x})}{\mathbb{E}}\left[\mathcal{C}'(\boldsymbol{\alpha}, \boldsymbol{x})\right]. \tag{6.10}$$

We chose the second approach since it proved to be precise enough, i.e.,

$$\frac{1}{M}\sum_{j=1}^{M} p(x_j) \approx 1,$$

with M being the sample size, and numerically faster. A summary of the optimization routine can be seen in Algorithm 5. We first sample a batch of random density matrices by sampling over the parameter space according to a probability distribution $p(\boldsymbol{x})$. Then we optimize the average output concurrence of the protocol as a function of the parameters $\boldsymbol{\alpha}$. The optimization routine provides us with output density matrices, which are re-inserted in the protocol for a successive purification round, controlled by a different unitary matrix $U(\boldsymbol{\alpha})$. This is also optimized, giving rise to a loop that breaks when the desired concurrence level or when the maximal number of iterations N_{max} is reached. We would like to highlight that the unitary matrices which build the optimized purification protocol are different from each other. Each one of them is optimized for its own purification step, a procedure that can be conceived as a form of adaptive purification. Due to the nature of the optimization, and the intrinsic complexity of differentiating, e.g., the concurrence, it is likely that some of the concurrence values output by our optimization routine are not true optima, but rather local optima. However, in general, restarting the algorithm multiple times with different initial conditions can help reduce the probability of it being stuck in a local minimum.
Now, we shed light on the meaning of the average cost function through the following two examples. We employ the CNOT gate

$$U_{\text{CNOT}} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} = e^{i\frac{3\pi}{4}} \times \underbrace{U'}_{\in SU(4)},$$

where U' in the Euler angle parametrization is given by setting

$$\alpha_3 = \alpha_5 = \alpha_7 = \frac{\pi}{4},$$
$$\alpha_4 = \alpha_6 = \alpha_{10} = \frac{\pi}{2},$$

and the remaining nine angles to zero in Eq. (6.6). By fixing the vector $\boldsymbol{\alpha}$, we are able to get a value for the average cost function of the CNOT gate and thus to evaluate its performance.
We remark that for the following three examples, we merely calculate the cost function of the CNOT gate in order to explain this particular process, and to show that this gate can be optimal in certain cases with a possible combination of other single-qubit gates. In Sec. 6.3 we run an optimization process where the cost function is evaluated for many two-qubit gates in order to find the optimal one.

Example II.1. Let us consider the state

$$\frac{1}{6}\begin{pmatrix} 1+2x & 0 & 0 & 1-4x \\ 0 & 2-2x & 0 & 0 \\ 0 & 0 & 2-2x & 0 \\ 1-4x & 0 & 0 & 1+2x \end{pmatrix}, \quad \text{with} \quad x \in [0,1] \tag{6.11}$$

subject to the purification protocol with CNOT gates. This state is a rotated Werner state [Wer89] which was employed in the seminal protocol of Ref. [BBP&al96] and its concurrence reads

$$\mathcal{C}(x) = \begin{cases} 2x-1, & x \in (0.5,1] \\ 0, & x \in [0,0.5]. \end{cases}$$

The output reads

$$\mathcal{C}'_{\text{CNOT}}(x) = \frac{3(4x^2-1)}{5-4x+8x^2} \quad \text{for} \quad x \in (0.5,1],$$

with success probability

$$P_{\text{CNOT}} = \frac{5-4x+8x^2}{9}.$$

For the sake of simplicity, we consider a uniform PDF $p(x)$ with $\text{supp}(p) = (0.5,1]$. Hence, the input average cost function reads

$$\bar{f}_{\text{input}} = 1 - \int_{0.5}^{1} \mathcal{C}(x)p(x)\,dx = 0.5$$

and the application of the purification protocol with CNOT gates yields

$$\bar{f}_{\text{CNOT}} = 1 - \int_{0.5}^{1} \mathcal{C}'(x)p(x)\,dx = 0.450103.$$

The result shows that the protocol with two identical copies of states allows us to increase on average the entanglement of the output states.

Example II.2. Now, we consider the state

$$\begin{pmatrix} \frac{x}{2} & 0 & 0 & -\frac{x}{2} \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1-x & 0 \\ -\frac{x}{2} & 0 & 0 & \frac{x}{2} \end{pmatrix}, \quad \text{with} \quad x \in [0,1]. \tag{6.12}$$

Restricted to the interval $x \in [2/3,1]$ it corresponds to a maximally entangled mixed state [IH00; MJW&al01]. In general for every $x \in (0,1]$, this state with concurrence $\mathcal{C}(x) = x$ is perfectly purifiable in just one iteration of the protocol [TB16]. We can find after one iteration that the concurrence becomes

$$\mathcal{C}'_{\text{CNOT}}(x) = 1,$$

with success probability

$$P_{\mathrm{CNOT}} = \frac{x^2}{2}.$$

It is immediate from Eq. (6.9) that for any PDF with $\mathrm{supp}(p) = [0,1]$,

$$\bar{f}_{\mathrm{CNOT}} = 1 - \int_0^1 \mathcal{C}'(x) p(x)\, dx = 0.$$

This means that the CNOT gate is optimal for this family of states.

Example II.3. As the last example let us consider the initial state

$$x\,|\Phi^+\rangle\langle\Phi^+| + (1-x)\,|\Phi^-\rangle\langle\Phi^-|, \quad \text{with} \quad x \in [0,1], \tag{6.13}$$

and concurrence $\mathcal{C}(x) = |1-2x|$. After one iteration of the purification protocol with bilaterally applied CNOT gates, it is not hard to realize that the resulting state has the concurrence

$$\mathcal{C}'_{\mathrm{CNOT}}(x) = (1-2x)^2$$

with success probability $P_{\mathrm{CNOT}} = 1$. The output concurrence $\mathcal{C}'_{\mathrm{CNOT}}(x)$ is less than or equal to $\mathcal{C}(x)$ for all $x \in [0,1]$. Thus, we conclude by using the properties of concurrence and integration that

$$\int_0^1 \mathcal{C}'_{\mathrm{CNOT}}(x) p(x)\, dx \leqslant \int_0^1 \mathcal{C}(x) p(x)\, dx \tag{6.14}$$

for any PDF with $\mathrm{supp}(p) = [0,1]$. Hence, the initial average cost function $\bar{f}_{input}$ is always less than or equal to $\bar{f}_{CNOT}$. This is an example where the purification fails with the sole implementation of the CNOT gate. It should be noted that for $x \neq 0.5$, the state in this example can be purified with previous protocols [BBP&al96; DEJ&al96] that work on the Bell basis, and where the implementation with the CNOT gate is now accompanied by local single-qubit gates.

These examples demonstrate the meaning of the average cost function. It is obvious that one or more two-qubit gates can be optimal for certain family of states and less optimal for others. In the subsequent section, we will investigate numerically several cases.

6.3 Results

In this section it is demonstrated how our proposed method can find optimal purification schemes. Our first case is the continuation of *Example II.1.* in Sec. 6.2. We have seen so far that the CNOT gate in $N = 1$ purification round can reduce the average cost function $\bar{f}$ approximately by 0.05. The simulation results with the input state in Eq. (6.11) show that the optimal $SU(4)$ gate for $N = 1$ has a similar improvement on $\bar{f}$ as the CNOT gate (see Fig. 6.2). The optimal gates found for $N = 2$ and 3 can further reduce $\bar{f}$; however, it is easy to check that the CNOT gate alone is not optimal anymore for these rounds of iterations. In Fig. 6.2 it is also shown that a higher number of iterations results

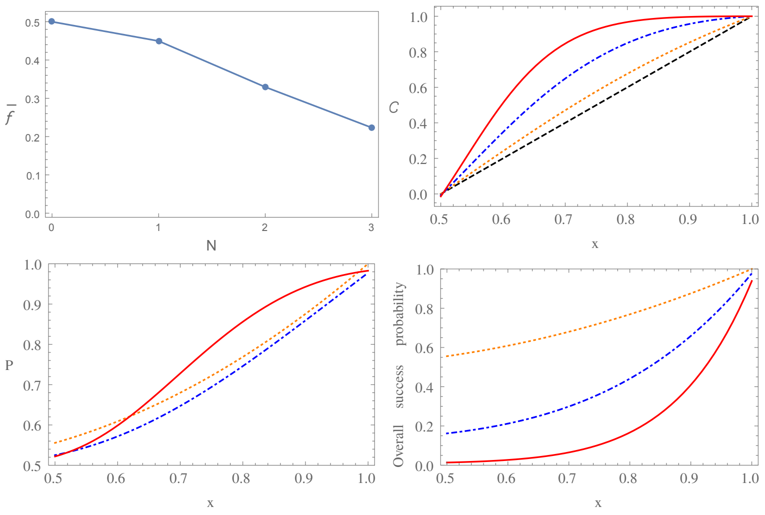


Figure 6.2: Optimized purification protocol for the family of states in Eq. (6.11). Top left panel: Average cost function $\bar{f}$ with a uniform PDF as a function of N, the number of iterations. Top right panel: Concurrence $\mathcal{C}$ as a function of x. Four curves are presented for different values of iterations N: 0 or the input concurrence (black dashed line), 1 (orange dotted line), 2 (blue dash-dotted line), and 3 (red solid line). Bottom left panel: Success probabilities as a function of x for different values of iterations N: 1 (orange dotted line), 2 (blue dash-dotted line), and 3 (red solid line). Bottom right panel: The overall success probability after $N = 1$ (orange dotted line), $N = 2$ (blue dash-dotted line), and $N = 3$ (red solid line) iterations as a function of x. The sample size has been set to $M = 1000$.

in more concave shapes of the corresponding concurrences. It is worth noting that the success probability of the second iteration is lower than the success probability of the first iteration. The overall success probability of three iterations, displayed also in Fig. 6.2, is defined as follows: In the first iteration four qubit pairs, in the second iteration two qubit pairs, and in the third iteration the final qubit pair are successfully purified. These results show that states close to maximally entangled states can be produced already in the third iteration, but the overall success probability of the procedure is getting smaller with the number of iterations. The only fixed point is $\mathcal{C} = 1$. Thus our method seems to provide the same success probabilities as the ones found in [BBP&al96]. As a result, we have investigated the circumstances where the CNOT gate is also optimal. It turns out that the local unitary transformation introduced by [DEJ&al96] and given by

$$b^\dagger_{A_1} \otimes b^\dagger_{A_2} \otimes b_{B_1} \otimes b_{B_2},$$

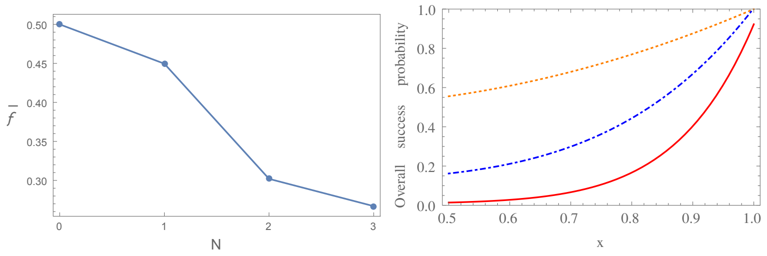


Figure 6.3: Optimized purification protocol for Werner states, see Eq. (6.8). Left panel: Average cost function $\bar{f}$ with a uniform PDF as a function of N, the number of iterations. Right panel: The overall success probability after $N = 1$ (orange dotted line), $N = 2$ (blue dash-dotted line), and $N = 3$ (red solid line) iterations as a function of x. The sample size has been set to $M = 1000$.

where

$$b = \frac{\mathbb{I}_2 + i\sigma_x}{\sqrt{2}}, \tag{6.15}$$

with the Pauli matrix σ_x and the identity map $\mathbb{I}_2$, is crucial for the CNOT gate. This is the transformation, which transforms a Werner state in Eq. (6.8) into the state in Eq. (6.11). Now if one applies Eq. (6.15) before all iterations of the protocol involving only the CNOT gate, then the same curves are obtained as in Fig. 6.2. This means that there are more optimal protocols which yield the same results and our approach can find them.

The one-parameter family of states in Eq. (6.11) has the same concurrence as the Werner state. Therefore, we consider the Werner state to be our next application. In Fig. 6.3 numerical results are presented for the average cost function $\bar{f}$ and the overall success probability which exhibit the same behavior found for the one-parameter family of states in Eq. (6.11). In this case, one can note an improvement of 0.04 in $\bar{f}$ after $N = 3$ iterations. This is less than the previously obtained value of 0.09 shown in Fig. 6.2. Furthermore, this is accompanied by another numerical inaccuracy: The overall success probability at $\mathcal{C} = 1$ is less than one. Given these results, the proposed method can find optimal entangling two-qubit gates for at least three iterations. It is also clear from these tests that the algorithm is always reducing $\bar{f}$, but from $N = 3$ iterations this might not be an optimal improvement of the concurrence. This originates from the fact that the gradient $\nabla f(\boldsymbol{\alpha})$ [see Eq. (6.7)] is almost flat in the neighborhood of $\mathcal{C} = 1$ for $N > 2$ iterations and thus the numerical search for the optimal gate, i.e, the search for $\boldsymbol{\alpha}^* \in \mathbb{R}^{15}$, becomes inefficient.

Now, to test our method further, let us consider another state whose entanglement purification procedure is known. For this purpose we note that the state of *Example II.2.* in Eq. (6.12) can be transformed using a separable gate $b \otimes b^\dagger$,with b defined in

Eq. (6.15), into the state

$$x\,|\Psi^-\rangle\langle\Psi^-| + (1-x)\,|\Upsilon\rangle\langle\Upsilon|\,, \tag{6.16}$$

with $x \in [0,1]$ and

$$|\Upsilon\rangle = \frac{1}{\sqrt{2}}\left(|\Psi^+\rangle + i\,|\Phi^-\rangle\right).$$

Using the unitary transformation $b^\dagger \otimes b$ on Eq. (6.16), one obtains the state in Eq. (6.12) which can be purified in one iteration using bilateral CNOT gates. For this reason, it is expected that the optimal two-qubit gate that purifies the states in Eq. (6.16) is the one that achieves the task in one iteration and given by a CNOT combined with single qubit gates in the form of b. As both states are connected via a separable gate, they have the same concurrence $\mathcal{C}(x) = x$. However, it is important to note that the CNOT gate is not optimal for the state in Eq. (6.16), because after one iteration round

$$\mathcal{C}'_{\text{CNOT}}(x) = \begin{cases} 0, & x \in [0, 0.5] \\ 2x, \frac{2x-1}{1+x^2} & x \in (0.5, 1], \end{cases}$$

and $\mathcal{C}'_{\text{CNOT}}(x) \leqslant x$. The success probability is

$$P_{\text{CNOT}} = \frac{1+x^2}{2}.$$

Thus, the CNOT gate without the non-symmetrical local transformation of Eq. (6.15) impairs the concurrence. In contrast to this analytical observation, numerical analysis with a uniform PDF yields already in the first iteration for both states an average cost function $\bar{f} \approx 0.0002$. To demonstrate the robustness of the numerical approach we provide examples of three non-uniform PDFs for $N = 1$ iteration. First, we consider

$$p(x) = 2x, \quad \text{with} \quad x \in [0,1],$$

which describes a situation, where states with higher concurrences are more likely to be subject to the purification. The resulting average cost function is $\bar{f} \approx 0.000004$. Second, we take

$$p(x) = 2(1-x) \quad \text{with} \quad x \in [0,1],$$

which puts more weight on states with low concurrences and find $\bar{f} \approx 0.0005$. Finally, we investigate a PDF

$$p(x) = 6x(1-x) \quad \text{with} \quad x \in [0,1],$$

i.e., the states around the concurrence $\mathcal{C}(x) = 0.5$ are more likely to participate in the purification, and obtain $\bar{f} \approx 0.00005$. These results demonstrate the effectiveness of our approach and up to a numerical precision these one-parameter families of states can be purified in one iteration.

Next we consider the following two-parameter family of states arising from a generation

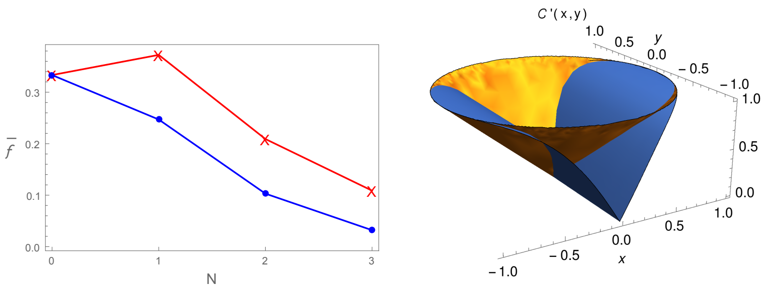


Figure 6.4: Optimized purification protocol for the state in Eq. (6.17) for 1000 parameter samples. Left panel: Average cost function $\bar{f}$ with a uniform PDF as a function of N, the number of iterations. Crosses are the results of the CNOT gate, whereas circles display the numerical optimization. They have been connected by lines to guide the eye. Right panel: Concurrence $\mathcal{C}'$ as a function of x and y after one purification round. The cone-type surface is obtained with our approach. A less optimal surface with minima at $x = 0$ and along the y axis is the result of the purification protocol with the CNOT gate. The sample size has been set to $M = 1000$.

of distant entanglement in the context of a hybrid quantum repeater [Ber17],

$$\rho(x,y) = \frac{1}{4}\begin{pmatrix} 1-x & iy & -iy & x-1 \\ -iy & x+1 & -x-1 & iy \\ iy & -x-1 & x+1 & -iy \\ x-1 & -iy & iy & 1-x \end{pmatrix}. \tag{6.17}$$

Here $x, y \in \mathbb{R}$ and $x^2 + y^2 \leqslant 1$. The concurrence of this state is $\sqrt{x^2+y^2}$. In order to relate the performance of our approach, we apply the CNOT gate based purification protocol to this family of states. We obtain

$$\mathcal{C}'_{\text{CNOT}}(x,y) = \frac{2|x|}{1+x^2}, \tag{6.18}$$
$$P_{\text{CNOT}} = \frac{1+x^2}{2} \tag{6.19}$$

after $N = 1$ and

$$\mathcal{C}'_{\text{CNOT}}(x,y) = \frac{4|x|(1+x^2)}{1+6x^2+x^4}, \tag{6.20}$$
$$P_{\text{CNOT}} = \frac{1+6x^2+x^4}{2(1+x^2)^2} \tag{6.21}$$

after $N = 2$ purification rounds. If one applies the unitary transformation in Eq. (6.15) on the state before the bilateral CNOT gates are performed, then the above results

remain unchanged. These results demonstrate that the CNOT gate and the additional tricks, which have yielded optimal purifications for the one-parameter family of states, are not optimal in this scenario, since they are outperformed on average by our optimized protocol.
Thus, the original CNOT-based purification protocols cannot exploit all the useful entanglement, because the concurrence and the success probability become independent of the y variable. Using these analytical results and the uniform PDF

$$p(x,y) = \frac{1}{\pi} \quad \text{with} \quad x^2 + y^2 \leqslant 1,$$

we compare our approach with the above-presented analytical results [see Eq. (6.18) and Eq. (6.20)]. In Fig. 6.4 we show that our optimized protocol provides after one iteration an x- and y-dependent concurrence. Although the numerically obtained concurrence is larger at $x = 0$ and $y \in [-1,1]$ than the surface given by Eq. (6.18), one can also observe the opposite at $y = 0$ and $x \in [-1,1]$. However, the concurrence has more improvement with our algorithm and the average cost function with the uniform PDF stays, for more iterations, lower than the original CNOT-based protocols [see the left panel in Fig. 6.4]. In regard to the overall success probability, we let our algorithm work until the third iteration and in Fig. 6.5 the results are compared with the CNOT-based purification protocols. The obtained surfaces differ only by a $\pi/2$ rotation around the z axis. Therefore, there is not much difference in the overall success probability and from this aspect the CNOT gate can also be considered optimal, though it still cannot improve the concurrence as effectively as the gate obtained with our approach. However, one might think that with a proper choice of local unitary transformations, like the transformation in Eq. (6.15) used for Werner and one-step purifiable states, the application of the CNOT gate may result in an optimal purification protocol. Let us consider then two general local unitary transformations at locations A and B for the preparation of two-qubit states. It is enough to consider unitary transformations in $SU(2)$, for which the Euler angle parametrization reads [TS02]

$$U_A = e^{i\sigma_z\alpha_1}e^{i\sigma_y\alpha_2}e^{i\sigma_z\alpha_3}, \tag{6.22}$$

$$U_B = e^{i\sigma_z\beta_1}e^{i\sigma_y\beta_2}e^{i\sigma_z\beta_3}, \tag{6.23}$$

where σ_y and σ_z are the Pauli matrices. Here $\alpha_1,\beta_1 \in [0,\pi]$, $\alpha_2,\beta_2 \in [0,\pi/2]$, and $\alpha_3,\beta_3 \in [0,2\pi]$ are the Euler angles for $SU(2)$. For the first iteration, we analyze the output unitary of the optimized protocol with *Mathematica* [Inc] and observe that the angles α_1 and β_1 do not affect the effectiveness of the CNOT-based purification protocol. Optimal concurrences for the remaining four angles yield either the result in Eq. (6.18) or

$$\mathcal{C}'_{\text{CNOT}}(x,y) = \frac{2|y|}{1+y^2}, \tag{6.24}$$

with success probability

$$P_{\text{CNOT}} = \frac{1+y^2}{2}, \tag{6.25}$$

for $\alpha_2 = \frac{\pi}{8}$, $\beta_2 = \frac{3\pi}{8}$, and $\alpha_3 = \beta_3 = \frac{3\pi}{4}$. For these results the average cost function with a uniform PDF yields $\bar{f} \approx 0.372$, which is still lower than the average concurrence found by our approach. These results demonstrate that even with local unitary transformations the CNOT gate is globally not optimal for all values of x and y. However, when x and y are known beforehand, one can combine the CNOT gate together with the local unitary transformation

$$U_A^\dagger \otimes U_B, \quad \text{with} \quad U = \cos(\theta)\mathbb{I}_2 + \sin(\theta)\sigma_x, \tag{6.26}$$

where the angle θ is a function of x and y. For example, for $x, y \geqslant 0$, we have

$$\cos(\theta) = \sqrt{\frac{1}{2} + \sqrt{\frac{x + \sqrt{x^2 + y^2}}{8\sqrt{x^2 + y^2}}}}.$$

Using this transformation before the purification protocol, we obtain the state

$$\frac{1 + \sqrt{x^2 + y^2}}{2} |\Psi^-\rangle\langle\Psi^-| + \frac{1 - \sqrt{x^2 + y^2}}{2} |\Phi^-\rangle\langle\Phi^-|. \tag{6.27}$$

Now, together with the single qubit gates in Eq. (6.26), the CNOT-based purification yields, after one iteration, the concurrence

$$\mathcal{C}'_{\text{CNOT}}(x, y) = \frac{2\sqrt{x^2 + y^2}}{1 + x^2 + y^2}, \tag{6.28}$$

with success probability

$$P_{\text{CNOT}} = \frac{1 + x^2 + y^2}{2}. \tag{6.29}$$

It is immediate that $\mathcal{C}'_{\text{CNOT}}(x, y) \geqslant \sqrt{x^2 + y^2}$, i.e., we are improving the concurrence. In this particular case, we do not need the average cost function in the numerical search, because the state is fixed. Our algorithm running with a single state with parameters (x, y) as defined in Eq. (6.17) instead of an ensemble of states can optimally improve the concurrence, but is unable to find this particular optimal solution presented above, because the transformation (6.26) together with CNOT gates is a nonsymmetrical transformation at nodes A and B.
To show that optimization becomes more effective with an increased number of angles, we consider a less general unitary gate than the one in Eq. (6.6). This gate consists of one CNOT gate and four local unitary transformations of Eq. (6.22)

$$\begin{aligned} \tilde{U}(\boldsymbol{\alpha}') = &\left[U_1(\alpha'_1, \alpha'_2, \alpha'_3) \otimes U_2(\alpha'_4, \alpha'_5, \alpha'_6)\right] \\ &U_{\text{CNOT}}\left[U_3(\alpha'_7, \alpha'_8, \alpha'_9) \otimes U_4(\alpha'_{10}, \alpha'_{11}, \alpha'_{12})\right], \end{aligned} \tag{6.30}$$

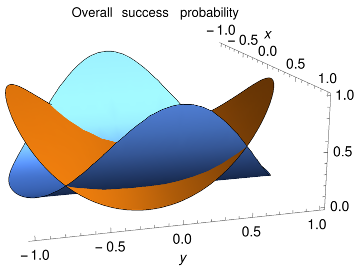


Figure 6.5: Overall success probability of the state in Eq. (6.17) after $N = 3$ iterations as a function of x and y. Both surfaces are similar in appearance. The one having maxima at $x = 1$ belongs to the CNOT-based protocol and the other one having maxima at $y = 1$ is obtained via the optimized numerical protocol.

N	$U(\boldsymbol{\alpha}^*)$	$\tilde{U}(\boldsymbol{\alpha}'^*)$
0	0.666	0.666
1	0.753	0.751
2	0.897	0.798
3	0.968	0.937

Table 6.1: Comparison of the average concurrences produced with different parametrizations of unitary matrices given in Eqs. (6.6) and (6.31) for $M = 1000$. The values in columns 2 and 3 represent the two best sequences found by the algorithm among ten different runs. We observe that the two different parametrizations provide us with improving concurrences. Nonetheless, the general two-qubit gate seems to perform slightly better.

i.e., the quantum gate has a clear quantum circuit representation. Now $\boldsymbol{\alpha}'$ is a 12-dimensional vector of the angles. We have inserted this gate into our algorithm and observe that the results are almost as good as the optimization of a general gate with 15 parameters (see Table 6.1). Furthermore, the two optimal gates after $N = 1$ iteration are presented in Fig. 6.6.
Finally, let us point out that our approach does not take into account operational or memory errors. These always depend on the implementation and if an experiment can provide us models for the errors then our approach can be easily extended. Another important experimental input is the PDF $p(\boldsymbol{x})$, which is usually subject to the method of generating entangled states between locations A and B. Furthermore, this PDF is assigned to the process of choosing a value $\boldsymbol{x}$ as a random event, i.e, we have the same two two-qubit pairs parametrized by $\boldsymbol{x}$ before the purification protocol.
In effect, we integrate away the parameter dependence of the states and thus our approach yields an optimal two-qubit gate on average. If the value of $\boldsymbol{x}$ is fixed in an experimental design, then our method provides an optimal two-qubit gate designed for

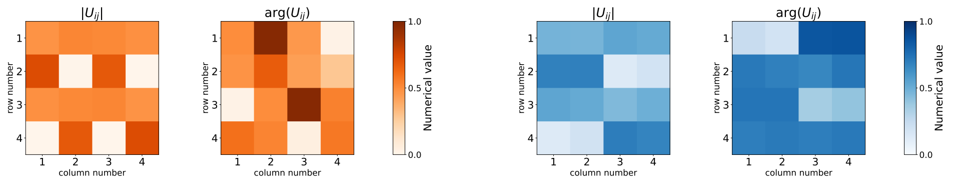


Figure 6.6: Optimized unitaries $U_N(\boldsymbol{\alpha}^*)$ – the Euler angle parametrization in Eq. (6.6) (left two color plots) – and $\tilde{U}_N(\boldsymbol{\alpha}'^*)$ – the CNOT circuit given in Eq. (6.30) (right two color plots) – for $N = 1$. The optimal protocols are obtained for the state given in Eq. (6.17) with $M = 1000$ and the procedure discussed in Algorithm 5. For each unitary, the first plot represents the absolute value of the matrix entries taken in the computational basis, whereas the second one represents the phase of the same entries divided by 2π. The color map describes values varying from 0 (white) to 1 (dark orange and dark blue).

this particular state. The optimal two-qubit gates can always be realized by three CNOT gates and additional single-qubit gates [VD04; VW04], and therefore the experimental generation of this gate is possible with high fidelity [Deb&al16]. Also in the context of trapped ions or superconducting quantum circuits, the generation of two-qubit entangling gates can be achieved with high precision using the Mølmer-Sørensen gate [SM99] or the $\sqrt{i\text{SWAP}}$ [FRS05] gate. Since two-qubit gates have a straightforward implementation in many physical settings, due to quantum compilation [MMN&al16], we argue that quantum computing and communication platforms could actually benefit from globally optimal gates for entanglement purification.

6.4 Conclusions

In the context of entanglement purification and recurrence protocols, we have presented a method to obtain optimal protocols. This method searches for the optimal two-qubit gate, which is applied bilaterally at the nodes A and B in order to distill from an ensemble of mixed entangled pairs a higher fidelity state with respect to a maximally entangled state. Here we assumed that the same copies of the states can be generated before the purification protocol takes place, but a different experimental run could result in different states. Errors originating from local operations, memory requirements, or even classical communication have been neglected for now.
We numerically demonstrated the optimality of our proposal for several states. In the case of the Werner state, we found that several optimal two-qubit gates and their performances are the same as in the CNOT-based Deutsch protocol [DEJ&al96]. Thus, for Werner states the optimality cannot be improved beyond the already known performance. Next we investigated a family of states, which can be purified in one step, i.e., two copies of mixed entangled states are enough to obtain a maximally entangled

state. Here we immediately obtained a minimal average cost function $\bar{f} \approx 0$, as expected. Finally, we considered a two-parameter family of states which is obtained in theoretical models of a quantum repeater [Ber17]. Our numerical investigation demonstrated that a single bilaterally applied CNOT gate cannot be globally optimal for these states. On the other hand, when the state is known beforehand, then parameter-dependent, non-symmetric local transformations allow again the CNOT gate to be also one of the optimal two-qubit gates. This case exemplifies the difference between protocols that are optimal for a full class of parametrized states and those that are only optimal for a single state. We also investigated with our algorithm a concrete quantum circuit consisting of four different single-qubit gates and a CNOT gate, which is only a subset of the $SU(4)$ group. We found that the optimal two-qubit gate among all elements of this quantum circuit seems to be slightly worse than the one found among all matrices in $SU(4)$. This suggests that the search after an ensemble of optimal two-qubit gates can benefit from a general parametrization.
In conclusion, we have proposed a general method to optimize entanglement purification for an arbitrary family of states. Our algorithm [Pre22] can find the two-qubit gates that on average induce the highest increase of entanglement. We remark that the method is general for the set of parameters defining the states. Optimizing for a single state is also possible, as shown in one of the presented examples. Furthermore, we focused on the degree of entanglement measured by the concurrence and not on the fidelity with respect to a particular Bell state. This is motivated by the fact that a general entanglement purification protocol may not always purify towards a given Bell state, but rather a maximally entangled state. As we optimize among many different protocols, it is necessary to use a measure for which all maximally entangled states are equivalent. A possible drawback is that one cannot know with certainty the final maximally entangled state produced by the protocol. This and other issues such as different input states, non-symmetric two-qubit gates, and nonideal local operations remain open questions. However, this work aims to introduce a concept of globally optimized recurrence protocols that is flexible enough to incorporate the aforementioned points in further investigations.

6.5 Acknowledgments

We thank Michael Schilling for his help with the plots. This work was supported by the DFG under Germany's Excellence Strategy, Cluster of Excellence Matter and Light for Quantum Computing (ML4Q) Grant No. EXC 2004/1-390534769, the German Ministry for Education and Research, under QSolid, Grant no. 13N16149, and AIDAS - AI, Data Analytics and Scalable Simulation - which is a Joint Virtual Laboratory gathering the Forschungszentrum Jülich (FZJ) and the French Alternative Energies and Atomic Energy Commission (CEA). The results were obtained using the libraries JAX [BFH&al18], Scipy [Vir&al20] and QuTiP [JNN13].

6.6 Appendix

6.6.1 Gell-Mann type basis

In this appendix, details concerning one of the possible bases of the Lie algebra of $SU(4)$ are shown. The matrices read [TBS02]:

$$\sigma_1 = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}, \quad \sigma_2 = \begin{pmatrix} 0 & -i & 0 & 0 \\ i & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}, \quad \sigma_3 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix},$$
$$\sigma_4 = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}, \quad \sigma_5 = \begin{pmatrix} 0 & 0 & -i & 0 \\ 0 & 0 & 0 & 0 \\ i & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}, \quad \sigma_6 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix},$$
$$\sigma_7 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & -i & 0 \\ 0 & i & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}, \quad \sigma_8 = \frac{1}{\sqrt{3}}\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -2 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}, \quad \sigma_9 = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix},$$
$$\sigma_{10} = \begin{pmatrix} 0 & 0 & 0 & -i \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ i & 0 & 0 & 0 \end{pmatrix}, \quad \sigma_{11} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}, \quad \sigma_{12} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -i \\ 0 & 0 & 0 & 0 \\ 0 & i & 0 & 0 \end{pmatrix},$$
$$\sigma_{13} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}, \quad \sigma_{14} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -i \\ 0 & 0 & i & 0 \end{pmatrix}, \quad \sigma_{15} = \frac{1}{\sqrt{6}}\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -3 \end{pmatrix}. \tag{6.31}$$

Actually, these matrices together with the identity matrix

$$\sigma_0 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix},$$

form an orthogonal basis for the space $M_n(\mathbb{C})$ of 4×4 matrices with complex entries equipped with the Hilbert-Schmidt scalar product

$$\langle A, B \rangle = \mathrm{Tr}\{A^\dagger B\}, \quad A, B \in M_n(\mathbb{C}),$$

where $A^\dagger$ is the adjoint of A. We first note that $\sigma_i^\dagger = \sigma_i$ for all $i \in \{0, 1, 2, \dots, 15\}$. Therefore, every matrix $X \in M_n(\mathbb{C})$ can be written as

$$X = \sum_{i=0}^{15} \frac{\langle \sigma_i, X \rangle}{\langle \sigma_i, \sigma_i \rangle} \sigma_i, \quad X^\dagger = \sum_{i=0}^{15} \frac{\langle \sigma_i, X \rangle^*}{\langle \sigma_i, \sigma_i \rangle} \sigma_i,$$

where z^* is the complex conjugate of $z \in \mathbb{C}$ [SBM&al21]. In this fashion, we can obtain another representation for any element $U \in SU(4)$ fulfilling $U^\dagger U = UU^\dagger = \sigma_0$ and including also that its determinant is equal to one. This is also a minimal parametrization, however a cumbersome one compared to the Euler angle parametrization. In general, a Gell-Mann-type basis for the Lie algebra of $SU(n)$ can always be obtained, see Refs. [TS02; BCC06].

7 Statistical evaluation and optimization of entanglement purification protocols

Disclaimer: A modified version of this chapter was published in Ref. [PB24]. FP and JZB developed the theoretical framework of the paper. FP developed the code based on the sampling method developed in [SBM&al21] and ran all the simulations. JZB and FP wrote the theoretical introduction of the manuscript together. FP composed all the plots and wrote the discussion of the results and the description of the numerical methods. Both authors worked together on the revision of the manuscript.

Quantitative characterization of two-qubit entanglement purification protocols is introduced. Our approach is based on the concurrence and the hit-and-run algorithm applied to the convex set of all two-qubit states. We demonstrate that pioneering protocols are unable to improve the estimated initial average concurrence of almost uniformly sampled density matrices, however, as it is known, they still generate pairs of qubits in a state that is close to a Bell state. We also develop a more efficient protocol and investigate it numerically together with a recent proposal based on an entangling rank-2 projector. Furthermore, we present a class of variational purification protocols with continuous parameters and optimize their output concurrence. These optimized algorithms turn out to surpass former proposals and our protocol by means of not wasting too many entangled states.

7.1 Introduction

Entanglement purification protocols aim to overcome the destructive effects of non-ideal channels by generating highly entangled states from a large number of noisy entangled states [DB07]. In this approach, the almost perfectly entangled pairs obtained are used in quantum teleportation [BBC&al93], and thus quantum data can be transmitted across the channels. The other possible solution to this problem is to use quantum error correction [DMN13], when quantum data is sent through the channel by adding enough redundancy, e.g., increasing the number of qubits, such that the original information is recoverable even in the presence of noise. While in quantum error correction the sources of errors and their models have to be identified, the first entanglement purification protocols [BBP&al96; DEJ&al96] offer solutions for general noisy two-qubit

states. However, these protocols convert a general two-qubit state to either Werner or Bell diagonal states by using local random transformations, which waste useful entanglement [BDS&al96; HHH96]. This was found by investigating particular entangled states, whose entanglement is destroyed in the first step of the protocol, and this issue was also confirmed in a different entanglement purification approach [TB16]. A quantitative characterization of the ratio between wasted and improved entangled states is missing and this paper is devoted to investigating this question.
The quest for optimality is ongoing and important research in quantum physics ranging from experimental protocols to algorithms. In most cases, numerical optimization based on continuous and discrete parameters is carried out to enhance performances with respect to different figures of merit. This has recently led to the improvement of entanglement generation [MPK&al18], unitary compilation and state preparation [PSJ&al24], quantum error correction [MMN&al16], communication [NDD&al19; WMD&al20], and algorithms [WHT16], which are only a few examples of the vast literature. Concerning entanglement purification, discrete optimizations have been considered for Werner states [KAJ19] and we have started to investigate continuous optimization for certain families of states [PCT&al22]. Therefore, in this paper we not only evaluate some existing proposals but also search for more optimal protocols with a lower amount of wasted entangled states.
In this study, we avail ourselves of the hit-and-run algorithm to generate asymptotically and effectively uniformly distributed two-qubit density matrices [SBM&al21]. It is known that approximately 24.24 % of the generated two-qubit states are separable, a numerical result that has also been confirmed by other methods [SD12; SSN&al15b; MS14; FJ16]. Therefore, almost one-fourth of the states are immediately useless in an entanglement purification protocol. In the case of entangled states, we use the concurrence [Woo98] to measure the improvement or deterioration induced by a protocol. The choice of the concurrence and its advantage over the usually employed fidelity will be explained in the context of those protocols, which have two Bell states as stable fixed points. We calculate the average concurrence over the whole sample, which will be our cost function. Any change of this cost function characterizes only the whole sample of two-qubit states, whereas the concurrence of individual states may show different behaviors. The obtained estimates allow us to compare quantitatively some existing entanglement purification protocols and to search numerically for more optimal scenarios. These scenarios consist of the use of the $SU(4) \times SU(4)$ group to find optimal locally entangling gates. We assume throughout the whole paper that these operations can be performed without errors on both sides of the noisy channel.
Our search for optimal protocols is based on the limited memory Broyden-Fletcher-Goldfarb-Shanno (LBFGSB) optimization algorithm with bounds [BLN&al95; BK03]. To use this method we employ the Euler angle parametrization of the $SU(4) \times SU(4)$ group [TBS02], where the group is constructed from angles, which form a hyperrectangle in a 30-dimensional Euclidean vector spaces. Thus, every vector represents an element of the group and its neighborhood is defined by the Euclidean norm. The resulting parametrization allows us to construct a variational cost function that depends on a unitary in $SU(4) \times SU(4)$ and that can be minimized with the LBFGSB method due

to automatic differentiation [BFH&al18]. With this strategy, we demonstrate that it is possible to increase the percentage of entangled states purified by computer-designed protocol and achieve better performances. Furthermore, our approach introduces a systematic and general way to compare different proposals for entanglement purification protocols, where more realistic experimental considerations can be incorporated upon request.

The paper is organized as follows. In Sec. 7.2, we recall the mathematical description of some known entanglement purification protocols. Furthermore, we present a controlled-NOT(CNOT) gate-based protocol and also our variational approach. A brief description of the numerical approach is given in Sec. 7.3. Numerical results are presented and discussed in Sec. 7.4. Section 7.5 contains a summary and our conclusions. Some technical details are provided in the Appendices.

7.2 Entanglement purification

Entanglement purification describes a protocol between two nodes of a quantum network with the task of extracting highly entangled states, e.g., Bell states, from arbitrarily entangled states. There are several categories of purification protocols, based on the way the entangled states are distilled: filtering, recurrence, hashing, and breeding protocols [DB07]. Here, our focus lies on bipartite recurrence protocols, where the two nodes A and B initially share a pair of identical two-qubit states:

$$\varrho = \rho^{A_1,B_1} \otimes \rho^{A_2,B_2}. \tag{7.1}$$

The goal is to increase the entanglement of one of the pairs by performing local entangling operations and measurements in the nodes. Finally, classical communication between A and B is used. In this paper, we consider two-qubit states in the representation:

$$\rho = \sum_{i,j=1}^{4} r_{ij} \left|i\right\rangle \left\langle j\right|, \tag{7.2}$$

with the Bell states

$$\begin{aligned} \left|1\right\rangle &= \frac{1}{\sqrt{2}}\left(\left|01\right\rangle - \left|10\right\rangle\right), \quad \left|2\right\rangle = \frac{1}{\sqrt{2}}\left(\left|01\right\rangle + \left|10\right\rangle\right), \\ \left|3\right\rangle &= \frac{1}{\sqrt{2}}\left(\left|00\right\rangle - \left|11\right\rangle\right), \quad \left|4\right\rangle = \frac{1}{\sqrt{2}}\left(\left|00\right\rangle + \left|11\right\rangle\right). \end{aligned}$$

Furthermore, we use the notation $r_j = r_{jj}$. The properties $\mathrm{Tr}\{\rho\} = 1$ and $\rho^\dagger = \rho$ yield the following relations:

$$r_1 + r_2 + r_3 + r_4 = 1, \quad r_{ij} = \left(r_{ji}\right)^*. \tag{7.3}$$

In the following, we give a short overview of three purification protocols and introduce a CNOT-based approach and our variational method for the search of more optimal strategies.

7.2.1 Bennett protocol

The seminal protocol introduced in Ref. [BBP&al96] is based on the CNOT gate and allows one to distill the Bell state $|1\rangle$ from a large ensemble of initial two-qubit states ρ. In fact, the state $|1\rangle$ is only reached in the asymptotic limit of the purification rounds. The protocol operates only on Werner states [Wer89]

$$\rho_W = r_1 |1\rangle\langle 1| + \frac{1-r_1}{3}(I_4 - |1\rangle\langle 1|), \tag{7.4}$$

where I_4 is the 4×4 identity matrix. The transformation of a general state ρ of Eq. (7.2) into a Werner state ρ_W can be achieved by local random unitary rotations [BDS&al96], i.e., the so-called twirling operation, which is given by

$$\rho_W = \frac{1}{12}\sum_{j=1}^{3} K_j^\dagger \left(\sum_{i=1}^{4} K_i^\dagger K_i^\dagger \rho K_i K_i \right) K_j \tag{7.5}$$

with the transformations

$$\begin{aligned} K_j &= u_j^A \otimes u_j^B, \quad u_1 = \frac{I_2 + i\sigma_x}{\sqrt{2}}, \quad u_2 = \frac{I_2 - i\sigma_y}{\sqrt{2}}, \\ u_3 &= i|0\rangle\langle 0| + |1\rangle\langle 1|, \quad u_4 = I_2, \end{aligned} \tag{7.6}$$

the Pauli matrices σ_x and σ_y, and the 2×2 identity matrix I_2. The protocol consists of the following steps:

- Bring the state ρ into Werner form by using Eq. (7.5).
- Apply local σ_y rotations on qubits A_1 and A_2.
- Perform the bilateral operation $U_{CNOT}^{A_1\to A_2} \otimes U_{CNOT}^{B_1\to B_2}$.
- Measure the target pair (A_2, B_2) in the eigenbasis of the Pauli matrix σ_z with corresponding results (m, n), where $m, n \in \{0, 1\}$. Keep the pair (A_1, B_1) if the measurement result is either $m = n = 0$ or $m = n = 1$ and finally perform a σ_y rotation on A_1.

An elementary step of the protocol yields ρ'_W with

$$r'_1 = \frac{1 - 2r_1 + 10r_1^2}{5 - 4r_1 + 8r_1^2} \tag{7.7}$$

and success probability $P_s = (5 - 4r_1 + 8r_1^2)/9$. The state ρ'_W becomes more entangled than ρ_W if

$$2r_1 - 1 > 0. \tag{7.8}$$

7.2.2 Deutsch protocol

This protocol is conceptually similar to the previous one and operates on Bell diagonal states [DEJ&al96]

$$\rho_B = \sum_{i=1}^{4} r_i \left|i\right\rangle \left\langle i\right| . \tag{7.9}$$

The transformation of a general state ρ of Eq. (7.2) into a Bell diagonal state reads

$$\rho_B = \frac{1}{4}\sum_{i=1}^{4} K_i^\dagger K_i^\dagger \rho K_i K_i, \tag{7.10}$$

where the K_i are given in Eq. (7.6). The Deutsch protocol can be summarized in three steps:

- Apply the unitary operation $u_1^{\dagger A_1} \otimes u_1^{\dagger A_2} \otimes u_1^{B_1} \otimes u_1^{B_2}$, see Eq. (7.6).
- Perform the bilateral operation $\hat{U}_{CNOT}^{A_1 \to A_2} \otimes \hat{U}_{CNOT}^{B_1 \to B_2}$.
- Measure the target pair (A_2, B_2) in eigenbasis of σ_z with corresponding results (m, n), where $m, n \in \{0, 1\}$. Keep the pair (A_1, B_1) if the measurement result is either $m = n = 0$ or $m = n = 1$.

After applying the purification protocol we obtain a new Bell diagonal state ρ_B', which is described by the map

$$\begin{aligned} r_1' &= \frac{2r_2r_3}{C}, \quad r_2' = \frac{r_2^2 + r_3^2}{C}, \\ r_3' &= \frac{2r_1r_4}{C}, \quad r_4' = \frac{r_1^2 + r_4^2}{C}, \end{aligned} \tag{7.11}$$

where $C = (r_1 + r_4)^2 + (r_2 + r_3)^2 = P_s$ is the success probability. Entanglement of the state ρ_B' compared to ρ_B is enhanced if

$$(2r_1 - 1)(1 - 2r_4) > 0 \quad \text{or} \quad (2r_2 - 1)(1 - 2r_3) > 0. \tag{7.12}$$

Depending on which one of the above conditions is fulfilled, the protocol distills asymptotically either $\left|4\right\rangle \left\langle 4\right|$ or $\left|2\right\rangle \left\langle 2\right|$ [Mac98], i.e, the Bell states $\left|4\right\rangle$ and $\left|2\right\rangle$.

7.2.3 Matter-field interaction-based protocol

A key step in the previous protocols is the application of an entangling unitary transformation in the nodes A and B. The motivation to choose the abstract CNOT gate comes mainly from classical computer science. However, any entangling quantum operation can serve the same purpose as shown in Ref. [BTK&al16] for a cavity QED setup, where an entangling transformation emerges from matter-field interactions and is modeled by a rank-2 projector [BTK&al16]

$$M = |1\rangle\langle 1| + |3\rangle\langle 3|. \tag{7.13}$$

The purification protocol based on this projector consists of the following steps:

- Apply M in both nodes which results in the state
$$\varrho' = \frac{\Pi\varrho\Pi^\dagger}{\mathrm{Tr}\left\{\Pi^\dagger\Pi\varrho\right\}}, \quad \Pi = M^{A_1,A_2} \otimes M^{B_1,B_2}.$$
- Measure one of the pairs, say pair (A_2, B_2), in eigenbasis of σ_z with corresponding results (m,n), where $m,n \in \{0,1\}$. This results in the state $\rho_{m,n}^{A_1,B_1}$.
- The final two-qubit state is then given by
$$\rho' = \left(v_m^{A_1} \otimes v_{n+1}^{B_1}\right)\rho_{m,n}^{A_1,B_1}\left(v_m^{A_1} \otimes v_{n+1}^{B_1}\right)^\dagger,$$
where $v_n = (i|0\rangle\langle 0| + |1\rangle\langle 1|)\sigma_x^n$.

The resulting state ρ' depends only on seven real parameters instead of fifteen and the map of the protocol reads

$$\begin{aligned} r_1' &= \frac{r_1^2 + r_3^2 - r_{13}^2 - r_{31}^2}{D}, \quad r_3' = 2\frac{r_2 r_4 - |r_{24}|^2}{D} \\ r_2' &= \frac{r_2^2 + r_4^2 - r_{24}^2 - r_{42}^2}{D}, \quad r_4' = 2\frac{r_1 r_3 - |r_{13}|^2}{D}, \\ r_{12}' &= \frac{r_{12}^2 + r_{34}^2 - r_{14}^2 - r_{32}^2}{D}, \quad r_{34}' = 2\frac{r_{21}r_{43} - r_{23}r_{41}}{D}, \\ r_{13}' &= r_{14}' = r_{23}' = r_{24}' = 0, \end{aligned} \tag{7.14}$$

where the success probability $P_s = D/2$ and

$$D = (r_1 + r_3)^2 + (r_2 + r_4)^2 - (r_{13} + r_{31})^2 - (r_{24} + r_{42})^2.$$

Due to the relations in Eq. (7.3), we also have $r_{21}' = \left(r_{12}'\right)^*$ and $r_{43}' = \left(r_{34}'\right)^*$. Therefore, r_1', r_2', r_3', r_4', r_{12}', r_{21}', r_{34}', and r_{43}' are the nonvanishing elements of ρ'. This protocol was analysed in Ref. [TB16] and it was found that either

$$(2r_1 - 1)(1 - 2r_3) > -(2\mathrm{I}m[r_{13}])^2 - (2\mathrm{R}e[r_{24}])^2 \tag{7.15}$$

or

$$(2r_2 - 1)(1 - 2r_4) > -(2\mathrm{I}m[r_{24}])^2 - (2\mathrm{R}e[r_{13}])^2 \tag{7.16}$$

is fulfilled, then ρ' becomes more entangled than ρ. In a further iteration, the output state ρ'' remains in the same form as the input state ρ' and according to Eq. (7.14) its elements read

$$\begin{aligned} r''_1 &= \frac{r'^2_1 + r'^2_3}{D'}, \quad r''_2 = \frac{r'^2_2 + r'^2_4}{D'}, \quad r''_3 = 2\frac{r'_2 r'_4}{D'} \\ r''_4 &= 2\frac{r'_1 r'_3}{D'}, \quad r''_{12} = \frac{r'^2_{12} + r'^2_{34}}{D'} = \left(r''_{21}\right)^*, \\ r''_{34} &= 2\frac{r'_{21} r'_{43}}{D'} = \left(r''_{43}\right)^*, \end{aligned}$$

where

$$D' = (r'_1 + r'_3)^2 + (r'_2 + r'_4)^2.$$

In the asymptotic limit, the protocol converts all states fulfilling Eq. (7.15) into $|1\rangle\langle 1|$ and Eq. (7.16) into $|2\rangle\langle 2|$.

7.2.4 A CNOT-based protocol

The proof presented in Ref. [TB16] is very general and one can apply it to obtain a better CNOT-based protocol, where the transformations into either Werner or Bell diagonal state are omitted. Our proposed protocol reads

- Apply the unitary operation $u_1^{\dagger A_1} \otimes u_1^{\dagger A_2} \otimes u_1^{B_1} \otimes u_1^{B_2}$, see Eq. (7.6).
- Perform the bilateral operation $\hat{U}_{CNOT}^{A_1 \to A_2} \otimes \hat{U}_{CNOT}^{B_1 \to B_2}$.
- Measure the target pair (A_2, B_2) in eigenbasis of σ_z with corresponding results (m, n), where $m, n \in \{0, 1\}$. Keep the pair (A_1, B_1) if the measurement result is $m = n = 1$.

The protocol is described by the map

$$\begin{aligned} r'_1 &= 2\frac{r_2 r_3 - |r_{23}|^2}{E}, \quad r'_2 = \frac{r_2^2 + r_3^2 + r_{23}^2 + r_{32}^2}{E} \\ r'_3 &= 2\frac{r_1 r_4 - |r_{14}|^2}{E}, \quad r'_4 = \frac{r_1^2 + r_4^2 + r_{14}^2 + r_{41}^2}{E}, \\ r'_{13} &= 2\frac{r_{24} r_{31} - r_{21} r_{34}}{E}, \quad r'_{24} = \frac{r_{21}^2 + r_{31}^2 + r_{24}^2 + r_{34}^2}{E}, \\ r'_{12} &= r'_{14} = r'_{23} = r'_{34} = 0, \end{aligned} \tag{7.17}$$

where the success probability $P_s = E/2$ and

$$E = (r_1 + r_4)^2 + (r_2 + r_3)^2 + (r_{14} - r_{41})^2 + (r_{23} - r_{32})^2.$$

According to Eq. (7.3), we also have $r'_{31} = \left(r'_{13}\right)^*$ and $r'_{42} = \left(r'_{24}\right)^*$. Therefore, r'_1, r'_2, r'_3, r'_4, r'_{13}, r'_{31}, r'_{24}, and r'_{42} are the nonvanishing elements of ρ'. To increase the degree of entanglement of ρ', the state ρ has to fulfill either

$$(2r_1 - 1)(1 - 2r_4) > -(2\mathrm{I}m[r_{23}])^2 - (2\mathrm{R}e[r_{14}])^2 \tag{7.18}$$

or

$$(2r_2 - 1)(1 - 2r_3) > -(2\mathrm{I}m[r_{14}])^2 - (2\mathrm{R}e[r_{23}])^2. \tag{7.19}$$

It is worth noting that one can keep the pair (A_1, B_1) if the measurement result on (A_2, B_2) is $m = n = 0$. However, in this case, the entanglement purification works only if either

$$(2r_1 - 1)(1 - 2r_4) > (2\mathrm{I}m[r_{23}])^2 + (2\mathrm{R}e[r_{14}])^2 \tag{7.20}$$

or

$$(2r_2 - 1)(1 - 2r_3) > (2\mathrm{I}m[r_{14}])^2 + (2\mathrm{R}e[r_{23}])^2. \tag{7.21}$$

is fulfilled. These conditions are more restrictive than their counterparts in Eqs. (7.18) and (7.19). They work for Bell diagonal states, but many states do not obey them, and therefore for a general purification strategy the pair (A_1, B_1) has to be discarded, whenever the pair (A_2, B_2) is measured in the state $|00\rangle$. In a further iteration, the output state ρ'' remains in the same form as the input state ρ' and according to Eq. (7.17) its elements read

$$\begin{aligned}
r''_1 &= 2\frac{r'_2 r'_3}{E'}, \quad r''_2 = \frac{r'^2_2 + r'^2_3}{E'}, \quad r''_3 = 2\frac{r'_1 r'_4}{E'} \\
r''_4 &= \frac{r'^2_1 + r'^2_4}{E'}, \quad r''_{13} = 2\frac{r'_{24} r'_{31}}{E'} = \left(r''_{31}\right)^*, \\
r''_{24} &= \frac{r'^2_{31} + r'^2_{24}}{E'} = \left(r''_{42}\right)^*,
\end{aligned}$$

where

$$E' = (r'_1 + r'_4)^2 + (r'_2 + r'_3)^2.$$

If the states fulfill the condition given in Eq. (7.18) then the protocol can distill $|4\rangle\langle 4|$, otherwise in the case of Eq. (7.19), the state $|2\rangle\langle 2|$ is obtained. It is worth noting that these Bell states can also be reached not only in the asymptotic limit. Based on the second example in Ref. [TB16], one can pick $r_2 = c$, $r_1 = r_4 = (1 - c)/2$, and $r_{14} = (r_{41})^* = i(1 - c)/2$ with $c \in (0, 1]$ to observe that one iteration of the protocol yields $|2\rangle\langle 2|$ with success probability $c^2/2$. This state cannot be purified by the Bennett protocol. If $c \in (0.5, 1]$, then the Deutsch protocol approaches the Bell state $|2\rangle$ only asymptotically.

7.2.5 Variational purification protocols

In order to implement a numerical search for optimal protocols, a cost function has to be defined. A convenient choice for the measure of the performance of a two-qubit-based entanglement purification protocol is the concurrence [PCT&al22], for which we use the definition introduced in Ref. [Woo98]:

$$\mathcal{C}(\rho) = \max\{0, \sqrt{\lambda_1} - \sqrt{\lambda_2} - \sqrt{\lambda_3} - \sqrt{\lambda_4}\}. \tag{7.22}$$

Here $\lambda_1, \lambda_2, \lambda_3, \lambda_4$ are the eigenvalues, listed in decreasing order, of the matrix

$$\tilde{\rho} = \rho(\sigma_y \otimes \sigma_y)\rho^*(\sigma_y \otimes \sigma_y),$$

where the asterisk represents the complex conjugation of ρ in the standard basis. The advantage of the concurrence lies in the fact that it treats all maximally entangled states equivalently. As we have discussed in the previous sections, the maps there can have two stable fixed points corresponding to two Bell states and they can approach both of them depending on the properties of the input density matrix. Therefore, picking a fidelity with respect to a Bell state would not work, because the optimization would suppress the convergence towards another Bell state or any maximally entangled state, which can be reached otherwise. A different argument against the use of fidelity with an example is given in Ref. [PCT&al22].

Let us consider a general unitary matrix $V \in SU(4) \otimes SU(4)$ acting on A_1, A_2 and B_1, B_2. We employ the Euler-angle parametrization of $SU(4)$ [TBS02],

$$\begin{aligned} U(\boldsymbol{\alpha}) = {}& e^{i\sigma_3\alpha_1} e^{i\sigma_2\alpha_2} e^{i\sigma_3\alpha_3} e^{i\sigma_5\alpha_4} e^{i\sigma_3\alpha_5} e^{i\sigma_{10}\alpha_6} e^{i\sigma_3\alpha_7} e^{i\sigma_2\alpha_8} \\ & e^{i\sigma_3\alpha_9} e^{i\sigma_5\alpha_{10}} e^{i\sigma_3\alpha_{11}} e^{i\sigma_2\alpha_{12}} e^{i\sigma_3\alpha_{13}} e^{i\sigma_8\alpha_{14}} e^{i\sigma_{15}\alpha_{15}}, \end{aligned}$$

with $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \dots, \alpha_{15})^T \in \mathbb{R}^{15}$ (T denotes the transposition), and

$$\begin{aligned} & 0 \leqslant \alpha_1, \alpha_3, \alpha_5, \alpha_7, \alpha_9, \alpha_{11}, \alpha_{13} \leqslant \pi, \\ & 0 \leqslant \alpha_2, \alpha_4, \alpha_6, \alpha_8, \alpha_{10}, \alpha_{12} \leqslant \frac{\pi}{2} \\ & 0 \leqslant \alpha_{14} \leqslant \frac{\pi}{\sqrt{3}}, \quad 0 \leqslant \alpha_{15} \leqslant \frac{\pi}{\sqrt{6}}, \end{aligned} \tag{7.23}$$

where we use the formulation of Ref. [TBS02]. The terms σ_i, $i = 1, \dots, 15$ are a Gell-Mann type basis of the Lie group $SU(4)$ (see the Appendix and the discussion in Ref. [PCT&al22], where the same approach to angle parametrization was used). Then we have

$$V(\boldsymbol{\alpha}_{AB}) = U^{A_1,A_2}(\boldsymbol{\alpha}_A) \otimes U^{B_1,B_2}(\boldsymbol{\alpha}_B) \tag{7.24}$$

where $\boldsymbol{\alpha}_{AB} = (\boldsymbol{\alpha}_A, \boldsymbol{\alpha}_B)^T$ and thus $\boldsymbol{\alpha}_{AB} \in \mathbb{R}^{30}$. The Euler-angle parametrization guarantees that the group is not naively overcounted [TBS02]. It is worth noting that due to the nature of the problem, one cannot consider operations that improve entanglement across the channel, i.e., general $SU(16)$ operations on A and B, because badly entangled states are the result of the noisy channel.

The purification protocol for our numerical investigations reads as follows.

(i) Apply entangling operations in both nodes which results in the state

$$\varrho' = \frac{\Pi\varrho\Pi^\dagger}{\mathrm{Tr}\left\{\Pi^\dagger\Pi\varrho\right\}}, \tag{7.25}$$

where Π is a quantum operation. In the numerical simulations, apart from one particular case, Π equals $V(\boldsymbol{\alpha}_{AB})$.

(ii) Measure the pair (A_2, B_2) in eigenbasis of σ_z with corresponding results (m, n), where $m, n \in \{0, 1\}$. The four projectors of the (m, n) measurement results are:

$$(0,0) \to P_1 = I_{A_1,B_1} \otimes |00\rangle\langle 00|_{A_2,B_2}, \tag{7.26}$$
$$(0,1) \to P_2 = I_{A_1,B_1} \otimes |01\rangle\langle 01|_{A_2,B_2}, \tag{7.27}$$
$$(1,0) \to P_3 = I_{A_1,B_1} \otimes |10\rangle\langle 10|_{A_2,B_2}, \tag{7.28}$$
$$(1,1) \to P_4 = I_{A_1,B_1} \otimes |11\rangle\langle 11|_{A_2,B_2}. \tag{7.29}$$

We define a selection function π (measurement policy) that chooses which measurement result is kept:

$$(\rho_1^{A_1,B_1}, \rho_2^{A_1,B_1}, \rho_3^{A_1,B_1}, \rho_4^{A_1,B_1}) \mapsto \{1, 2, 3, 4\}, \tag{7.30}$$

where

$$\rho_k^{A_1,B_1} = \frac{\mathrm{Tr}_{A_2,B_2}\{P_k\varrho' P_k\}}{\mathrm{Tr}\{P_k\varrho'\}}. \tag{7.31}$$

This function can be arbitrarily modified to suit different implementations. A possibility is to choose a greedy policy:

$$\pi = \operatorname*{argmax}_{k=1,2,3,4} \left[\mathcal{C}(\rho_k^{A_1,B_1})\right]. \tag{7.32}$$

In this case, one measures the subsystem (A_2, B_2) along all four measurement directions and keeps the measurement that corresponds to the highest concurrence. Alternatively, we also consider the case with only one of the measurement results, e.g., $k = 1$ $(m = n = 0)$. We will explore both these policies in our optimization.

If we employ $\Pi = V(\boldsymbol{\alpha}_{AB})$ as the entangling operation, the output density matrix ρ_{out} of the variational purification protocol will depend on the value of $\boldsymbol{\alpha}_{AB}$:

$$\rho \to \rho_{\text{out}}(\boldsymbol{\alpha}_{AB}). \tag{7.33}$$

7.3 Numerical methods

7.3.1 Markov chain Monte Carlo sampler

In this approach, the main mathematical object is the vector space $M_4(\mathbb{C})$ of 4×4 matrices with complex entries. This vector space with the Hilbert-Schmidt inner product $\langle A, B\rangle_{HS} = \mathrm{Tr}\{A^\dagger B\}$ where $A, B \in M_4(\mathbb{C})$ is a 16-dimensional Hilbert space. In this vector space, self-adjoint matrices form a subspace. This subspace can be identified with the Euclidean space $\mathbb{R}^{16}$, if the orthonormal basis is constructed from tensor products of Pauli matrices and the 2×2 unit matrix I_2. There are also other possibilities, such as the Gell-Mann-type basis of $SU(4)$ or the Weyl operator basis [BK08], and they either result in the same Euclidean structure or are unsuitable for our numerical approach based on $\mathbb{R}^{16}$. If ρ is a density matrix, then

$$\rho = \sum_{i=1}^{16} a_i B_i \quad \text{with} \quad \mathrm{Tr}\{\rho\} = 1, \quad \rho \geqslant 0, \tag{7.34}$$

where the B_i are the basis vectors and $a_i \in \mathbb{R}$ for all i. The positive semidefinite condition $\rho \geqslant 0$ implies that the a_i have to fulfill three conditions based on Newton identities and Descartes' rule of signs [Kim03; SBM&al21]. Finally, one arrives at the result that all 4×4 density matrices are presented by a 15 dimensional convex body K around the origin of $\mathbb{R}^{15}$, because due to $\mathrm{Tr}\{\rho\} = 1$ one real parameter out of sixteen is fixed. Hence, if we consider $B_{16} = I_4/2$ then $a_{16} = 1/2$ and the origin of $\mathbb{R}^{15}$ is the maximally mixed state. The vector $\boldsymbol{a} = (a_1, a_2, \dots a_{15})^T \in \mathbb{R}^{15}$ with the positive semidefinite condition yields a complete description of K. The hit-and-run algorithm introduced by Smith [Smi84] realizes a random walk inside K and it was shown that the underlying Markov chain converges to the uniform stationary distribution in polynomial time [Lov99]. The convergence to the uniform distribution is independent of the starting point inside K [LV06]. This algorithm provides a fast method of sampling large numbers $(10^6 - 10^7)$ of density matrices. There exist other numerical approaches [SBM&al21; SSN&al15b; ŻS01; SSN&al15a; MS14] that can also sample uniformly distributed density matrices, albeit with longer running times. All the statistical evaluations of the purification protocols are based on samples generated by the hit-and-run algorithm, whose details are shown in Appendix 7.7.1.

7.3.2 Optimization with a quasi-Newton method

Our task is to improve the concurrence of the output state, which is a nonlinear function of $\boldsymbol{\alpha}_{AB}$ (see Sec. 7.2.5). In addition, based on Eq. (7.23) $\boldsymbol{\alpha}_{AB}$ has lower and upper bounds. Then, by using Eq. (7.33) we define the cost function $f : \mathbb{R}^{30} \to \mathbb{R}$ as

$$f(\boldsymbol{\alpha}_{AB}) = 1 - \mathcal{C}\left[\rho_{\text{out}}(\boldsymbol{\alpha}_{AB})\right]. \tag{7.35}$$

The cost function is based on the concurrence of the two-qubit state emerging after one iteration of the protocol. Let us consider that the sample of two-qubit density matrices

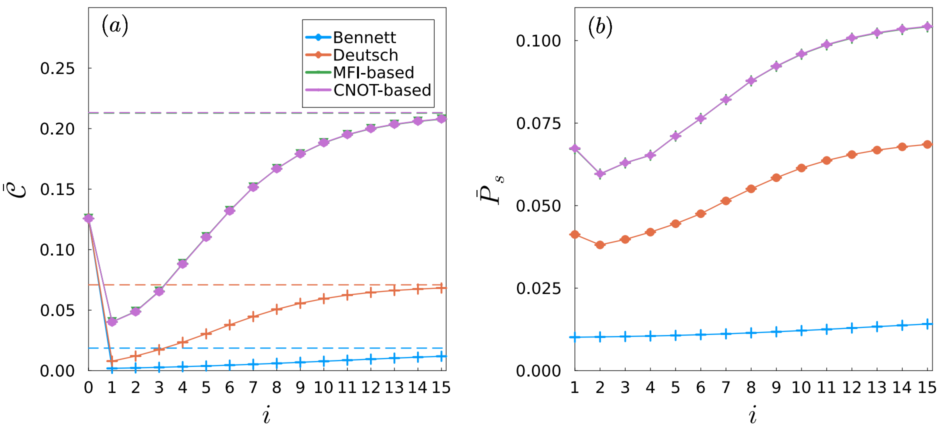


Figure 7.1: (a) Average concurrence and (b) success probability as a function of the number of iterations for which a sample of $N = 10^7$ density matrices was used (ten runs of the hit-and-run algorithm, each one outputting 10^6 density matrices). Numerical evaluations are done for the four different purification protocols presented in Secs. 7.2.1–7.2.4, i.e., Bennett, Deutsch, MFI-based, and CNOT-based protocols, respectively. Horizontal dashed lines show the theoretical limits of the protocols for the average concurrence, which are given in Eqs. (7.49)–(7.52) for the Bennett, Deutsch, MFI-based, and CNOT-based protocols, respectively. The points are connected by lines to guide the eye. The MFI-based protocol and our proposed CNOT protocol turn out to produce the same average values, to the point that the green points are barely visible beyond the purple ones. The standard errors of the means are not visible in the plots. Values for the unbiased sample variance are available in Appendix 7.7.2.

consists of N elements $\rho_1, \dots, \rho_N$, which are mapped onto $\rho_{\text{out},1}, \dots, \rho_{\text{out},N}$. The average output concurrence is estimated by

$$\bar{\mathcal{C}}(\boldsymbol{\alpha}_{AB}) = \frac{1}{N} \sum_{j=1}^{N} \mathcal{C}\left[\rho_{\text{out},j}(\boldsymbol{\alpha}_{AB})\right]. \tag{7.36}$$

Then, the average cost function reads

$$\bar{f}(\boldsymbol{\alpha}_{AB}) = 1 - \bar{\mathcal{C}}(\boldsymbol{\alpha}_{AB}), \tag{7.37}$$

which ensures that the optimal unitary transformations characterized by $\boldsymbol{\alpha}_{AB}$ increase the average concurrence of the whole sample. This method is the extension of the one developed in our previous investigation [PCT&al22], where the cost function depends only on 15-dimensional vectors, and the optimal search is constrained to specific families of two-qubit states. We implement again one of the most effective quasi-Newton methods, the L-BFGS-B optimization algorithm [BNS94; BLN&al95]. The gradient $\nabla\bar{f}(\boldsymbol{\alpha}_{AB})$ is obtained via automatic differentiation [BFH&al18]. The algorithm does not require second derivatives, because the Hessian matrix is approximated. This approach yields a local minimum $\boldsymbol{\alpha}^*_{AB}$ of $\bar{f}$, i.e., $\bar{f}(\boldsymbol{\alpha}^*_{AB}) \leqslant \bar{f}(\boldsymbol{\alpha}_{AB})$ for all $\boldsymbol{\alpha}_{AB}$ sufficiently close to $\boldsymbol{\alpha}^*_{AB}$. Now, we briefly describe the methodology used to optimize the variational purification protocols of Sec. 7.2.5. The optimization is subject to general two-qubit density matrices parametrized by a 15-dimensional real vector $\boldsymbol{a}$. As the number of samples needed to cover the space of two-qubit quantum states with sufficient precision is particularly large ($10^6 - 10^7$ samples; see [SB22]), the algorithm struggles with a particularly slow optimization. Therefore, as a first attempt, we use smaller subsets of density matrices to find optimal unitary matrices and then test their performance on the whole sample.

Algorithm 6 Optimization with the hit-and-run (HR) given in Algorithm 7

```
    Input $\boldsymbol{\rho}(\boldsymbol{a}), \boldsymbol{a} \in \mathbb{R}^{15}$, $P_1$, $P_2$, $P_3$, $P_4$ as in Eqs. (7.26), (7.27), (7.28), and (7.29), $L_{\max}$ iterations, optimizer (OPT)
    Output $\boldsymbol{\rho}(\boldsymbol{a})$
 1: $\boldsymbol{a}_0 = \mathbf{0}$
 2: for $j = 1$ to $N$ do
 3:     $\boldsymbol{a}_j = \mathrm{HR}(\boldsymbol{a}_{j-1})$
 4:     $\boldsymbol{\rho}_j = \boldsymbol{\rho}(\boldsymbol{a}_j)$
 5: end for
 6: for $i = 1$ to $L_{\max}$ do                                   ▷ with random restart
 7:     $V(\boldsymbol{\alpha}_{AB}) = U^{A_1,A_2}(\boldsymbol{\alpha}_A) \otimes U^{B_1,B_2}(\boldsymbol{\alpha}_B)$
 8:     for $j = 1$ to $N$ do
 9:         $\boldsymbol{\varrho}^j = \boldsymbol{\rho}_j^{A_1,B_1} \otimes \boldsymbol{\rho}_j^{A_2,B_2}$
10:         $\boldsymbol{\sigma}^j(\boldsymbol{\alpha}_{AB}) = V(\boldsymbol{\alpha}_{AB})\boldsymbol{\varrho}^j V^\dagger(\boldsymbol{\alpha}_{AB})$
11:         for $k = 1$ to $4$ do
12:             $\boldsymbol{\rho}^{jk}(\boldsymbol{\alpha}_{AB}) = \frac{\mathrm{Tr}_{A_2,B_2}\{P_k\boldsymbol{\sigma}^j(\boldsymbol{\alpha}_{AB})P_k\}}{\mathrm{Tr}\{P_k\boldsymbol{\sigma}^j(\boldsymbol{\alpha}_{AB})\}}$
13:         end for
14:         $k_{\max} = \pi(\boldsymbol{\rho}^{j1}, \boldsymbol{\rho}^{j2}, \boldsymbol{\rho}^{j3}, \boldsymbol{\rho}^{j4})$
15:         $\boldsymbol{\rho}^j(\boldsymbol{\alpha}_{AB}) = \boldsymbol{\rho}^{jk_{\max}}(\boldsymbol{\alpha}_{AB})$
16:     end for
17:     $\bar{f}(\boldsymbol{\alpha}_{AB}) = 1 - \frac{1}{N}\sum_{j=1}^{N}\mathcal{C}\left[\boldsymbol{\rho}^j(\boldsymbol{\alpha}_{AB})\right]$
18:     $\boldsymbol{\alpha}^*_{AB} = \mathrm{OPT}\left[\bar{f}(\boldsymbol{\alpha}_{AB}), \nabla_{\boldsymbol{\alpha}_{AB}}\bar{f}(\boldsymbol{\alpha}_{AB})\right]$
19:     for $j = 1$ to $N$ do
20:         $\boldsymbol{\rho}_j = \boldsymbol{\rho}^j(\boldsymbol{\alpha}^*_{AB})$
21:     end for
22: end for
```

The general optimization routine is presented in Algorithm 6. First, general density matrices are sampled, which is followed by the optimization of the average cost function. The output density matrices are used again in the next purification step. This is also optimized, until the maximal number of iterations $L_{\max}$ is reached. In every step of the iteration, we find different optimal unitary matrices, which yields an adaptive purification protocol [PCT&al22].

7.4 Results

7.4.1 Statistics of purification protocols

In this section, we compare the recurrence protocols of Secs. 7.2.1, 7.2.2, 7.2.3, and 7.2.4 based on their average performance on a sample of two-qubit states drawn from an almost uniform distribution [SBM&al21]. More formally, let $\rho_1, ..., \rho_N$ be N density matrices generated by the hit-and-run algorithm. As the composition of completely positive maps is again completely positive [Pau03], every entanglement purification protocol presented in this work acts as a completely positive trace-preserving non-linear map. We denote this by Φ, which maps the set of two-qubit quantum states

$$D(\mathbb{C}^4) = \{\rho \in M_4(\mathbb{C}) : \rho \geq 0,\ \mathrm{Tr}\{\rho\} = 1\}, \tag{7.38}$$

i.e., positive semidefinite matrices with unit trace, onto itself

$$\Phi : D(\mathbb{C}^4) \mapsto D(\mathbb{C}^4), \quad \Phi(\rho) = \rho'. \tag{7.39}$$

A purification map Φ is iteratively applied to a density matrix to purify it towards a maximally entangled state, that is, to extract a state with higher concurrence. If successful, the protocol approaches usually the concurrence value $\mathcal{C} = 1$ in the limit of infinite iterations, but there are also known cases of one-step purifiable states [BDS&al96; TB16]. If unsuccessful, then a state with non-zero concurrence is mapped to a state with zero concurrence, thereby destroying entanglement, which happens at the first iteration of each protocol. The average concurrence after i iterations is estimated by

$$\bar{\mathcal{C}}^{(i)} = \frac{1}{N}\sum_{j=1}^{N} \mathcal{C}\left[\Phi^i(\rho_j)\right]. \tag{7.40}$$

The sample standard deviation reads

$$s_{\mathcal{C}}^{(i)} = \sqrt{\frac{1}{N-1}\sum_{j=1}^{N}\left\{\mathcal{C}\left[\Phi^i(\rho_j)\right] - \bar{\mathcal{C}}^{(i)}\right\}^2} \tag{7.41}$$

and the standard error, i.e., the standard deviation of $\bar{\mathcal{C}}^{(i)}$, is given by:

$$\sigma_{\bar{\mathcal{C}}}^{(i)} = \frac{s_{\mathcal{C}}^{(i)}}{\sqrt{N}}. \tag{7.42}$$

Similarly, the average success probability can be estimated. In this case, we should first point out that for a general two-qubit density matrix ρ the success probability can only be defined for those states whose concurrence is non-zero, that is in those cases where the entanglement purification protocols are not failing. The probability of a measurement at the i-th iteration can be described as

$$p_k = \mathrm{Tr}\left\{P_k \Phi^i(\rho_j)\right\}, \quad k \in \{1,2,3,4\}, \tag{7.43}$$

$$P_s^{(i)}(k,\rho_j) = \begin{cases} p_k & \text{if } C\left[\Phi^i(\rho_j)\right] \geq 0, \\ 0 & \text{otherwise,} \end{cases} \tag{7.44}$$

where P_k is the projector on (A_2, B_2) defined in Eqs. (7.26), (7.27), (7.28), and (7.29). In the case of the protocols presented in Secs. 7.2.1, 7.2.2, 7.2.3, different measurement results yield the same probability, and analytical formulas are given for the success probability $P_s^{(i)}(\rho_j)$. In the case of the MFI-based protocol, the success probability is given by the probability of successfully performing the quantum operation given in Eq. (7.13) and the probability of measuring the state in one of the four possible outcomes is 1/4. The success probability of the approach in Sec. 7.2.4 is also shown. However, in the optimized variational case, $P_s^{(i)}(\rho_j)$ is a function of $P_s^{(i)}(k,\rho_j)$, where k is selected by the policy π introduced in Eq. (7.30). It is worth noting that the condition in Eq. (7.44) is decisive only in the first step for the protocols in Secs. 7.2.1 and 7.2.2, because after that all the remaining entangled two-qubit states are improved towards a Bell state. The average success probability is estimated after each iteration step by

$$\bar{P}_s^{(i)} = \frac{1}{N}\sum_{j=1}^{N} P_s^{(i)}(\rho_j), \tag{7.45}$$

which corresponds to the probability of successfully implementing the i-th step of the entanglement purification protocol on an unknown two-qubit state with non-zero concurrence. In this case, the standard deviation of the sample is then given by

$$s_{P_s}^{(i)} = \sqrt{\frac{1}{N-1}\sum_{j=1}^{N}\left\{P_s^{(i)}(\rho_j) - \bar{P}_s^{(i)}\right\}^2} \tag{7.46}$$

with the standard error

$$\sigma_{\bar{P}_s}^{(i)} = \frac{s_{P_s}^{(i)}}{\sqrt{N}}. \tag{7.47}$$

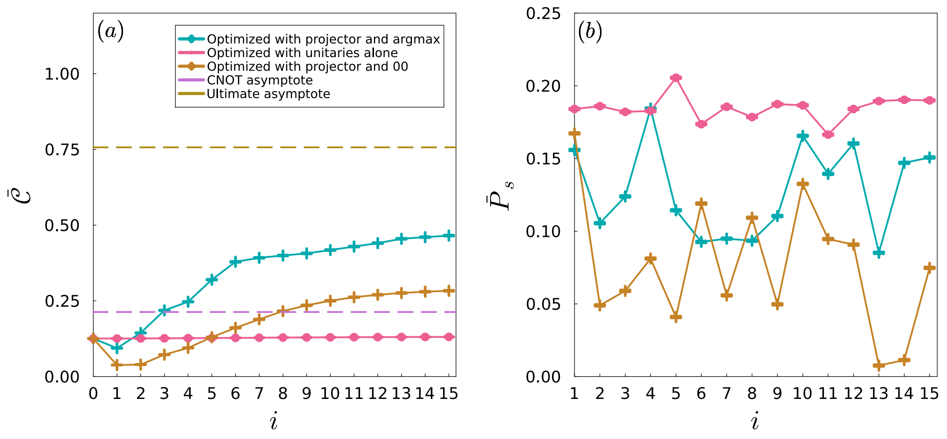


Figure 7.2: (a) Average concurrence and (b) success probability of the optimized purification protocols as a function of the number of iterations. The values shown in the plot represent the average over $N = 10^7$ density matrices (ten runs of the hit-and-run algorithm, each one outputting 10^6 density matrices). The asymptote of the MFI-based protocol in Eq. (7.51) and the ultimate limit of Eq. (7.53) are also shown. The points are connected by lines to guide the eye. The strategy of first destroying entanglement and then improving it (green points) delivers the best performance compared also to the protocols of Fig. 7.1. If we only keep the measurement result $k = 1$ $(m = 0, n = 0)$, the average concurrence turns out to be lower compared to the case where the argmax policy of Eq. (7.32) is used. The standard error of the mean is not visible in the plot. Values for the unbiased sample variance are available in Appendix 7.7.2.

The initial average concurrence is calculated over $N = 10^7$ density matrices and yields

$$\bar{\mathcal{C}}^{(0)} = 0.1257(2), \tag{7.48}$$

which represents the amount of entanglement present in the sample. Here we use parentheses to denote the standard error. We consider this number as a threshold and expect that entanglement purification protocols can improve it. The results of the four protocols are shown in Fig. 7.1. It is immediate to see that all protocols destroy entanglement in their very first iteration. The twirling operations used in both the Bennett and Deutsch protocols to obtain Werner or Bell diagonal states, respectively, destroy a significant portion of the entangled states. The remaining entangled states reach $\mathcal{C} = 1$ in the limit of infinite iterations. We find these limits by imposing the conditions given by Eqs. (7.8) and (7.12) on the whole sample. If a state does not satisfy it, then we assign $\mathcal{C} = 0$ to it, otherwise, we set $\mathcal{C} = 1$. In other words, every state satisfying the conditions can be purified into a Bell state in the limit of infinite iterations. This limit for the Bennett protocol yields

$$\bar{\mathcal{C}}^{(\infty)}_{\text{Bennett}} = 0.01865(4), \tag{7.49}$$

while the Deutsch approach results in

$$\bar{\mathcal{C}}^{(\infty)}_{\text{Deutsch}} = 0.0709(1). \tag{7.50}$$

We would like to stress that this value is lower than the initial average concurrence in Eq. (7.48) for both protocols. Furthermore, they require several iterations to approach these limits. The other two protocols yield

$$\bar{\mathcal{C}}^{(\infty)}_{\text{MFI}} = 0.2128(1), \tag{7.51}$$

and

$$\bar{\mathcal{C}}^{(\infty)}_{\text{CNOT}} = 0.2133(1), \tag{7.52}$$

which are based on the conditions in Eqs. (7.15), (7.16), (7.18), and (7.19). The performance of these protocols is almost the same and despite destroying entanglement in the first iteration, they can improve the average concurrence of the sample beyond its starting value. They achieve this already after four iterations (see Fig. 7.1). It is worth noting again that the asymptote of the Bennett protocol contains only the Bell state $|1\rangle$, being a stable fixed point, while the other three protocols have two relevant stable fixed points in the limit of infinite iterations. These findings together with the average success probabilities characterize the performances and they show that both the MFI-based and our CNOT protocols are superior to the pioneering ones. However, one can define an ultimate limit of every known or future proposal for entanglement purification, namely, when all entangled two-qubit states are converted to a maximally entangled state, which for our sample results in the numerical estimate

$$\bar{\mathcal{C}}^{(\infty)}_{\text{ultimate}} = 0.7569(1). \tag{7.53}$$

Given this number, one can conclude that all four evaluated protocols are not very effective. The performance is confirmed by the analysis of the fidelities with respect to the stable fixed-point states of the protocols, i.e., r_1 for the Bennett, r_4 and r_2 for the Deutsch, r_1 and r_2 for the MFI and r_4 and r_2 for the CNOT protocols, as a function of the number of protocol iterations, which are shown in Appendix 7.7.3. This raises the question of how one can obtain better performance, which will be discussed in the subsequent section.

7.4.2 Optimization of variational recurrence protocols

In this section, we investigate numerically the variational purification protocol described in Sec. 7.2. The optimization is based on the method given in Sec. 7.3.2, where the gradient of the cost function associated with the protocol is computed through automatic differentiation. First, we generate $N = 10^6$ density matrices using the hit-and-run algorithm to obtain an almost uniformly distributed sample. Afterwards, to speed up the optimization, we randomly pick $N_s = 1000$ density matrices from this sample and calculate the gradient. This approach guarantees that optimization is less time consuming and the convex set of all two-qubit states is well represented. Nevertheless, the sampling of only 10^3 density matrices is not enough to ensure uniform distribution, in particular for the statistics of the CNOT and MFI protocols. The optimization results in a numerical value for $\boldsymbol{\alpha}^*_{AB}$. We then apply the $\boldsymbol{\alpha}^*_{AB}$-dependent protocol on the whole sample. We also try different strategies for the measurement policy π of Eq. (7.30). In particular, we test the policy of Eq. (7.32) together with the case in which we only track the measurement result $k = 1$ ($m = n = 0$) in each iteration.

Our first try is the case when Π in Eq. (7.25) is equal to $V(\boldsymbol{\alpha}_{AB})$ with policy π of Eq. (7.32). In Fig. 7.2 we present the result, which demonstrates that this strategy does not destroy initial entanglement, as one expects. However, the actual increase in average concurrence is quite limited, which implies that an enormous number of iterations are needed to purify the density matrices towards a maximally entangled state. As we seek to obtain a more efficient protocol, we need to consider additional operations performed on the sample of density matrices.

An interesting feature of Sec. 7.4.1 is that every protocol first destroys entanglement before it starts to improve the remaining two-qubit quantum states. Next, we consider the hypothesis that entanglement must be destroyed in order to find an efficient protocol. Therefore, for the first iteration, we use the projector

$$\Pi = M_2^{A_1,A_2} \otimes M_2^{B_1,B_2} \quad \text{with} \quad M_2 = I_4 - |2\rangle\langle 2|, \tag{7.54}$$

as the operation of the purification protocol, and for the rest of the iterations we make use of the optimization with $\Pi = V(\boldsymbol{\alpha}_{AB})$, which leads to higher average concurrence. Here, we have tested both of our measurement policies π. In both cases, these approaches outperform the limits of the MFI-based and the CNOT protocols, which are shown in Fig. 7.2. We remind the reader that every optimized step of the variational protocol yields a different entangling gate. The resulting performance for many iterations seemingly approaches a limit, which is almost halfway between the ultimate asymptote and the asymptote defined by the MFI-based or the CNOT protocol. The average success probability of this optimized protocol oscillates as a function of the number of iterations. We do not have a proper explanation for this effect, as it depends on the non-linear optimizer. However, we can see that the values of the success probability are almost always higher than those produced by the protocols considered in Fig. 7.1.

Finally, we need to address the realization of these optimized and abstract protocols. Concerning the two-qubit gates in $V(\boldsymbol{\alpha}_{AB})$, one can always employ quantum compilation strategies [MMN&al16; Deb&al16; OGB21; PSJ&al24], where the optimal unitary matrix $V(\boldsymbol{\alpha}^*_{AB})$ is translated into native gates on the chosen experimental platform. However, we are not aware of any possible implementation of M_2 in Eq. (7.54), but it seems necessary to first destroy entanglement before any variational purification protocol is applied.

7.5 Summary and conclusions

To summarize, we have presented a numerical method that is capable of characterizing the performance of entanglement purification protocols. We have presented a CNOT-based protocol, which was evaluated together with two pioneering protocols and a recent proposal based on matter-field interactions. Our results show that all the protocols destroy entanglement in the very first iteration. This was known for the pioneering protocols, and here in addition we have demonstrated quantitatively that only a small set of entangled states is kept. Even though these states are turned into a Bell state in the limit of infinite iterations, the average concurrence of the whole sample stays below its starting value. The MFI-based and the CNOT-based protocols perform better and they can turn slightly more than 21% of the two-qubit states into a Bell state.
We have defined the ultimate limit of all possible purification protocols, which is nothing other than the percentage of all entangled states within the set of all quantum states, i.e., approximately 75%. In other words, an ultimate protocol can purify all entangled states into maximally entangled ones. In this context, we have searched for optimal asymmetric entangling gates in the nodes A and B. We have found that this approach is improving the average concurrence very slowly. Therefore, motivated by the other approaches, we have included in the first iteration an entangling projection, which destroys some entanglement. This strategy turns out to be a boost for the optimized variational approach, which can outperform all protocols discussed in this paper. However, even the variational approach seems unable to reach the ultimate bound for entanglement purification. At the current stage, we cannot determine whether this result could be improved by implementing different quantum operations or optimization methods. There might also be an upper bound for this family of protocols that lies below the ultimate asymptote.
Our numerical analysis focuses on the improvement of the average concurrence. However, we would like to point out that all the protocols investigated in this work have a common property. If some information about the input state is known, e.g., the state has an overlap strictly larger than 0.5 with one of the Bell states, then we know beforehand whether or not these protocols convert entangled states into a fixed maximally entangled state, which is usually a Bell state. A further difficulty is that the iterations of the protocol are nonlinear quantum state transformations, which lead to chaotic behavior [KVT&al11; GNX&al13; GKJ16; PKJ&al22]. Therefore, in the trace norm topology [Pau03], seemingly close density matrices might have different future trajectories. Our approach avoids this interesting but complicated behavior by using the average concurrence, a choice, which we have demonstrated to be also successful in finding different and effective entanglement purification protocols.
Finally, some comments on the average success probabilities are in order. It is known that improving the concurrence alone is not a good enough measure, because success probabilities play a crucial role in the identification of required resources, i.e., how many qubits are required to perform some iterations. To improve the success probabilities as well, this leads to a multiobjective optimization task. This is not included here, because this work aims to introduce a general evaluation scheme based on the

hit-and-run algorithm and the concurrence, which have given insight into the performance of entanglement purification protocols and shown that improvements are possible in computer-based protocol designs.

7.6 Acknowledgments

This work was supported by AIDAS-AI, Data Analytics and Scalable Simulation, which is a Joint Virtual Laboratory gathering the Forschungszentrum Jülich and the French Alternative Energies and Atomic Energy Commission, by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy – Cluster of Excellence Matter and Light for Quantum Computing (ML4Q) EXC 2004/1 – 390534, and by the Jülich Supercomputing Center. We are grateful to J. M. Torres and T. Calarco for stimulating discussions. Simulations were realized in PYTHON using the library JAX [BFH&al18] and in JULIA. The code and the data are available at `https://github.com/franz3105/soepp` (code) and `https://zenodo.org/uploads/10996551` (data sets).

7.7 Appendix

7.7.1 Hit-and-run algorithm

In this Appendix we briefly describe the steps of the hit-and-run algorithm in Algorithm 7. Given a $K \subseteq \mathbb{R}^{15}$, we generate for an $\boldsymbol{a} \in K$ a random uniform vector $\boldsymbol{x}$ on the sphere, which is around $\boldsymbol{a}$ and has unit radius. We generate a random uniform number λ on the interval $[-\sqrt{3}/2, \sqrt{3}/2]$, because K is inside the sphere of radius $\sqrt{3}/2$ around the origin [SBM&al21]. If $\boldsymbol{a'} = \boldsymbol{a} + \lambda \boldsymbol{x} \in K$, then we move there, otherwise we start all over from $\boldsymbol{a}$. We always start the sampling from $\boldsymbol{a} = \boldsymbol{0}$, i.e., the maximally mixed state.

Algorithm 7 Hit-and-run

```
1: $j = 1$ and $\boldsymbol{a}^{(1)} = \mathbf{0}$.
2: while $j < N$ do
3:     $\boldsymbol{x}^{(j)} \sim \mathcal{N}(\mathbf{0}, I_{15})$.
4:     $\boldsymbol{x}^{(j)} = \boldsymbol{x}^{(j)}/\|\boldsymbol{x}^{(j)}\|$
5:     Set $I = [-r, r]$ with $r = \sqrt{3}/2$.
6:     $m = 0$
7:     while $m = 0$ do
8:         $\lambda \sim \mathcal{U}_I$.
9:         if $\boldsymbol{a}^{(j)} + \lambda \boldsymbol{x}^{(j)} \in K$ then
10:            $\boldsymbol{a}^{(j+1)} = \boldsymbol{a}^{(j)} + \lambda \boldsymbol{x}^{(j)}$
11:            $j = j + 1$
12:            $m = 1$
13:        else
14:            if $\lambda > 0$ then
15:                $I = [-r, \lambda]$
16:            else
17:                $I = [\lambda, r]$
18:            end if
19:        end if
20:    end while
21: end while
```

7.7.2 Statistics of entanglement purification

In this Appendix, details concerning the sample mean and standard deviation are discussed. The primary task is to estimate numerically different averages of the concurrence or the success probability over all two-qubit density matrices. In the main text we have already identified this set with the convex body K in the Euclidean space $\mathbb{R}^{15}$. Thus, every density matrix ρ in Eq. (7.34) is uniquely described by a vector $\boldsymbol{a} \in K$. In this context, the average concurrence reads

$$\bar{\mathcal{C}} = \frac{1}{\mathrm{vol}(K)} \int_{\boldsymbol{a}\in K} \mathcal{C}\left[\rho(\boldsymbol{a})\right] d^{15}\boldsymbol{a} \tag{7.55}$$

with respect to the Lebesgue measure in $\mathbb{R}^{15}$. If $\rho_1, ..., \rho_N$ are generated by the hit-and-run algorithm, then the estimated value of $\bar{\mathcal{C}}$ reads $\sum_{j=1}^{N} \mathcal{C}(\rho_j)/N$. Similarly, the average

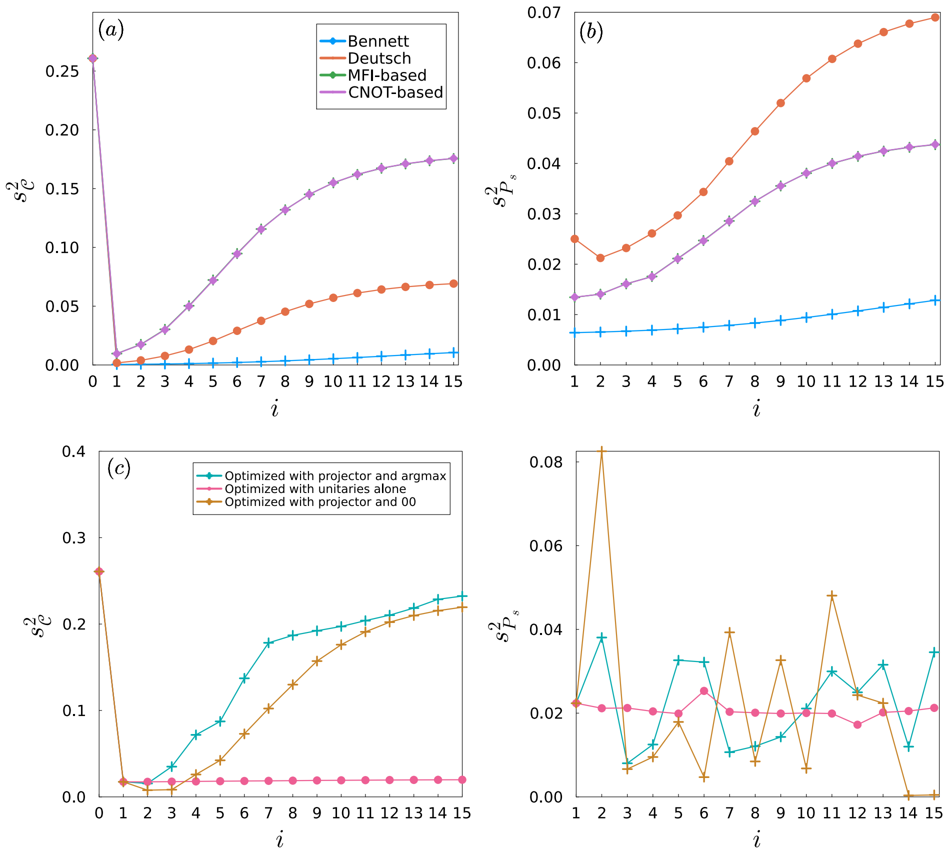


Figure 7.3: Unbiased sample variances of the concurrence and the success probability as a function of the number of iterations for (a) and (b) the non-variational and (c) and (d) the optimized variational protocols. We use the same color scheme as in Figs. 7.1 and 7.2. We observe that the sample variance grows with the number of iterations and reaches its maximum asymptotic values when all the density matrices are all mapped to either one or zero concurrence [see Eq. (7.58)]. The fraction of density matrices with non-zero concurrence in the limit of infinite iterations determines the performance of the protocol and also the maximum of the sample variance.

success probability

$$\bar{P}_s = \frac{1}{\mathrm{vol}(K)} \int_{\boldsymbol{a} \in K} P_s \left[\rho(\boldsymbol{a})\right] d^{15}\boldsymbol{a} \tag{7.56}$$

is estimated by $\sum_{j=1}^{N} P_s(\rho_j)/N$. The standard deviations of these means are given in Eqs. (7.42) and (7.47). In the limit of infinite iterations, the distribution of the con-

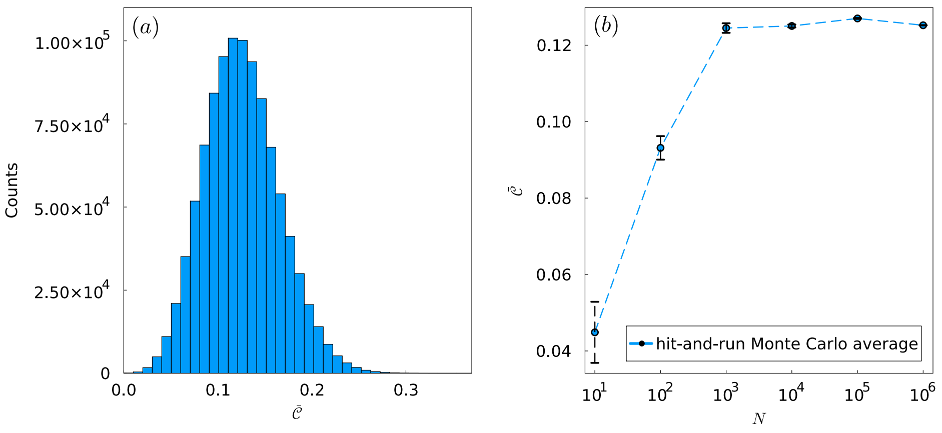


Figure 7.4: (a) Bar chart of the distribution of the concurrence for a sample of two-qubit density matrices after removing the states with concurrence zero; and (b) average initial concurrence as a function of the number of samples generated by the hit-and-run Monte Carlo algorithm. Vertical bars show the standard error of the mean. We see that the average approximately stabilizes after $N = 10^3$ samples. This is only true for the initial concurrence, as the average concurrence after each run of the purification protocol needs a higher resolution (empirically we find $N > 10^4$, especially for the MFI protocol). Plot (a) uses 10^6 density matrices, obtained by running the hit-and-run algorithm once.

currence assumes a specific form: The entanglement purification protocol has purified a certain number S of density matrices, while the other $N - S$ have concurrence zero. This means that the average concurrence in Eq. (7.40) is given by

$$\lim_{i\to\infty} \bar{\mathcal{C}}^{(i)} = \frac{S}{N}, \tag{7.57}$$

and the sample standard deviation in Eq. (7.41) reads

$$\begin{aligned}\lim_{i\to\infty} s_{\mathcal{C}}^{(i)} &= \sqrt{\frac{S}{N-1}\left(1-\frac{S}{N}\right)^2 + \frac{N-S}{N-1}\frac{S^2}{N^2}}\\ &= \sqrt{\frac{S}{N-1}\left(1-\frac{S}{N}\right)}.\end{aligned} \tag{7.58}$$

Now, let us assume that after the ith iteration quantum state ρ_j belonging to the set of purifiable density matrices has the concurrence

$$\mathcal{C}^{(i)}(\rho_j) = 1 - \epsilon_j^{(i)} \quad \text{with} \quad 1 > \epsilon_j^{(i)} > 0. \tag{7.59}$$

This model describes how far the concurrence of ρ_j from $\mathcal{C} = 1$ is, which is attained for $i \to \infty$. Now, we have

$$\bar{\mathcal{C}}^{(i)} = \frac{S}{N} - \frac{1}{N} \sum_{j=1}^{S} \epsilon_j^{(i)} < \lim_{i \to \infty} \bar{\mathcal{C}}^{(i)}, \tag{7.60}$$

i.e., the average concurrence is always smaller than S/N [see Eq. (7.57)]. Then the sample standard deviation reads

$$\begin{aligned} s_{\mathcal{C}}^{(i)} &= \sqrt{\frac{1}{N-1} \sum_{j=1}^{S} \left(1 - \epsilon_j^{(i)} - \bar{\mathcal{C}}^{(i)}\right)^2 + \frac{N-S}{N-1} \left(\bar{\mathcal{C}}^{(i)}\right)^2} \\ &= \sqrt{\frac{S - 2\epsilon_i}{N-1} \left(1 - \frac{S}{N}\right) - \frac{\epsilon_i^2}{N(N-1)} + \frac{\beta_i}{N-1}}, \end{aligned} \tag{7.61}$$

where

$$\epsilon_i = \sum_{j=1}^{S} \epsilon_j^{(i)}, \quad \beta_i = \sum_{j=1}^{S} \left(\epsilon_j^{(i)}\right)^2. \tag{7.62}$$

Since $\left(\epsilon_j^{(i)}\right)^2 < \epsilon_j^{(i)}$ for all j, we get $\beta_i < \epsilon_i$. Hence

$$-2\epsilon_i \left(1 - \frac{S}{N}\right) - \frac{\epsilon_i^2}{N} + \beta_i < -\epsilon_i \left(1 - \frac{2S}{N}\right) - \frac{\epsilon_i^2}{N}. \tag{7.63}$$

If $2S < N$, which is the case of the protocols discussed in 7.4.1, we obtain based on Eqs. (7.61) and (7.63) that

$$s_{\mathcal{C}}^{(i)} < \lim_{i \to \infty} s_{\mathcal{C}}^{(i)}, \tag{7.64}$$

i.e., the sample standard deviation reaches its maximum in the limit of infinite iterations. This property is shown in Fig. 7.3. Finally, we also want to visualize the distribution of the concurrence for the samples of density matrices that are generated by the hit-and-run algorithm using bar charts. We know that approximately 24.24 % of them are separable quantum states and have concurrence zero. These cannot be used by the purification protocols discussed in the main text. Therefore, we remove the bar chart corresponding to these matrices from our histograms, as they would be simply represented by a single enormous peak around zero. The distribution of the concurrence for the remaining randomly sampled two-qubit density matrices is given in Fig. 7.4. It seems that the concurrence resembles a (skew) Gaussian distribution, however, this is only a hypothesis and one should prove or disprove it by using methods developed in random matrix theory concerning density matrices [ŻPN&al11].

In Fig. 7.5, we see the iteration-based evolution of the distribution of the concurrence for $\mathcal{C} > 0$. Each column represents one of three different points in the purification protocols: (a), (d), (g) $i = 2$; (b), (e), (h) $i = 7$ and (c), (f), (i) $i = 15$. We show here nine plots, where the first row (a), (b), (c) represents the Bennett protocol, the second row (d),

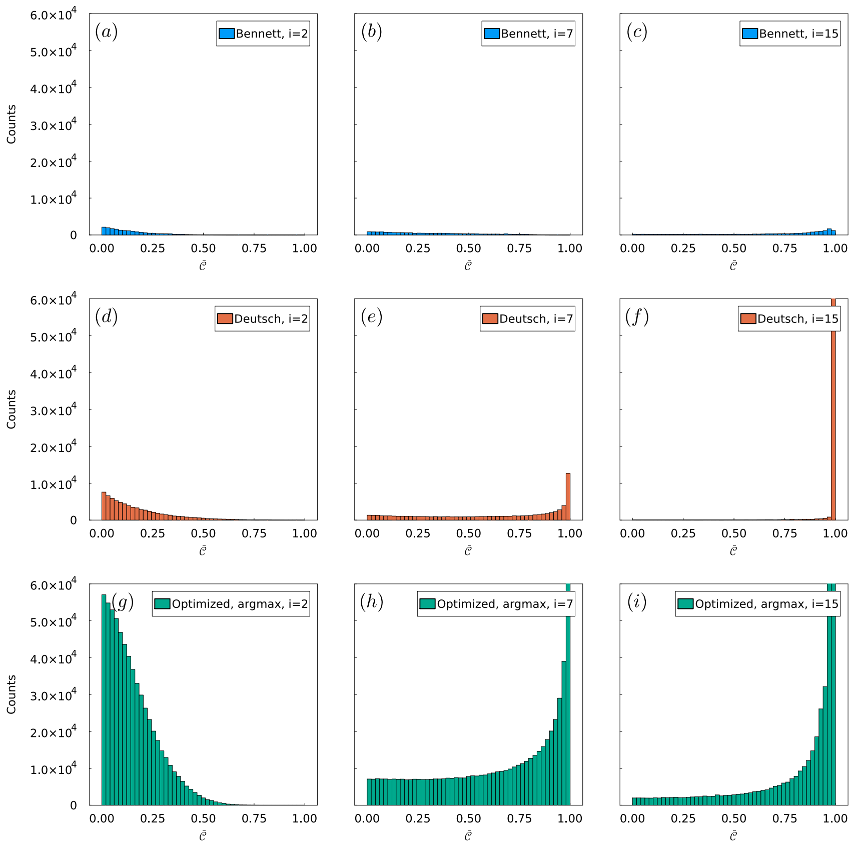


Figure 7.5: Bar charts representing 50 bins of the concurrence distribution for the Bennett – (a), (b), (c), Deutsch – (d), (e), (f), and optimized protocol [using the argmax policy in Eq. (7.32) and the projector Π in Eq. (7.54)] – (g), (h), (i), each one for the second, seventh, and fifteenth iteration – corresponding to the first, second and third column of the plots. States with concurrence zero have been removed to allow us to visualize the action of the protocols, as they skew the distribution due to their generally high number. As a consequence, the bar charts exhibit different heights, since the three protocols map different numbers of states to concurrence zero. Vertical axes have been set so to range between 0 and 6×10^4, so that the reader can visualize the growth of the number of states with concurrence equal to one on the right as the number of iterations grows. We see that the optimized protocol preserves the most states, whereas the number of purified states for the Deutsch and Bennett protocols is significantly lower. These bar charts use $N = 10^6$ samples of the concurrence (one run of the hit-and-run algorithm).

(e), (f) the MFI protocol and the third one (g), (h), (i) the CNOT protocol. We find, as expected, that the number of states with non-zero concurrence is much lower in the case of the Bennett protocol than for the other two, although every protocol improves the distribution of the concurrence towards $\bar{\mathcal{C}} = 1$.

7.7.3 Statistical evaluation of the fidelities

To provide the reader with a better understanding of how the protocols transform noisy entangled states into Bell states, we investigate here the evolution of the fidelities with respect to the stable fixed points. The study of fidelity is always conditioned on the properties of the input states. As we have already discussed in Sec. 7.2, these properties define different stable fixed points towards which the state is mapped by the protocol. Thus, the whole sample of input states will be separated into sets according to their stable fixed points and fidelities will be evaluated only in the corresponding set. This consideration allows us to exclude situations when the output state is separable and converges towards the maximally mixed state, which has an overlap of 0.25 with any Bell states and therefore would contribute to and at the same time skew the average output fidelity of the protocol. It is obvious that the set with the maximally mixed state as a fixed point will be not considered. This step is not necessary when one uses the concurrence.
For a density matrix ρ, we consider the overlaps

$$r_k = \operatorname{tr}\{\rho \left|k\right\rangle \left\langle k\right|\}, \tag{7.65}$$

where the states $k = 1, 2, 3, 4$ are the Bell states defined before Eq. (7.3). We then use them to define the output fidelity for the Bennett protocol [BBC&al93]

$$F_{\text{Bennett}}(\rho) = \begin{cases} r_1 & \text{if } 2r_1 > 1, \\ 0 & \text{otherwise.} \end{cases} \tag{7.66}$$

Similarly, we define the output fidelities for the other protocols using their respective conditions for purification. For the Deutsch protocol, we have

$$F^{(4)}_{\text{Deutsch}}(\rho) = \begin{cases} r_4 & \text{if } (2r_1 - 1)(1 - 2r_4) > 0, \\ 0 & \text{otherwise.} \end{cases} \tag{7.67}$$

$$F^{(2)}_{\text{Deutsch}}(\rho) = \begin{cases} r_2 & \text{if } (2r_2 - 1)(1 - 2r_3) > 0, \\ 0 & \text{otherwise.} \end{cases} \tag{7.68}$$

In the case of the MFI protocol, we get

$$F^{(1)}_{\text{MFI}}(\rho) = \begin{cases} r_1 & \text{if } (2r_1 - 1)(1 - 2r_3) > -(2\text{I}m[r_{13}])^2 - (2\text{R}e[r_{24}])^2, \\ 0 & \text{otherwise} \end{cases} \tag{7.69}$$

$$F^{(2)}_{\text{MFI}}(\rho) = \begin{cases} r_2 & \text{if } (2r_2 - 1)(1 - 2r_4) > -(2\text{I}m[r_{24}])^2 - (2\text{R}e[r_{13}])^2, \\ 0 & \text{otherwise,} \end{cases} \tag{7.70}$$

and finally CNOT protocol yields

$$F^{(4)}_{\text{CNOT}}(\rho) = \begin{cases} r_4 & \text{if } (2r_1 - 1)(1 - 2r_4) > -(2\text{I}m[r_{23}])^2 - (2\text{R}e[r_{14}])^2 \\ 0 & \text{otherwise} \end{cases} \tag{7.71}$$

$$F^{(2)}_{\text{CNOT}}(\rho) = \begin{cases} r_2 & \text{if } (2r_2 - 1)(1 - 2r_3) > -(2\text{I}m[r_{14}])^2 - (2\text{R}e[r_{23}])^2 \\ 0 & \text{otherwise.} \end{cases} \tag{7.72}$$

Output fidelities need to be computed according to Eqs. (7.66)–(7.72) for each sampled density matrix at each iteration of the corresponding protocol. The average fidelity is defined as

$$\bar{F}^{(k)}_{\text{protocol}}(\rho) = \frac{1}{N} \sum_{j=1}^{N} F^{(k)}_{\text{protocol}}(\rho_j), \tag{7.73}$$

where $k \in \{1, 2, 4\}$ is the label of the Bell states which are stable fixed points, and N is the sample size.
The average fidelities for the purifiable states with respect to all stable fixed points for the Bennett, Deutsch, MFI, and CNOT protocols are given in Fig. 7.6. We see that the number of states that can be brought to a stable fixed point by the MFI and CNOT protocols is significantly larger than those of the Bennett and Deutsch protocols. This is in accordance with the findings presented in the main text.
Prior knowledge of the input state is essential because most of the protocols currently available in the literature can only work if one knows that the states to be purified have certain underlying properties, which need to be known to avoid failure. In the subsequent discussion, we elucidate this argument by a simple demonstration. Let us now consider a general two-qubit density matrix that we wish to purify towards a Bell state. The Bennett protocol starts with a twirling operation, whose only goal is to bring the state ρ into the Werner form

$$\rho \overset{\text{twirling}}{\longrightarrow} \rho_W = r_1 \left|1\right\rangle \left\langle 1\right| + \frac{1 - r_1}{3} \left(I_4 - \left|1\right\rangle \left\langle 1\right|\right). \tag{7.74}$$

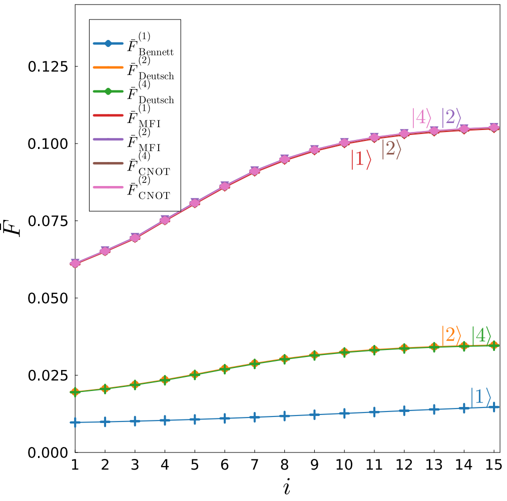


Figure 7.6: Average output fidelities [calculated using Eqs. (7.66)–(7.72), and (7.73) for the different fixed points] with respect to the attractors of the different protocols as a function of the number of iterations. For the Bennett protocol, $|1\rangle$ is the only fixed point. For the other protocols, the attractors are $|2\rangle$ and $|4\rangle$ (Deutsch), $|1\rangle$ and $|2\rangle$ (MFI) and $|4\rangle$ and $|2\rangle$ (CNOT). We see that for each of their attractors, the shares of purifiable states of both the MFI and the CNOT protocols are significantly larger than the ones of Bennett and Deutsch. The plot is obtained by running ten simulations of the hit-and-run algorithm, each one with $N = 10^6$ samples.

This operation, however, does not guarantee the success of the purification because the Bennett protocol can only work if the condition $r_1 > 0.5$ is fulfilled. If it does, then the Bennett protocol in the asymptotic limit brings this state to the Bell state $|1\rangle$. In all other cases, the purification protocol fails and leads to a mixed state. The infinite-limit output of the Bennett protocol for a general two-qubit density matrix is therefore a classical mixture of the purified states (with concurrence one) and the mixed states for which the protocol failed (with concurrence zero). It turns out that for most of the density matrices, the protocol fails, i.e., the twirling operation maps them to Werner states with $r_1 < 0.5$, which is why the blue curve in Fig. 7.1 has such small values. The asymptotic value of the curve represents exactly the fraction of states that can be purified. The output of the Bennett protocol with asymptote $\bar{\mathcal{C}}^{(\infty)}_{\mathrm{Bennett}}$ for a general random density matrix in the limit of infinite iterations will be approximately

$$\rho_{\mathrm{Bennett}} = \bar{\mathcal{C}}^{(\infty)}_{\mathrm{Bennett}} |1\rangle\langle 1| + \frac{(1-\bar{\mathcal{C}}^{(\infty)}_{\mathrm{Bennett}})}{3}(I_4 - |1\rangle\langle 1|) \tag{7.75}$$

which is again a Werner state, but with fidelity $\bar{\mathcal{C}}^{(\infty)}_{\text{Bennett}}$. This presents a limitation of using the Bennett protocol on larger classes of states, especially if their generation cannot be controlled in such a way that the condition $r_1 > 0.5$ is met. This implies that for the Bennett protocol information about the input state has to be given otherwise we get a very noisy entangled output state even after infinitely many iterations.
The Deutsch protocol, unlike the Bennett, has three stable fixed points with two Bell states, which is often not mentioned in the literature [DB07], but after the appearance of the proposal this has been thoroughly investigated in Ref. [Mac98]. Now, one obtains a classical mixture of three states with weights defined by $\bar{\mathcal{C}}^{(\infty)}_{\text{Deutsch}}$, where the states $|2\rangle\langle 2|$ and $|4\rangle\langle 4|$ have the same weights as it is shown in Fig. 7.1. This leads also to a noisy entangled state. In conclusion, one cannot start with an unknown density matrix and just run the protocols without having some prior information beforehand, because, as we see, the protocols mostly fail if certain conditions are not met. These conditions, as we have shown, may be very restrictive or more relaxed, in which case a broader class of input states can be successfully purified.

8 Estimation of observables in quantum systems

Disclaimer: This chapter contains an analysis and a partial review of some relevant sampling techniques used in quantum circuits, in particular sampling and estimation methods that use the so-called LCU approach (Linear Combination of Unitaries) [CW12]. These methods are of particular importance in quantum simulation [GAN14; Ort&al01], but they can also be applied to various estimation problems. A more in-depth review of the topic, with a detailed analysis of the advantages and disadvantages of implementing LCU algorithms in different ways, is provided in Ref. [Cha24]. The manuscript was written autonomously by the author, who thanks both Jószef Zsolt Bernad and Felix Motzoi for the relevant feedback. Relevant parts of this chapter, with appropriate modifications, have been included in a more comprehensive publication, which can be found in Ref. [Pre&al25]. Other co-authors listed therein did not contribute to the writing of this chapter.

Cost functions used to optimize the quantum dynamics in optimal quantum control and variational quantum circuits can be represented as mean values of specific observables. However, how to measure a certain observable on an arbitrary quantum system is not always obvious. One of the possible methods to measure such quantities is given by LCU algorithms. In this section, we discuss some of the properties of such algorithms and some advantages and disadvantages of their implementation. For a more detailed analysis, we refer, e.g., to Refs. [Cha24] and [Pre&al25].

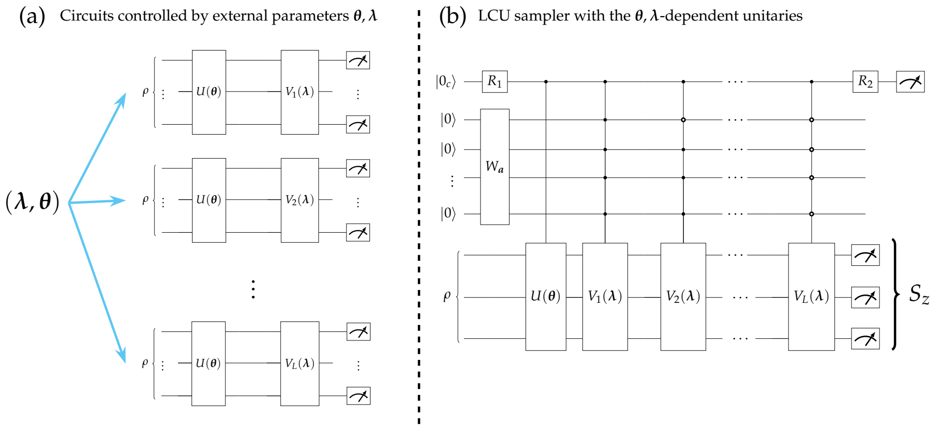
8.1 Introduction

A quantum algorithm is usually implemented as a family of one or multiple quantum circuits, which represent physical experiments on one or more of the available quantum computing platforms, such as superconducting quantum circuits [KKY&al19; BGG&al21], trapped-ions [HRB08] or Rydberg atoms [SWM10]. In these models, a quantum state is first prepared, evolves under the action of unitary operations and is then measured. Qubits can be measured in between unitary operations [DeC&al23] (and subsequently reset if needed), so that more complex maps involving mixed states can also be implemented in quantum algorithms. By executing a quantum algorithm multiple times, we can collect a statistics about the possible different outcomes. In variational quantum algorithms [Cer&al21a] the statistics is used to estimate a cost function, which is then

optimized with respect to variational parameters using classical optimization methods. As an example, mean values of arbitrary observables are estimated by sampling from different quantum circuits, each one representing an element of an operator basis – e.g., the Pauli basis –[SOG&al02; FGG14; Per&al14; Bia&al17; Koc&al14; Jer&al23a; Sch&al22; PCM22; CKA21; WIW&al22; HKP20; DMS22]. The mean values of the single elements of the operator basis can be evaluated using multiple copies of the same circuit. In the most straightforward implementation the scaling is linear in the number of circuits L: $O(L/\epsilon^2)$. If we can implement a shadow tomography model [HKP20; HKP21], the scaling can be dramatically improved at the cost of larger errors $O(\log(L)/\epsilon^4)$, whereas using amplitude amplification the scaling becomes sub-linear, at the cost of having to implement the amplitude amplification routine [HWM&al22] $O(\sqrt{L}/\epsilon)$. Once the mean values have been estimated, they are summed together with appropriate coefficients. This further increases the variance linearly in the number of terms [BBN19].
Linear combinations of measurements represent the standard approach to the estimation of observables [Hay17] and they also appear in situations where the expected value of an observable is averaged over other parameters native to the quantum system. For instance, there are cases in which mixtures of classical and quantum expectation values need to be computed, using, e.g., quasi-Monte Carlo approaches. Quantum machine learning requires the computation of averages over data sets [Bia&al17; Jer&al23a; SAC&al21]. Quantum control and optimization, on the other hand, for example in the context of so-called robust [Sch&al22] or adaptive control/meta-optimization [PCM22; CKA21], require to compute averages of cost functions over a certain parameter space [ORC&al22; DWM22] (robustness). Control pulses obtained this way are potentially robust against experimental parameter drifts. Another example of estimators that use both classical and quantum sampling are (stochastic) parameter-shift rules [LYP&al17; WIW&al22], which are used to evaluate gradients of quantum cost-functions sampled using variational quantum circuits.
Several quantum algorithms make instead use of linear combination of unitary operations [SOG&al02; CW12; CKS17; Cer&al21a; CS17] to create superpositions. The question is whether this implementation can be beneficial in specific contexts. In this work, we compare Linear Combination of Unitaries-based (LCU) estimators to the standard estimator for quantum observables and determine the conditions in which the implementation of the former is detrimental or beneficial, i.e., where it provides us with a speed-up over the classical counterpart thanks to amplitude estimation.

8.2 Problem statement

Sampling from one or multiple quantum circuits involves computing linear combinations of binary counts corresponding to different outputs. An example is given by QUBO problems [Koc&al14], where a quantity, which is in a quadratic form with binary arguments, needs to be sampled from various quantum systems. In the case of variational quantum eigensolvers [Per&al14], the goal is to minimize the energy of a Hamiltonian given a certain input state. The n-qubit observable as a whole is usually not available



(a) Circuits controlled by external parameters $\boldsymbol{\theta}, \boldsymbol{\lambda}$
$(\boldsymbol{\lambda}, \boldsymbol{\theta})$
ρ
$U(\boldsymbol{\theta})$
$V_1(\boldsymbol{\lambda})$
$V_2(\boldsymbol{\lambda})$
$V_L(\boldsymbol{\lambda})$
(b) LCU sampler with the $\boldsymbol{\theta}, \boldsymbol{\lambda}$-dependent unitaries
$|0_c\rangle$
$|0\rangle$
R_1
R_2
W_a
S_z

Figure 8.1: In the context of the aforementioned sampling problems we consider two different approaches: (a) the trivial estimator (TE), which prepares various circuits with indices $i = 1, ..., L$ that have ρ as input density matrix and an arbitrary unitary evolution operator $U(\boldsymbol{\theta})$, $\boldsymbol{\theta} \in \mathbb{R}^{n_p}$. Different circuits are characterized by unitaries $V_1, V_2, ..., V_L$, where $\boldsymbol{\lambda}$ is a parameter sampled from an external distribution $\boldsymbol{\lambda} \sim P$. In (b) we see the LCU approach, in which the unitaries are multi-controlled on a register of qubits that encodes the linear combination through a unitary matrix $W_{\boldsymbol{a}}$. The unitaries $W_{\boldsymbol{a}}$, R_1 and R_2 encode the properties of the cost function to implement.

directly but it can be represented as a linear combination of hermitian matrices, e.g., Pauli strings P_i, which can potentially be measured using quantum circuits. We consider the observable:

$$\mathcal{O} = \sum_{i=1}^{L} a_i P_i, \quad a_i \in \mathbb{R}. \tag{8.1}$$

We limit ourselves to the case in which $\mathcal{O}$ can be realized by considering only one element of the generalized Pauli group, in which case the expression above becomes:

$$\mathcal{O} = \sum_{i=1}^{L} a_i V_i Z_{\text{prod}} V_i^{\dagger}, \tag{8.2}$$

where $Z_{\text{prod}} = \bigotimes_{i=1}^{N} \sigma_z^{(i)}$ is the n-qubit σ_z operator and $V_i, i = 1, ..., L$ are appropriate unitary matrices – e.g., they map Z_{prod} to other elements of the Pauli basis. The choice of Z_{prod} is arbitrary: another possibility is to map the operator to a single-qubit σ_z operator $Z_{\text{prod}}^{(i)} = \mathbb{I} \otimes ... \otimes \sigma_z^{(i)} \otimes \mathbb{I}$ via CNOT operations, but any generalized Pauli operator can be used in principle, as matrices mapping generalized Pauli operator to each other can all be generated using CNOT, Hadamard and Phase gates [BBC&al95; CW12]. The mean value of the observable $\mathcal{O}$ is computed with respect to a quantum state ρ, such that for the expected value of $\mathcal{O}$ we can write:

$$\langle \mathcal{O} \rangle = \operatorname{tr}\{\rho \mathcal{O}\} = \sum_{i=1}^{L} a_i \operatorname{tr}\left\{\rho V_i Z_{\text{prod}} V_i^{\dagger}\right\}. \tag{8.3}$$

Using this representation, we can implement the unitaries $V_i, i = 1, \dots, L$ on different quantum circuits and then measure the register of qubits in the computational basis. We assume that the input state ρ undergoes a parametric evolution generated by a variational unitary. As a result, the expression

$$\langle \mathcal{O}(\boldsymbol{\theta}) \rangle = \sum_{i=1}^{L} a_i \operatorname{tr}\left\{U(\boldsymbol{\theta}) \rho U^{\dagger}(\boldsymbol{\theta}) V_i Z_{\text{prod}} V_i^{\dagger}\right\} \tag{8.4}$$

encodes an energy minimization problem in up to L different quantum circuits using real parameters $\boldsymbol{\theta} \in \mathbb{R}^{n_p}$ and a n-qubit variational quantum circuit $U(\boldsymbol{\theta}) \in \mathrm{U}(d)$, $d = 2^n$. Let us assume that any two different Pauli strings considered in Eq. (8.1) commute. If this is the case, they can be estimated within the same circuit run, which significantly reduces the amount of measurements needed – if L of them commute, estimating their mean values scales as in $O(\lceil \log(L) \rceil / \epsilon^2)$ [HWM&al22]. If they do not commute, up to L circuits need to be executed. Moreover, cost functions for variational circuits are also averaged over additional external parameters, where a relevant parameter $\boldsymbol{\lambda} \sim P$ is sampled from a probability distribution P:

$$C(\boldsymbol{\theta}) = \mathbb{E}_{\boldsymbol{\lambda} \sim P}\left[\langle \mathcal{O}(\boldsymbol{\theta}, \boldsymbol{\lambda}) \rangle\right]. \tag{8.5}$$

Therefore, there are two types of parameters: *meta-parameters*, denoted by $\boldsymbol{\lambda}$, which are sampled and averaged over – an example of this is given by Monte-Carlo sampling, where we want to average a value over a data set of parameters – and *variational parameters*, denoted by $\boldsymbol{\theta}$ which are generally used for numerical optimization in the context of variational algorithms.
Sampling using Eq. (8.4) is not the only option to evaluate the mean value of the observable. We can construct an estimator for $\langle \mathcal{O}(\boldsymbol{\theta}) \rangle$ by first constructing estimators for L different circuits. A different estimator can be constructed based on linear combination of unitaries [SOG&al02; CW12] using a circuit that forks [PSF&al19] the state evolution in different directions based on controlled operations. Our goal is to analyze the behaviour of such an estimator compared to the standard sequential procedure that uses L circuits.

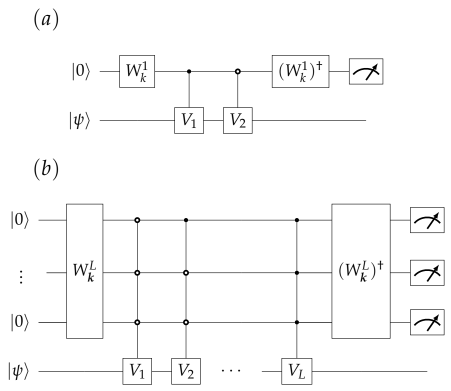


Figure 8.2: (a) Circuit implementing the sum of two unitaries V_1 and V_2 on a quantum computer using one control qubit and (b) circuit implementing the sum of L unitaries using up to $\lceil \log(L) \rceil$ qubits (both are based on the circuits given in Ref. [CW12]). Upon measuring the control qubit in either 0 or 1, the whole state collapses in a state proportional to either $V_1 + V_2$ or $V_1 - V_2$. The LCU can therefore be used to probabilistically implement arbitrary operators acting on a state $|\psi\rangle$, as those found, e.g., in Hamiltonian simulation. In its generalized implementation (b), the LCU generates all possible combinations using coefficients $\boldsymbol{k}$ of sums and differences of L unitaries. The linear combination with only positive terms is mapped to the zero state, however the probability of measuring it decreases with $1/L$.

8.3 Linear combinations of estimates

8.3.1 Trivial estimator (TE): linear combinations of measurements

Our goal is to construct an estimator $\tilde{C}$ that, using the measurement outcomes collected from the quantum circuits, can successfully approximate C in Eq. (8.5). Let us consider a collection of circuits numbered 1 to L, each one implementing a unitary $V_1, ..., V_L$ that we use to perform a measurement of Z_{prod}. We refer to this estimator as the trivial estimator (TE). This is a straightforward approach in most of the sampling problems in variational quantum circuits, so we use this name just for clarity. The principle is simple: we have different circuits that are initialized independently (i.i.d). On each one of these circuits we prepare an identical initial state ρ. We first limit ourselves to the case in which the coefficients a_i are all non-negative. Formally, we consider first an estimator denoted by the pair $\left(M^{(i)}_{j_1 j_2 \cdots j_n}, \tilde{C}\right), i = 1, ..., L$ for a state ρ [Hay17], where $M^{(i)}_{j_1 j_2 \cdots j_n}$ are projectors of the form:

$$M^{(i)}_{j_1 j_2 \cdots j_n} = V_i \left(\Pi_{j_1} \otimes \Pi_{j_2} \otimes \dots \Pi_{j_n}\right) V_i^\dagger, \tag{8.6}$$

where $j_1, j_2, \dots j_n \in \{0,1\}$ and

$$\Pi_0 = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}, \quad \Pi_1 = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix}. \tag{8.7}$$

Then, we have

$$V_i Z_{\text{prod}} V_i^\dagger = \sum_{j_1, j_2, \dots, j_n = 0,1} (-1)^{\sum_{l=1}^n j_l} M^{(i)}_{j_1 j_2 \dots j_n}. \tag{8.8}$$

This allows us to estimate $m_i = \operatorname{tr}\left\{\rho V_i Z_{\text{prod}} V_i^\dagger\right\}$ for a given ρ and thus

$$\langle \mathcal{O} \rangle_\rho = \sum_{i=1}^{L} a_i \sum_{j_1, j_2, \dots, j_n = 0,1} (-1)^{\sum_{l=1}^n j_l} \operatorname{tr}\left\{\rho M^{(i)}_{j_1 j_2 \dots j_n}\right\}. \tag{8.9}$$

Each outcome of a circuit measurement corresponds to a binary string $x^{(i)}_{k_{(i)}} \in 1, \dots, 2^n$ and is weighted with either a coefficient plus or minus one, depending on its binary Hamming weight:

$$b(x) = \sum_{l=0}^{n-1} j_l(x), \tag{8.10}$$

where $x = \sum_l j_l(x) 2^l$ and $j_l(x) \in \{0,1\}$ are the binary digits of x. Finally, $\tilde{C}$ is the map from the measurement data set to the real line. Our first choice is given by the TE estimator:

$$\tilde{C}_{\text{TE}} = \sum_{i=1}^{L} \frac{a_i}{n_s^{(i)}} \sum_{k_i=1}^{n_s^{(i)}} (-1)^{b(x^{(i)}_{k_i})}. \tag{8.11}$$

We consider here the estimation of the expected values of L Pauli strings $P_1, \dots, P_L$. The mean value of a Pauli string for a quantum state ρ has the following variance:

$$\sigma^2_{P_i} = \langle P_i^2 \rangle - \langle P_i \rangle^2, \tag{8.12}$$

and due to $P_i^2 = \mathbb{I}$ for any Pauli string, we have:

$$\sigma^2_{P_i} = 1 - m_i^2, \tag{8.13}$$

where $m_i = \operatorname{Tr}\{\rho P_i\} \in [-1, 1]$. Eq. (8.13) can be considered as the variance of a Rademacher variable, which can be transformed into a Bernoulli variable with mean $p_i = \frac{1}{2}(m_i + 1)$ with variance

$$\sigma^2_{p_i} = p_i(1 - p_i). \tag{8.14}$$

Eq. (8.14) can be also seen as the variance of a projective measurement Π_i, with $p_i = \mathrm{Tr}\{\Pi_i \rho\} \in [0,1]$. Any TE estimator draws measurement values $x^{(j)}_{k_{(1)}}, \ldots, x^{(j)}_{k_{(L)}}, 1 \le j \le n_s^{(i)}$ and $1 \le i \le L$ from circuits $V_1, \ldots, V_L$ acting upon Hilbert spaces $\mathcal{H} = \mathcal{H}^{(1)} \otimes \mathcal{H}^{(2)} \otimes \ldots \otimes \mathcal{H}^{(L)}$ with input density matrices ρ. Each circuit is used to estimate the mean value of Z_{prod} using $n_s^{(i)}$ shots. Hence, the TE estimator has a variance of

$$\mathrm{Var}(\tilde{C}_{\mathrm{TE}}) = \sum_{i=1}^{L} \frac{a_i^2}{n_s^{(i)}} p_i (1 - p_i). \tag{8.15}$$

Due to the absence of entanglement between the circuits, there are no correlations between the estimates and we can write:

$$\begin{aligned} \sigma^2_{\tilde{C}_{\mathrm{TE}}} &= \sum_{i=1}^{L} \frac{a_i^2}{n_s^{(i)}} \left(\mathrm{Tr}\{\rho \Pi_i^2\} - \mathrm{tr}\{\rho \Pi_i\}^2 \right) \\ &= \sum_{i=1}^{L} \frac{a_i^2}{n_s^{(i)}} p_i (1 - p_i). \end{aligned} \tag{8.16}$$

If we set $n_s^{(i)} \ge \min_{i=1,\ldots,L} \left[n_s^{(i)} \right] = n_s$ for all $i = 1, \ldots, L$ and use Chebyshev inequality [Was10], we can lower-bound the number of shots per circuit as $n_s \ge \frac{L a_{\mathrm{max}}^2}{4\epsilon^2}$, where ϵ is the precision of the estimation. For a total of L circuits, this results in a circuit sampling complexity of $O[L^2 a_{\mathrm{max}}^2 / (4\epsilon^2)]$, where $a_{\mathrm{max}} = \max_{i=1,..,L} [a_i]$ – see also Refs. [WHT15; BBN19; RBM18].

8.3.2 Linear combination of unitaries

The Linear Combination of Unitaries [CW12] is a quantum algorithm that allows to perform sums and differences of unitaries on a quantum computer. In its simplest form it uses the operator:

$$W_k = \begin{pmatrix} \sqrt{\frac{k}{k+1}} & -\sqrt{\frac{1}{k+1}} \\ \sqrt{\frac{1}{k+1}} & \sqrt{\frac{k}{k+1}} \end{pmatrix}, \tag{8.17}$$

to create a superposition between $|0\rangle$ and $|1\rangle$. Afterwards, conditional operations V_1 and V_2 are applied on an arbitrary state $|\psi\rangle$, followed by a second operation $W_k^\dagger$. This leads to a superposition of $V_1 + V_2$ and $V_1 - V_2$ with different probability amplitudes – see also Fig. 8.2 (a). In particular, we see that the probability of finding measuring the control qubit in its $|1\rangle$ state is given by

$$p_1 = \frac{k}{(k+1)^2} \| (V_1 - V_2) |\psi\rangle \|^2 \le \frac{4k}{(k+1)^2}, \tag{8.18}$$

so that the algorithm implements $\frac{1}{\sqrt{\alpha_+}}(V_1 + V_2)\,|\psi\rangle$ for $k \longmapsto \infty$ and $\frac{1}{\sqrt{\alpha_-}}(V_1 - V_2)\,|\psi\rangle$ for $k \longmapsto 0$, where $\alpha_\pm = \left|\langle\psi|\,(V_1 \pm V_2)^\dagger(V_1 \pm V_2)\,|\psi\rangle\right|$. If the goal is to apply the sum of L operators, one either applies the circuit represented in Fig. 8.2 (a) recursively or uses a circuit with multi-controlled gates – see Fig. 8.2 (b) and also Appendix in Ref. [CW12]. In this case the success probability will be always smaller than p_1.
The ancillas that implement the summation procedure can also be encoded in a larger number of qubits [MKW17; Ara&al21]. If, e.g., L qubits instead of $\lceil\log(L)\rceil$ qubits are used, the implementation requires only single-qubit controlled gates and no multi-controlled gate. The overall depth of the circuit is then lower [BBC&al95], but the number of control qubits is exponentially larger.

8.3.3 Estimator with linear combination of unitaries (LCU)

Variants of the LCU circuit have been applied to estimation problems (with some claims of speedup) [PSF&al19; SOG&al02]. Our goal here is to characterize the variance properties of these two estimators. Ref. [Cha24] gives a more in-depth overview of different variants of the algorithm, including continuous variables and integrals. We consider a circuit of the type given in Fig. 8.1 (b). The circuit is a LCU circuit that uses a unitary $W_{\boldsymbol{a}}$ to generate the state

$$W_{\boldsymbol{a}}\,|0\rangle = |a\rangle \tag{8.19}$$

on the LCU register, which uses $O(\lceil\log(L)\rceil)$ qubits [BBC&al95], where

$$|a\rangle = \frac{1}{\sqrt{\|\boldsymbol{a}\|_1}} \sum_{i=1}^{L} \sqrt{|a_i|}\,|i-1\rangle \tag{8.20}$$

for $\|\boldsymbol{a}\|_1 = \sum_{i=1}^{L} |a_i|$ and where the weights of the observable are now written in vector form $\boldsymbol{a} = \sum_{i=1}^{L} a_i \boldsymbol{e}_i$, where $\boldsymbol{e}_i$ are unit vectors. We refer to the normalized probabilities associated with each amplitude of the state as w_i:

$$w_i = |\langle i|a\rangle|^2 = \frac{|a_i|}{\sum_{l=1}^{L} |a_l|}. \tag{8.21}$$

It is clear than any convex combination of such weights with values $0 \le p_i \le 1$ lies itself between zero and one. The circuit measures the upper control qubit and the n system qubits (analogously to the case of the TE estimator). We calculate the mean and variance of the circuit output measurements. We consider here the case in which measurements are drawn from the computational basis.

Theorem 1 (Mean and variance of the LCU estimator)**.** *The expected value and variance* *of the observable* $\Pi_{LCU} = |0_c\rangle\langle 0_c| \otimes \mathbb{I}_L \otimes \Pi$ *measured with the LCU circuit are:*

$$\bar{p} = \sum_{i=1}^{L} w_i p_i, \tag{8.22}$$

$$\sigma_{\bar{p}}^2 = \sum_{i=1}^{L} w_i p_i - \left(\sum_{i=1}^{L} w_i p_i\right)^2 = \bar{p}(1-\bar{p}), \tag{8.23}$$

with $p_i = \frac{1}{2}\operatorname{tr}\left\{V_i \rho V_i^{\dagger} \Pi\right\}$. *If instead the observable* $Z_{LCU} = |0_c\rangle\langle 0_c| \otimes \mathbb{I}_L \otimes Z_{prod}$ *is measured, then the corresponding expected value and variance are:*

$$\bar{m} = \sum_{i=1}^{L} w_i m_i, \tag{8.24}$$

$$\sigma_{\bar{m}}^2 = 1 - \left(\sum_{i=1}^{L} w_i m_i\right)^2 = 1 - \bar{m}^2, \tag{8.25}$$

with $m_i = \operatorname{tr}\left\{V_i \rho V_i^{\dagger} Z_{prod}\right\}$.

Proof. The initial input state of the LCU circuit is given by the tensor product of the n-qubit input density matrix, the zero state of the LCU register and the zero state of the control qubit:

$$\rho_{\text{in}} = |0_c\rangle\langle 0_c| \otimes \underbrace{|0\rangle\langle 0| \otimes \ldots \otimes |0\rangle\langle 0|}_{n_{\text{LCU}}} \otimes \rho, \tag{8.26}$$

where $n_{\text{LCU}} = \lceil \log L \rceil$. The evolved density matrix of the LCU circuit is given by

$$\rho_{\text{out}} = \begin{pmatrix} A & B \\ B^{\dagger} & C \end{pmatrix}, \tag{8.27}$$

where A, C, B are given by

$$A = \sum_{j=1}^{L}\sum_{i=1}^{L} \frac{\sqrt{w_i}\sqrt{w_j}}{2} \left(|0_c\rangle\langle 0_c| \otimes |i\rangle\langle j| \otimes \rho\right), \tag{8.28}$$

$$B = \sum_{j=1}^{L}\sum_{i=1}^{L} \frac{\sqrt{w_i}\sqrt{w_j}}{2} \left(|1_c\rangle\langle 0_c| \otimes |i\rangle\langle j| \otimes (V_i \rho)\right), \tag{8.29}$$

$$C = \sum_{j=1}^{L}\sum_{i=1}^{L} \frac{\sqrt{w_i}\sqrt{w_j}}{2}\left(|1_c\rangle\langle 1_c| \otimes |i\rangle\langle j| \otimes V_i \rho V_j^\dagger\right). \tag{8.30}$$

The entries of A, B, C may change depending on the values of the single-qubit unitary gates R_1, R_2 – see Fig. 8.1 – acting on the first control lines. We consider here the case in which the first gate in the control line corresponds to the Hadamard gate $R_1 = H$ and the second one to the X gate $R_2 = X$. Hence, the output probability distribution corresponding to the projector Π acting on the subspace of the density matrix ρ and the measurement line is given by

$$\bar{p} = \mathrm{Tr}\{\rho_{\text{out}}\Pi_{\text{LCU}}\}, \tag{8.31}$$

where $\Pi_{\text{LCU}} = \Pi_1 \otimes \mathbb{I}_L \otimes \Pi$, $\Pi_1 = |1_c\rangle\langle 1_c|$. The value of $\bar{p}$ is given by

$$\bar{p} = \sum_{i=1}^{L} \frac{w_i}{2}\mathrm{Tr}\left\{V_i \rho V_i^\dagger \Pi\right\} = \sum_{i=1}^{L} w_i p_i. \tag{8.32}$$

Each mean value p_i represents the probability of success of a Bernoulli distribution, and it lies between 0 and 1, which introduces constraints on the values p_i that depend on the different unitaries V_i:

$$p_i = \frac{1}{2}\mathrm{Tr}\left\{V_i \rho V_i^\dagger \Pi\right\}. \tag{8.33}$$

The variance of the estimator is given by

$$\sigma_{\bar{p}}^2 = \mathrm{Tr}\left\{\rho_{\text{out}}\Pi_{\text{LCU}}^2\right\} - \mathrm{Tr}\{\rho_{\text{out}}\Pi_{\text{LCU}}\}^2, \tag{8.34}$$

which corresponds to the variance of a Bernoulli-type distribution, because $\Pi_{\text{LCU}}^2 = \Pi_{\text{LCU}}$. Using Eq. (8.32), we have

$$\sigma_{\bar{p}}^2 = \bar{p}(1-\bar{p}) = \sum_{i=1}^{L} w_i p_i - \left(\sum_{i=1}^{L} w_i p_i\right)^2. \tag{8.35}$$

The mean value of the estimator is bounded between zero and one, whereas the variance has its maximum at $\sigma_{\bar{p}}^2 = \frac{1}{4}$ when $\bar{p} = \frac{1}{2}$. Now we turn to the estimation of Z_{prod}. If instead of the projective measurement, a measurement of Z_{prod} is carried out on the n-qubit subspace, on the whole Hilbert space we will be dealing with the measurement of the observable $Z_{\text{LCU}} = \Pi_1 \otimes \mathbb{I}_L \otimes Z_{\text{prod}}$. In this case we have $m_i = \mathrm{Tr}\left\{V_i \rho V_i^\dagger Z_{\text{prod}}\right\}$

and the mean of the measurement:

$$\bar{m} = \sum_{i=1}^{L} w_i m_i, \tag{8.36}$$

as well as the variance

$$\sigma_{\bar{m}}^2 = 1 - \left(\sum_{i=1}^{L} w_i m_i \right)^2, \tag{8.37}$$

where we used the property $Z_{\mathrm{LCU}}^2 = \Pi_1 \otimes \mathbb{I}_L \otimes \mathbb{I}$ and $\sum_{i=1}^{L} w_i \operatorname{tr}\left\{V_i \rho V_i^\dagger\right\} = 1$. □

We can define a new estimator using the framework described before: Let $\bar{x}^{(j)}, j = 1, ..., n_s$ be shots of the LCU circuit that are sampled by measuring the control qubit in 0 and the corresponding $S_z = Z_{\mathrm{prod}}$ operator, then

$$\tilde{C}_{\mathrm{LCU}} = \frac{\|\boldsymbol{a}\|_1}{n_s} \sum_{j=1}^{n_s} (-1)^{b(\bar{x}^{(j)})}, \tag{8.38}$$

with variance

$$\mathrm{Var}(\tilde{C}_{\mathrm{LCU}}) = \frac{\|\boldsymbol{a}\|_1^2}{n_s} \sigma_{\bar{m}}^2 \leq \frac{\|\boldsymbol{a}\|_1^2}{n_s}. \tag{8.39}$$

Theorem 2. *Let $\forall i, j, 1 \leq i, j \leq L : 0 \leq p_i, p_j \leq 1$ and $\forall i, j, 1 \leq i, j \leq L : 0 \leq w_i, w_j \leq 1$, $\sum_{i=1}^{L} w_i = 1$, and $X_i, X_j : \Omega \mapsto \{0, 1\}$ – Ω is a probability space – are random variables with mean p_i and p_j respectively and for which $\forall i = 1, ..., L$: $\mathbb{E}\left[X_i^2\right] = \mathbb{E}\left[X_i\right] = p_i$. Then we have:*

$$\sum_{i=1}^{L} \sum_{j=1}^{L} w_i w_j \, \mathbb{E}[X_i X_j] \leq \sum_{i=1}^{L} w_i p_i. \tag{8.40}$$

Proof. By the Cauchy-Schwarz inequality, we know that

$$|\mathbb{E}[X_i X_j]| \leq \sqrt{\mathbb{E}[X_j]} \sqrt{\mathbb{E}[X_j]} = \sqrt{p_i} \sqrt{p_j}, \tag{8.41}$$

where we used the fact that $\mathbb{E}[X_i^2] = \mathbb{E}\left[X_i\right] = p_i$ and so

$$\begin{aligned} &\sum_{i=1}^{L} \sum_{j=1}^{L} w_i w_k \, \mathbb{E}[X_i X_j] \leq \\ &\leq \sum_{i=1}^{L} \sum_{j=1}^{L} w_i w_j \sqrt{p_i} \sqrt{p_j} = \left(\sum_{i=1}^{L} w_i \sqrt{p_i} \right)^2. \end{aligned} \tag{8.42}$$

Now we employ one of the generalized-mean inequalities [Bul03], i.e., we use the fact that for $p < q$:

$$\left(\sum_{i=1}^{L} w_i p_i^p \right)^{\frac{1}{p}} \leq \left(\sum_{i=1}^{L} w_i p_i^q \right)^{\frac{1}{q}}, \tag{8.43}$$

setting $p = \frac{1}{2}$ and $q = 1$. After applying this equation to Eq. (8.42), we have

$$\sum_{i=1}^{L} \sum_{j=1}^{L} w_i w_k \, \mathbb{E}[X_i X_j] \leq \sum_{i=1}^{L} w_i p_i. \tag{8.44}$$

□

Therefore, we have that for such variables $X_1, ..., X_L$ the variance of their linear combination can be bounded from above as follows:

$$\sum_{i=1}^{L} \sum_{j=1}^{L} w_i w_j \mathrm{Cov}(X_i, X_j) \leq \sum_{i,j=1}^{L} w_i w_j p_i (1 - p_j). \tag{8.45}$$

The Cauchy-Schwarz inequality provides us with the upper bound for the variance of the linear combination, i.e.:

$$\mathrm{Var}\left(\sum_{i=1}^{L} w_i X_i \right) \leq \left(\sum_{i=1}^{L} w_i \sqrt{p_i (1 - p_i)} \right)^2. \tag{8.46}$$

If instead we consider shifted estimates Y_i in the interval $I = [-1, 1]$, we obtain that the covariance is always bounded by one, which is the first term in Eq. (8.25). Thus, also in this case we have:

$$\sum_{i=1}^{L} \sum_{j=1}^{L} w_i w_j \mathrm{Cov}(Y_i, Y_j) \leq 1 - \left(\sum_{i=1}^{L} w_i m_i \right)^2. \tag{8.47}$$

The maximum variance for this case can be derived by applying the same principle as the one used in Eq. (8.46). We can see immediately that the variance of the LCU sampler is always larger than the variance of the TE sampler in any circumstance. For the sampling of a bounded quantity, such the expected value of a Pauli string with respect to variational angles, the variance of the LCU sampler is $O(1)$ and its sampling complexity is $O(1/\epsilon^2)$. Vice versa, the variance of the TE sampler for the mean is $O(1/L)$, but L circuits need to be evaluated, so the overall sampling complexity is $O(1/\epsilon^2)$. The two algorithms are therefore equivalent again, although the LCU that uses logarithmic encoding has a slightly – $\log(L)$ – deeper circuit. Unless classical correlations are present or are not negligible, some of the applications that have been considered for the LCU circuit [PSF&al19; SOG&al02; Qis23] may be useful in specific contexts, but it cannot

	LCU	TE
(embarrassing) parallelization	no	yes
sampling complexity (i.i.d.)	$O(L^2/\epsilon^2)$	$O(L^2/\epsilon^2)$
sampling complexity (AE)	$O(L/\epsilon)$	$O(L^2/\epsilon)$

Table 8.1: A table representing the differences in terms of sampling complexity the LCU and TE approaches. In case of normalized sums, all sampling complexities have to be re-scaled by L^2. AE denotes the full amplitude estimation algorithm [Bra&al02], or one of its approximate versions – see also Section 8.5.2.

naively provide a speedup with respect to L in terms of sampling complexity (not even in terms of state preparation, as the variance of the TE sampler is quadratically smaller, which reduces the total number of shots needed from the L circuits).

8.3.4 Extension to general linear combinations

If the coefficients of the linear combination have both positive and negative values, we need to partially modify the encoding defined above and the derivations of the variance scaling. We first observe that we can separate the coefficients in positive and negative terms:

$$\tilde{C} = \sum_{i=1}^{L} a_i x_i = \sum_{i=1}^{L} (a_i^+ - a_i^-) x_i, \tag{8.48}$$

and

$$a_i^+ = \begin{cases} a_i, & \text{if } a_i \geq 0 \\ 0, & \text{otherwise,} \end{cases} \tag{8.49}$$

$$a_i^- = \begin{cases} -a_i, & \text{if } a_i < 0 \\ 0, & \text{otherwise.} \end{cases} \tag{8.50}$$

We define therefore $L_{+/-}$ as the number of coefficients with positive and negative signs respectively, s.t. $L = L_+ + L_-$. In order to be able to represent Eq. (8.11) using positive weights, we have to first decompose it in its positive and negative terms:

$$C = \sum_{i=1}^{L} a_i p_i = \sum_{i=1}^{L} a_i^+ p_i - \sum_{i=1}^{L} a_i^- p_i. \tag{8.51}$$

Afterwards, each term in the sum is redefined s.t.:

$$\sum_{i=1}^{L} a_i p_i = \left(\sum_{l=1}^{L} a_l^+\right) p_+ - \left(\sum_{k=1}^{L} a_k^-\right) p_-, \tag{8.52}$$

with $p_\pm = \sum_{i=1}^{L} w_i^\pm p_i$, where $w_i^\pm = a_i^\pm / \left(\sum_{l=1}^{L} a_l^\pm\right)$. Using the formulation of Eq. (8.3), we see that for each one of the positive or negative coefficients we have a correspondent unitary $V_i^{+/-}$ for the positive and negative coefficient respectively. To use the LCU circuit to sample both negative and positive linear combinations, we use the first control qubit line – see Fig. 8.1 (b). The unitaries $V_i^+, i = 1, ..., L^+$ are controlled on the value zero of the qubit and the unitaries $V_i^-, i = L^+, ..., L$ are controlled on value one, whereas the rest of the circuit remains unchanged. This leads to, e.g., the modified output of the circuit:

$$q_0 = \frac{1}{2}\left(\sum_{i=1}^{L^+} |w_i| m_i^+ + \sum_{i=L^+ +1}^{L} |w_i| \operatorname{Tr}\{\rho Z_{\text{prod}}\}\right), \tag{8.53}$$

$$q_1 = \frac{1}{2}\left(\sum_{i=L^+ +1}^{L} |w_i| m_i^- + \sum_{i=1}^{L^+} |w_i| \operatorname{Tr}\{\rho Z_{\text{prod}}\}\right). \tag{8.54}$$

The difference between q_0 and q_1 gives the final result up to a bias, i.e., the term $\langle Z_{\text{prod}}(0)\rangle = \operatorname{Tr}\{\rho Z_{\text{prod}}\}$. An interesting case is one where the number of positive and negative terms in the sum is the same, i.e., $L^+ = L^- = \frac{L}{2}$ and the absolute value of each coefficients is $|w_i| = \frac{1}{L}$. In this case, the bias is the same for both q_0 and q_1 and is removed when subtracting them. In the case of uniform coefficients, i.e., $|w_i| = \frac{1}{L}$, we can write the expression as

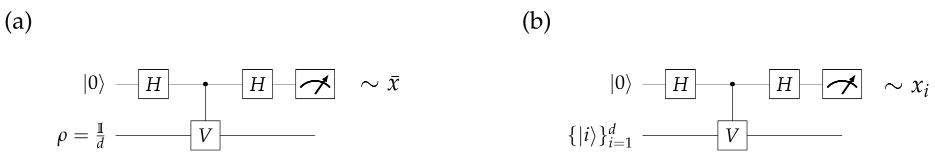
$$q_0 = \frac{1}{2}\left(\frac{1}{L}\sum_{i=1}^{L^+} m_i^+ + \frac{L - L^+}{L}\langle Z_{\text{prod}}(0)\rangle\right), \tag{8.55}$$

$$q_1 = \frac{1}{2}\left(\frac{1}{L}\sum_{i=L^+ +1}^{L} m_i^- + \frac{L^+}{L}\langle Z_{\text{prod}}(0)\rangle\right), \tag{8.56}$$

which leaves us with

$$z = 2(q_0 - q_1) + \left(\frac{2L^+}{L} - 1\right)\langle Z_{\text{prod}}(0)\rangle, \tag{8.57}$$

which, as expected, reduces to the case $z = 2(q_0 - q_1)$ for $L^+ = L^- = \frac{L}{2}$. This procedure allows us therefore to estimate linear combinations of expected values of, e.g., Pauli strings or projectors that contain both positive and negative coefficients. Now, let us analyze the variance in the case of this more general kind of linear combination of estimates. Each estimate q_0 and q_1 will have a Bernoulli-like variance – see Eq. (8.23). The bias will affect the variance with a quadratic factor $(2L^+/L - 1)^2 \leq 1$, which depends



$$\langle \bar{x} \rangle = \bar{p}$$
$$\sigma_{\bar{x}}^2 = \bar{p}(1-\bar{p})$$

$$\langle x_i \rangle = p_i$$
$$\sigma_{x_i}^2 = p_i(1-p_i)$$

Figure 8.3: In this figure we consider different types of sampling problems that use the Hadamard-like test circuit normally employed for DQC1 tasks. In (a) and (b) the same circuits are considered with a fixed unitary V. The values $\bar{x}$ and $x_i, i = 1, ..., d$ represent the Bernoulli counts sampled using the circuits (a) and (b), respectively.

on the problem considered.

8.4 Trace estimation

8.4.1 DQC1

When dealing with NMR quantum computers [Jon11], it is common to have access to n-qubit mixed states. So the question arose whether quantum computation could be performed using only mixed states and some control qubits, rather than using pure states that undergo unitary evolutions, which are significantly harder to produce using these systems. It turns out that there exist problems that can be solved exponentially faster on such a quantum computer compared to a classical computer [SJ08]. However, it turns out that this computation model is also significantly weaker than standard quantum computation [SJ08; She06]. This model is referred to nowadays as DQC1 [KL98] (Deterministic Quantum Computation with One Clean Qubit). Several problems involving computing distance measures between unitaries and states using quantum circuits are DQC1-complete or -hard [Pou&al04; KSC&al20; Bra&al23]. One for all, trace estimation of unitaries is DQC1-complete. The same is true for energy estimation for Hamiltonians with very low (logarithmic) connectivity [She06]. There is a deep connection between all problems involving linear combination of unitaries and DQC1 [Bra&al23]. In general, one can construct a DQC1-cost function by implementing a LCU-type circuit with controlled operations [PSF&al19].

8.4.2 Sampling Unitary traces and DQC1 basics

The complexity class DQC1 that allows direct computation with one clean qubit uses a single- or multi-qubit controlled unitary V acting on a maximally mixed state (which corresponds to uniformly sampled random pure states) to perform quantum computation

[KL98]. This model has been shown to be hard to simulate on classical computers if more than three output qubits are considered [MFF14]. Estimating the re-normalized real part of the trace of a unitary V is a complete problem for DQC1 [She06]. The circuit that allows this estimation is given in Fig. 8.3 (a) for mixed-states inputs and (b) for pure states. We describe these circuits in details below.
The probability of measuring the control qubit for the circuit that uses mixed states – see Fig. 8.3 (a) – in the zero (+) or one (-) state is given by

$$\bar{p}_{\pm} = \frac{1}{2}\left(1 \pm \frac{1}{d}\operatorname{Re}\{\operatorname{Tr}\{V\}\}\right). \tag{8.58}$$

The imaginary part can be obtained by inserting an S gate before the final measurement in the circuits given in Fig. (8.3). We can see that if we want to compute the trace of a unitary, the variance of the estimator for the trace behaves as a Bernoulli variable [AJL06]:

$$\operatorname{Var}(\bar{x}) = \bar{p}_{+}(1 - \bar{p}_{+}). \tag{8.59}$$

Here, we consider only one of the two outcomes and thus set $\bar{p} = \bar{p}_{+}$. Without using a maximally mixed state $\rho = \frac{\mathbb{I}}{d}$, computing the trace requires preparing $d = 2^n$ orthonormal basis states $\{|i\rangle\}_{i=1}^{d}$ and estimating their corresponding probability distribution p_i:

$$p_i = \frac{1}{2}(1 + \operatorname{Re}\{\langle i| V |i\rangle\}). \tag{8.60}$$

As the real part of the trace of V is given by $\sum_{i=1}^{d}\operatorname{Re}\{\langle i| V |i\rangle\}$, we can construct an estimator for the trace by creating an estimator $\bar{x}'$ that collects the counts of the d different circuits and sums them together.
The variance of this estimator behaves as a sum of independent Bernoulli variables, each one with variance $p_i(1 - p_i)$:

$$\operatorname{Var}(\bar{x}') = \sum_{i=1}^{d} p_i(1 - p_i), \tag{8.61}$$

and is d-times smaller than the variance of the estimator based on the maximally mixed state.

Proof. We prepare the d states $|i\rangle, i = 1, ..., d$ and apply the Hadamard test on each of them using the controlled unitary V. Since the states are prepared on d different Hilbert spaces, the variance of the estimates is simply the sum of the variances, each one equal to $p_i(1 - p_i)$ for $i = 1, ..., d$. Afterwards, we use the fact that $p_i \leq \sqrt{p_i}$ for $0 \leq p_i \leq 1$ and write:

$$\frac{1}{d}\sum_{i=1}^{d} p_i(1 - p_i) \leq \left(\frac{1}{d}\sum_{i=1}^{d}\sqrt{p_i}\sqrt{1 - p_i}\right)^2. \tag{8.62}$$

Then we apply one of the generalized mean inequalities [Bul03], which follows from the Jensen's inequality [Jen06]:

$$\left(\frac{1}{d}\sum_{i=1}^{d}\sqrt{p_i}\sqrt{1-p_i}\right)^2 \overset{\text{GM}}{\leq} \frac{1}{d^2}\sum_{i=1}^{d}p_i\sum_{j=1}^{d}(1-p_j). \tag{8.63}$$

The last term is the variance of the DQC1 estimator. Therefore, we have

$$\text{Var}(\bar{x}') \leq \frac{1}{d}\text{Var}(\bar{x}). \tag{8.64}$$

□

We see that the variance of the first estimator is at least d times smaller than the one of the DQC1 estimator. However, the DQC1 estimator does not require the preparation of d different states, as long as we have access to a source of randomly distributed pure states in input (or, equivalently, a maximally mixed state). In summary, the query complexity of both estimators with precision ϵ is $O(d^2/\epsilon^2)$ – in the former case, we need to sample the sum of d circuit outputs with precision $O(d/\epsilon^2)$, in the latter we sample from one circuit with precision $O(d^2/\epsilon^2)$.

8.5 Quantum sampling

8.5.1 Heisenberg scaling and phase estimation

For specific problems, phase estimation can be used to sample values at the Heisenberg limit [GLM04; GLM11]. If we consider a Hadamard test estimating the value p, with $p = \cos^2(\phi)$, we can implement the nth power of the corresponding operator to obtain:

$$p_n = \cos^2(n\phi). \tag{8.65}$$

The uncertainty in the estimation of p_n is given by a Bernoulli variance $p_n(1-p_n)$, which is $O(1)$. If we apply the propagation of the uncertainty to the value ϕ knowing the value of p_n with precision $\sqrt{p_n(1-p_n)}$, we can see that the value of ϕ has a variance that scales as $O(\frac{1}{n})$ [GLM04]. This is in contrast to classical estimation, where the standard error in standard Monte Carlo sampling scales with $O(1/\sqrt{n})$. The same phase value can be then used to estimate p with precision $O(1/n)$. Phase estimation has found application in different contexts outside of quantum metrology, for example in noise mitigation [PAO23] and Hamiltonian simulation [DLL&al24]. The amplitude estimation algorithm [Bra&al02] can be considered a generalization of this procedure to arbitrary sampled cost functions that do not exhibit the simple structure given in Eq. (8.65). An example is a situation in which the sampled quantity does not depend on a single phase value, but rather on a sum of different phase contributions. In this case, this algorithm can offer quadratic speedup over classical estimation routines.

8.5.2 Amplitude amplification and estimation

Amplitude amplification (AA) [Bra&al02] describes a class of algorithms based on Grover search [Gro96] that allows to perform faster sampling on quantum computers. If applied to estimation problems in the context of quantum circuits, it is often referred to as amplitude estimation (AE) and can be used to sample the information hidden inside of amplitudes of quantum states of the type:

$$|\psi\rangle = \sqrt{p}\,|0\rangle\,|\psi_1\rangle + \sqrt{1-p}\,|1\rangle\,|\psi_2\rangle\,, \tag{8.66}$$

where p is value to estimate. There exists a quantum algorithm that outputs to an estimator $\tilde{p}$ of p [Bra&al02], which is bounded from above as follows:

$$|p-\tilde{p}| \le 2\pi k \frac{\sqrt{p(1-p)}}{n_q} + \frac{\pi^2 k^2}{n_q^2}. \tag{8.67}$$

The AE routine succeeds with probability $p_{\text{succ}} = 8/\pi^2$ if $k = 1$ and $p_{\text{succ}} \ge 1 - \frac{1}{2(k-1)}$ if $k > 1$, where n_q is the number of oracle calls. Generally, the routine for AA is defined for pure states and not for mixed inputs, such as the DQC1 circuit given in Fig. 8.3 (a) and (b). However, the routine that generates the mixed state can be expressed as tracing out a pure state of higher dimensionality than the one considered as input to the sampling circuit. As a result, to apply AE to one of our problems, we have to consider the entire algorithm, the one that generates the mixed state and the one that generates the state in Eq. (8.69). As given in Ref. [KLP&al19], the n-qubit maximally mixed state $\rho = \frac{\mathbb{I}}{d}$ can be generated by creating a $2n$-qubit pure state

$$|\psi\rangle = \frac{1}{\sqrt{d}} \sum_{i=1}^{d} |i\rangle\,, \otimes |i\rangle \tag{8.68}$$

and tracing out one of the n-qubit subsystems. So the $(n+1)$-qubit DQC1 circuit can be written as a $(2n+1)$-qubit circuit by including the generation of the mixed state in its original routine.

8.5.2.1 Sampling with amplitude estimation

We consider here the TE sampler for the trace estimation problem. In this case we need to apply the AE scheme to each sub-problem. It is clear that this bound is quadratically better than classical Monte Carlo sampling. We assume that each quantum algorithm of the TE estimator can be written as [SUR&al20]:

$$\mathcal{V}_i\,|0\rangle = \sqrt{p_i}\,|0\rangle\,|\psi_{i,1}\rangle + \sqrt{1-p_i}\,|1\rangle\,|\psi_{i,2}\rangle\,, \tag{8.69}$$

for $i = 1, ..., L$. Thus, the Grover operator for AE is simply given by

$$Q_i = -\mathcal{V}_i S_0 \mathcal{V}_i^{-1} S_\chi, \tag{8.70}$$

where S_χ is the reflection operator that switches the sign of the state if the state is equal to the target (also known as the *good* state) [Bra&al02; SUR&al20] and the operator S_0 flips the state if it is the zero state. For each amplitude encoded in a quantum algorithm $\mathcal{V}_i$ we have:

$$|p_i - \tilde{p}_i| \le 2\pi k \frac{\sqrt{p_i(1-p_i)}}{n_q} + \frac{\pi^2 k^2}{n_q^2}. \tag{8.71}$$

Our aim is to achieve an overall precision ϵ of the full estimation, which gives us:

$$\epsilon = |\sum_{i=1}^{L} p_i - \sum_{i=1}^{L} \tilde{p}_i| \overset{\Delta}{\le} \sum_{i=1}^{L} |p_i - \tilde{p}_i| \le \\ \sum_{i=1}^{L} \frac{2\pi k}{n_q}\sqrt{p_i(1-p_i)} + \frac{k^2 L \pi^2}{n_q^2}, \tag{8.72}$$

where n_q is the number of oracle calls for each algorithm $i = 1, \dots, L$ and $k \in \mathbb{N}_{>0}$. The estimation is probabilistic, i.e., it is successful with probability p_{succ}. In the next steps we mention some aspects of the original derivation in Ref. [Bra&al02]. The probability of success of the AE algorithm can be bound from below using the trigamma function $\psi^{(1)}(k) = \sum_{m=k}^{\infty} 1/m^2$ [Bra&al02; AS64]. For any $k \in \mathbb{N}_{>0}$ we have:

$$p_{\text{succ}} \ge 1 - \frac{1}{2}\psi^{(1)}(k). \tag{8.73}$$

As the trigamma function fulfills the identity [Bra&al02]:

$$\psi^{(1)}(k) \le \frac{1}{k-1}, \tag{8.74}$$

the lower bound for the success probability can be written as

$$p_{\text{succ}} \ge 1 - \frac{1}{2(k-1)}, \tag{8.75}$$

where p_{succ} is the success probability of one quantum AE algorithm running on one of the estimation problems. For $k = 1$ we have $p_{\text{succ}} = \frac{8}{\pi^2}$ and considering that $p_i(1-p_i)$ is always smaller than $\frac{1}{4}$, we can write the following relation for the precision ϵ of the estimation:

$$\epsilon = \frac{\pi L}{n_q} + \frac{\pi^2 L}{n_q^2}, \tag{8.76}$$

and after solving the quadratic equation for small ϵ we obtain $\frac{\pi L}{n_q} \approx \epsilon$. The number of evaluations needed is thus $O(L^2/\epsilon)$ – L circuits with n_q oracle calls each, in analogy with the TE estimator – to have convergence with probability p_{succ}. If the sum given

in Eq. (8.72) is divided by L – which corresponds to the case where we sample a mean –, then the complexity reduces to $O(1/\epsilon)$. In the case of the LCU estimator we have instead $O(L/\epsilon)$ for the non-renormalized estimation and $O(1/\epsilon)$ for the renormalized one. In classical estimation, the renormalization factor causes the variance of the LCU estimator to grow quadratically with the number of estimates. In the case of AE, it only increases the error linearly. Despite this advantage, the implementation of the AE oracle is non-trivial.
We can briefly analyze a further important difference between the two algorithms. In the TE case, we run L different AE routines, each one with a success probability $p_{\text{succ}} \geq \frac{8}{\pi^2}$. In the LCU case, we apply the routine to one single circuit. Clearly, the second approach has an advantage in terms of overall probability of failure. In fact, in the former case there are L independent algorithmic runs that can return the wrong outcome. In the latter case, however, only one circuit with a corresponding AE routine and success probability p_{succ} is used.
If we consider NISQ circuits, which are not fault-tolerant, we quickly realize that the depth of the Grover's algorithm, which lies at the core of the AE routine, cannot be easily implemented on such circuits. However, alternative versions of the AE can be potentially applied on noisy quantum devices [SUR&al20; Gri&al21; PRF&al22]. AE can be used to estimate means of data sampled from arbitrary distributions [Shy21] and has a potentially vast range of industry-relevant applications, e.g., in finance. In this context, it can be used to sample quadratically faster from stochastic processes [SES&al20].

8.6 Discussion and Conclusion

In Chapter 2, 4 and, to a certain extent, 6 and 7 we consider several optimization problems involving (quantum) cost functions, i.e., cost functions that depend on variational parameters governing on the quantum dynamics. However, it is not always clear a priori how these cost functions can be estimated using quantum circuits. For example, measuring the average gate infidelity, which is quadratic in the trace, requires the tensor product of two unitaries [KLP&al19; MSG&al11]. Arbitrary cost functions can thus be implemented using both LCU and TE estimators, which exhibit the same sampling complexity. In the context of these two approaches, we provided the reader with an overview of different estimators used to determine the mean value of observables within acceptable precision. It is clear from our analysis – see also [Cha24] – that the complexity of estimating observables using LCU and TE samplers without any quantum AE routine is the same, i.e., $O(L^2/\epsilon^2)$. In the case of AE, the sampling complexity of the LCU estimator reduces to $O(L/\epsilon)$, which therefore proves to be quadratically faster than the TE estimator. Moreover, when using AE, the LCU approach benefits from the reduced execution time due to the success probability being considered for just one circuit and not for L circuits.

Conclusion and Outlook

8.1 Summary

Optimization and machine learning have become ubiquitous in quantum science and technology [KLF&al23]. In this work, we covered several optimization problems for quantum systems that are potentially relevant for present and future applications of quantum science. We start in Chapter 4 with functional learning of pulses for optimal quantum control problems. We show that it is possible to train adaptive solutions characterized by a functional dependence on system parameters similar to analytical control solutions. Such neural network-based solutions are significantly more expressive than standard optimal control pulses, even when trained using so-called robust cost functions. This enables us to perform gate-set learning using optimal control pulses: a family of such pulses can be trained to produce a continuous entangling gate as a function of its rotation angles, which enables the creation of universal gate sets with continuous parameters from an optimal control ansatz.
After showing that it is possible to generate variational gate sets using optimal control pulses, we move to a higher-level optimization problem. To compile arbitrary gates on quantum computers, a proper circuit compiler is needed. Such compilers exist and can be quite fast [YIL&al21], but they often do not provide optimal solutions. In this context, we deal not just with continuous, but also with discrete optimization of gate decompositions. In Chapter 5 we show how a reinforcement learning agent minimizes the length of compiled variational quantum circuits to produce relevant gates such as Toffoli or UCC unitary operators.
In Chapter 6 we introduce the optimization of purification protocols for specific families of states as a function of variational parameters. We show that the concurrence proves to be a powerful instrument to quantify the performance of such protocols for arbitrary input and output states and that the proposed variational protocol can surpass a basic CNOT protocol that does not use variational rotations.
In Chapter 7 we generalize this approach to arbitrary two-qubit quantum states. We compare the performances of static protocols based on the twirling operation and protocols that do not implement it against variational protocols with optimized angles. We conclude that the twirling operation seems to be affecting the overall performance of the original purification protocols, which are severely limited when dealing with random two-qubit input states. On the contrary, our optimized variational protocols do not employ twirling, but rather use a rank-two projector before applying layers of optimized non-linear maps. Our results show that thanks to such protocols it is possible to significantly increase the average concurrence output of purification protocols.

All the optimization problems considered above make use of sophisticated quantum cost functions that are often evaluated in experiments, such as the average gate fidelity or the concurrence. Therefore, we investigate how to construct estimators for such cost functions efficiently using measurement outcomes of different quantum circuits. In Chapter 8 we considered two different types of estimators. The former is the standard estimator (TE), which utilizes multiple quantum circuits, each one estimating the mean value of non-commuting observables. The latter uses the LCU approach (Linear Combination of Unitaries).
By analyzing the properties of both estimators, we observe that the LCU cannot lead to speedups with respect to the trivial estimators in the case of classical sampling. However, both estimators can be combined with amplitude amplification/estimation routines, which seems more beneficial for the LCU approach. Estimation routines based on AE are often complex and their corresponding circuits are of significant depth. However, recent implementations [SUR&al20; PRF&al22] have considered approaches to the AE algorithm that are more suitable for near-term quantum computers and can still retain part of the quantum speedup. The use of such near-term routines in variational and meta-variational optimization problems which use the LCU can be potentially beneficial for quantum experiments and deserves further investigation.

8.2 Outlook

Our work focused on optimizing the design of various quantum operations. As future goals of SOMA methods for continuous quantum gate sets we plan to implement such algorithms on experimental platforms. This implies training the meta-optimization control method in an open-loop setting and then testing it in a real-world scenario (open-loop) with the help of, e.g., interleaved randomized benchmarking [MGJ&al12] and black-box optimization algorithms [CCM11; WSG&al14]. Under certain conditions, the experimental implementation can also benefit from gradient estimation techniques [WIW&al22; KK23]. Exploring the possibility of using filters [SZC&al23] could drastically increase the precision of the optimal quantum control solution. Such optimal control methods could also benefit from reinforcement learning applications and adaptive estimators such as contextual bandits [BAL21], which could potentially address parameter drifts. Another possible research direction is represented by reinforcement learning algorithms specifically developed for quantum circuits and quantum control applications [PPM23]. The optimization routine developed for purification protocols can, in principle, be extended to other entanglement distillation procedures, for example in the case of so-called pumping protocols [DB07] and in the case of qudits [SB22]. The potential of the LCU approach is currently still under investigation. We aim at combining it with near-term amplification routines to estimate different types of cost functions and to study its benefit in mitigating classical experimental noise [OPR&al21].

Bibliography

[AAA&al24] Rajeev Acharya, Laleh Aghababaie-Beni, Igor Aleiner, et al. *Quantum error correction below the surface code threshold.* 2024. URL: https://arxiv.org/abs/2408.13687.

[ABB&al21] David Awschalom, Karl K. Berggren, Hannes Bernien, et al. "Development of Quantum Interconnects (QuICs) for Next-Generation Information Technologies". In: *PRX Quantum* 2 (1 Feb. 2021), p. 017002.

[ACd89] B. Apolloni, C. Carvalho, and D. de Falco. "Quantum stochastic optimization". In: *Stochastic Processes and their Applications* 33.2 (1989), pp. 233–244.

[Aga19] Abien Fred Agarap. *Deep Learning using Rectified Linear Units (ReLU).* 2019. URL: https://arxiv.org/abs/1803.08375.

[Aha03] Dorit Aharonov. *A Simple Proof that Toffoli and Hadamard are Quantum Universal.* 2003. URL: https://arxiv.org/abs/quant-ph/0301040.

[Aiz63] Kêitsiro Aizu. "Parameter Differentiation of Quantum-Mechanical Linear Operators". In: *Journal of Mathematical Physics* 4.6 (June 1963), pp. 762–775.

[AJL06] Dorit Aharonov, Vaughan Jones, and Zeph Landau. *A Polynomial Quantum Algorithm for Approximating the Jones Polynomial.* 2006. URL: https://arxiv.org/abs/quant-ph/0511096.

[AKM84] Suguru Arimoto, Sadao Kawamura, and Fumio Miyazaki. "Bettering operation of Robots by learning". In: *Journal of Robotic Systems* 1.2 (1984), pp. 123–140.

[AL18] Tameem Albash and Daniel A. Lidar. "Adiabatic quantum computation". In: *Reviews of Modern Physics* 90.1 (Jan. 2018).

[Alz&al21] Laith Alzubaidi et al. "Review of deep learning: concepts, CNN architectures, challenges, applications, future directions". In: *Journal of Big Data* 8.1 (Mar. 2021), p. 53.

[AM86] C. Atkeson and J. McIntyre. "Robot trajectory learning through practice". In: *Proceedings. 1986 IEEE International Conference on Robotics and Automation.* Vol. 3. San Francisco, California, 1986, pp. 1737–1742.

[Ant&al03] Igor Antonov et al. "Activity-Dependent Presynaptic Facilitation and Hebbian LTP Are Both Required and Interact during Classical Conditioning in Aplysia". In: *Neuron* 37.1 (2003), pp. 135–147.

[Ara&al21] Israel F. Araujo et al. “A divide-and-conquer algorithm for quantum state preparation”. In: *Scientific Reports* 11.1 (Mar. 2021), p. 6329.

[Aru&al19] Frank Arute et al. “Quantum supremacy using a programmable superconducting processor”. In: *Nature* 574.7779 (Oct. 2019), pp. 505–510.

[AS64] Milton Abramowitz and Irene A. Stegun. *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables*. 1964.

[BAA22] Lucas Böttcher, Nino Antulov-Fantulin, and Thomas Asikis. “AI Pontryagin or how artificial neural networks learn to control dynamical systems”. In: *Nature Comm.* 13.1 (Jan. 2022), p. 333.

[BAH&al21] Yuval Baum, Mirko Amico, Sean Howell, et al. “Experimental Deep Reinforcement Learning for Error-Robust Gate-Set Design on a Superconducting Quantum Computer”. In: *PRX Quantum* 2 (4 Nov. 2021), p. 040324.

[Bak02] Andrew Baker. *Matrix Groups*. Springer London, 2002.

[BAL21] Alberto Bietti, Alekh Agarwal, and John Langford. *A Contextual Bandit Bake-off*. 2021. URL: `https://arxiv.org/abs/1802.04064`.

[Bar21] Nikhil Barhate. *Minimal PyTorch Implementation of Proximal Policy Optimization*. 2021. URL: `https://github.com/nikhilbarhate99/PPO-PyTorch`.

[BB14] Charles H. Bennett and Gilles Brassard. “Quantum cryptography: Public key distribution and coin tossing”. In: *Theoretical Computer Science* 560 (2014). Theoretical Aspects of Quantum Cryptography – celebrating 30 years of BB84, pp. 7–11.

[BBB&al21] Andreas Bayerstadler, Guillaume Becquin, Julia Binder, et al. “Industry quantum computing applications”. In: *EPJ Quantum Technology* 8.1 (Nov. 2021).

[BBC&al93] Charles H. Bennett, Gilles Brassard, Claude Crépeau, et al. “Teleporting an unknown quantum state via dual classical and Einstein-Podolsky-Rosen channels”. In: *Phys. Rev. Lett.* 70 (13 Mar. 1993), pp. 1895–1899.

[BBC&al95] Adriano Barenco, Charles H. Bennett, Richard Cleve, et al. “Elementary gates for quantum computation”. In: *Phys. Rev. A* 52.5 (Nov. 1995), pp. 3457–3467.

[BBN19] Ryan Babbush, Dominic W. Berry, and Hartmut Neven. “Quantum simulation of the Sachdev-Ye-Kitaev model by asymmetric qubitization”. In: *Phys. Rev. A* 99 (4 Apr. 2019), p. 040301.

[BBP&al96] Charles H. Bennett, Gilles Brassard, Sandu Popescu, et al. “Purification of Noisy Entanglement and Faithful Teleportation via Noisy Channels”. In: *Phys. Rev. Lett.* 76.5 (Jan. 1996), pp. 722–725.

[BCC06] Stefano Bertini, Sergio L. Cacciatori, and Bianca L. Cerchiai. “On the Euler angles for SU(N)”. In: *Journal of Mathematical Physics* 47.4 (Apr. 2006), p. 043510.

[BCM&al19] Colin D. Bruzewicz, John Chiaverini, Robert McConnell, et al. “Trapped-ion quantum computing: Progress and challenges”. In: *Applied Physics Reviews* 6.2 (June 2019), p. 021314.

[BCO&al09] S. Blanes, F. Casas, J.A. Oteo, et al. “The Magnus expansion and some of its applications”. In: *Phys. Rep.* 470.5–6 (Jan. 2009), pp. 151–238.

[BCP&al16] Greg Brockman, Vicki Cheung, Ludwig Pettersson, et al. *Openai gym.* 2016. URL: `https://arxiv.org/abs/1606.01540`.

[BCR10] Constantin Brif, Raj Chakrabarti, and Herschel Rabitz. “Control of quantum phenomena: past, present and future”. In: *New Journal of Physics* 12.7 (July 2010), p. 075008.

[BD12] Hans J. Briegel and Gemma De las Cuevas. “Projective simulation for artificial intelligence”. In: *Scientific Reports* 2.1 (May 2012), p. 400.

[BDL24] Daniel Basilewitsch, Clemens Dlaska, and Wolfgang Lechner. “Comparing planar quantum computing platforms at the quantum speed limit”. In: *Phys. Rev. Res.* 6 (2 Apr. 2024), p. 023026.

[BDS&al18] Marin Bukov, Alexandre G.R. Day, Dries Sels, et al. “Reinforcement Learning in Different Phases of Quantum Control”. In: *Phys. Rev.X* 8.3 (Sept. 2018), p. 031086.

[BDS&al96] Charles H. Bennett, David P. DiVincenzo, John A. Smolin, et al. “Mixed-state entanglement and quantum error correction”. In: *Phys. Rev. A* 54 (5 Nov. 1996), pp. 3824–3851.

[Bel22] Gabriel Belouze. *Optimization without Backpropagation.* 2022. URL: `https://arxiv.org/abs/2209.06302`.

[Bel52] Richard Bellman. “On the Theory of Dynamic Programming”. In: *Proceedings of the National Academy of Sciences* 38.8 (Aug. 1952), pp. 716–719.

[Ben80] Paul Benioff. “The computer as a physical system: A microscopic quantum mechanical Hamiltonian model of computers as represented by Turing machines”. In: *Journal of Statistical Physics* 22 (May 1980), pp. 563–591.

[Ber17] József Zsolt Bernád. “Hybrid quantum repeater based on resonant qubit-field interactions”. In: *Phys. Rev. A* 96 (5 Nov. 2017), p. 052329.

[Ber87] Michael Victor Berry. “Quantum phase corrections from adiabatic iteration”. In: *Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences* 414.1846 (1987), pp. 31–46.

[BFH&al18] James Bradbury, Roy Frostig, Peter Hawkins, et al. *JAX: composable transformations of Python+NumPy programs.* Version 0.2.5. 2018.

[BGG&al21] Alexandre Blais, Arne L. Grimsmo, S. M. Girvin, et al. “Circuit quantum electrodynamics”. In: *Rev. Mod. Phys.* 93 (2 May 2021), p. 025005.

[BH18] Lukas Balles and Philipp Hennig. “Dissecting Adam: The Sign, Magnitude and Variance of Stochastic Gradients”. In: *Proceedings of the 35th International Conference on Machine Learning.* Ed. by Jennifer Dy and Andreas Krause. Vol. 80. Proceedings of Machine Learning Research. PMLR, July 2018, pp. 404–413.

[BH96] V. Bužek and M. Hillery. “Quantum copying: Beyond the no-cloning theorem”. In: *Phys. Rev. A* 54 (3 Sept. 1996), pp. 1844–1852.

[BHC10] Troy W. Borneman, Martin D. Hürlimann, and David G. Cory. “Application of optimal control to CPMG refocusing pulse design”. In: *Journal of Magnetic Resonance* 207.2 (2010), pp. 220–233.

[BHL&al16] C. J. Ballance, T. P. Harty, N. M. Linke, et al. “High-Fidelity Quantum Logic Gates Using Trapped-Ion Hyperfine Qubits”. In: *Phys. Rev. Lett.* 117 (6 Aug. 2016), p. 060504.

[Bia&al17] Jacob Biamonte et al. “Quantum machine learning”. In: *Nature* 549.7671 (Sept. 2017), pp. 195–202.

[Bie&al09] Michael J. Biercuk et al. “Optimized dynamical decoupling in a model quantum memory”. In: *Nature* 458.7241 (Apr. 2009), pp. 996–1000.

[Bis07] Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics).* 1st ed. Springer, 2007.

[BK03] Mark S. Byrd and Navin Khaneja. “Characterization of the positivity of the density matrix in terms of the coherence vector representation”. In: *Phys. Rev. A* 68 (6 Dec. 2003), p. 062322.

[BK08] Reinhold A Bertlmann and Philipp Krammer. “Bloch vectors for qudits”. In: *J. Phys. A: Math. Theor.* 41.23 (May 2008), p. 235303.

[BKP13] Peter Brooks, Alexei Kitaev, and John Preskill. “Protected gates for superconducting qubits”. In: *Phys. Rev. A* 87 (5 May 2013), p. 052306.

[BLN&al95] Richard H Byrd, Peihuang Lu, Jorge Nocedal, et al. “A limited memory algorithm for bound constrained optimization”. In: *SIAM Journal on scientific computing* 16.5 (1995), pp. 1190–1208.

[BM07] Rodney J. Bartlett and Monika Musiał. “Coupled-cluster theory in quantum chemistry”. In: *Rev. Mod. Phys.* 79 (1 Feb. 2007), pp. 291–352.

[BNS94] Richard H. Byrd, Jorge Nocedal, and Robert B. Schnabel. “Representations of quasi-Newton matrices and their use in limited memory methods”. In: *Math. Program.* 63.1 (Jan. 1994), pp. 129–156.

[Bou&al12] J. Bourassa et al. "Josephson-junction-embedded transmission-line resonators: From Kerr medium to in-line transmon". In: *Phys. Rev. A* 86 (1 July 2012), p. 013814.

[Boy&al20] W. L. Boyajian et al. "On the convergence of projective-simulation–based reinforcement learning in Markov decision processes". In: *Quantum Machine Intelligence* 2.2 (Nov. 2020), p. 13.

[BPR&al18] Atilim Gunes Baydin, Barak A. Pearlmutter, Alexey Andreyevich Radul, et al. "Automatic Differentiation in Machine Learning: a Survey". In: *Journal of Machine Learning Research* 18.153 (2018), pp. 1–43.

[BPS&al22] Atılım Güneş Baydin, Barak A. Pearlmutter, Don Syme, et al. *Gradients without Backpropagation.* 2022. URL: https://arxiv.org/abs/2202.08587.

[BQA20] Leonardo Banchi, Nicolás Quesada, and Juan Miguel Arrazola. "Training Gaussian boson sampling distributions". In: *Phys. Rev. A* 102 (1 July 2020), p. 012417.

[Bra&al02] Gilles Brassard et al. "Quantum amplitude amplification and estimation". In: *Contemporary Mathematics* 305 (2002), pp. 53–74.

[Bra&al23] Carlos Bravo-Prieto et al. "Variational Quantum Linear Solver". In: *Quantum* 7 (Nov. 2023), p. 1188.

[Bri&al09] H. J. Briegel et al. "Measurement-based quantum computation". In: *Nature Physics* 5.1 (Jan. 2009), pp. 19–26.

[BRO70] C. G. BROYDEN. "The Convergence of a Class of Double-rank Minimization Algorithms 1. General Considerations". In: *IMA Journal of Applied Mathematics* 6.1 (Mar. 1970), pp. 76–90.

[BRS15] Alex Bocharov, Martin Roetteler, and Krysta M. Svore. "Efficient Synthesis of Universal Repeat-Until-Success Quantum Circuits". In: *Phys. Rev. Lett.* 114 (8 Feb. 2015), p. 080502.

[BSK&al21] Sangkha Borah, Bijita Sarma, Michael Kewming, et al. "Measurement-Based Feedback Quantum Control with Deep Reinforcement Learning for a Double-Well Nonlinear Potential". In: *Phys. Rev. Lett.* 127 (19 Nov. 2021), p. 190403.

[BSL&al16] R. Barends, A. Shabani, L. Lamata, et al. "Digitized adiabatic quantum computing with a superconducting circuit". In: *Nature* 534.7606 (June 2016), pp. 222–226.

[BTK&al16] József Zsolt Bernád, Juan Mauricio Torres, Ludwig Kunz, et al. "Multiphoton-state-assisted entanglement purification of material qubits". In: *Phys. Rev. A* 93 (3 Mar. 2016), p. 032317.

[Bul03] P. S. Bullen. *Handbook of Means and Their Inequalities.* Springer Netherlands, 2003.

[BV97] Ethan Bernstein and Umesh Vazirani. “Quantum Complexity Theory”. In: *SIAM Journal on Computing* 26.5 (Oct. 1997), pp. 1411–1473.

[BWK22] Lennart Bittel, Jens Watty, and Martin Kliesch. *Fast gradient estimation for variational quantum algorithms*. 2022. URL: `https://doi.org/10.48550/arXiv.2210.06484`.

[BWP24] Joseph Bowles, David Wierichs, and Chae-Yeun Park. *Backpropagation scaling in parameterised quantum circuits*. 2024. URL: `https://arxiv.org/abs/2306.14962`.

[CCM11] Tommaso Caneva, Tommaso Calarco, and Simone Montangero. “Chopped random-basis quantum optimization”. In: *Phys. Rev. A* 84 (2 Aug. 2011), p. 022326.

[CDG98] Claude Cohen-Tannoudji, Jacques Dupont-Roc, and Gilbert Grynberg. *Atom-Photon Interactions*. Nashville, TN: John Wiley & Sons, Apr. 1998.

[Cer&al21a] M. Cerezo et al. “Cost function dependent barren plateaus in shallow parametrized quantum circuits”. In: *Nature Communications* 12.1 (Mar. 2021), p. 1791.

[Cer&al21b] M. Cerezo et al. “Variational quantum algorithms”. In: *Nature Reviews Physics* 3.9 (Sept. 2021), pp. 625–644.

[CGT&al09] J. M. Chow, J. M. Gambetta, L. Tornberg, et al. “Randomized Benchmarking and Process Tomography for Gate Errors in a Solid-State Qubit”. In: *Phys. Rev. Lett.* 102 (9 Mar. 2009), p. 090502.

[Cha24] Shantanav Chakraborty. “Implementing any Linear Combination of Unitaries on Intermediate-term Quantum Computers”. In: *Quantum* 8 (Oct. 2024), p. 1496.

[Cho&al21] Alexandre Choquette et al. “Quantum-optimal-control-inspired ansatz for variational quantum algorithms”. In: *Phys. Rev. Res.* 3 (2 May 2021), p. 023092.

[Cho75] Man-Duen Choi. “Completely positive linear maps on complex matrices”. In: *Linear Algebra and its Applications* 10.3 (1975), pp. 285–290.

[CJ00] H K Cummins and J A Jones. “Use of composite rotations to correct systematic errors in NMR quantum computation”. In: *New Journal of Physics* 2.1 (Mar. 2000), p. 006.

[CK97] P. Crescenzi and V. Kann. “Approximation on the web: A compendium of NP optimization problems”. In: *Randomization and Approximation Techniques in Computer Science*. Ed. by José Rolim. Berlin, Heidelberg: Springer Berlin Heidelberg, 1997, pp. 111–118.

[CKA21] Alba Cervera-Lierta, Jakob S. Kottmann, and Alán Aspuru-Guzik. “Meta-Variational Quantum Eigensolver: Learning Energy Profiles of Parameterized Hamiltonians for Quantum Simulation”. In: *PRX Quantum* 2 (2 May 2021), p. 020329.

[CKS17] Andrew M. Childs, Robin Kothari, and Rolando D. Somma. “Quantum Algorithm for Systems of Linear Equations with Exponentially Improved Dependence on Precision”. In: *SIAM Journal on Computing* 46.6 (2017), pp. 1920–1950.

[CLG&al24] M. Cerezo, Martin Larocca, Diego García-Martín, et al. *Does provable absence of barren plateaus imply classical simulability? Or, why we need to rethink variational quantum computing.* 2024. URL: https://arxiv.org/abs/2312.09121.

[CLM&al14] Eric Chitambar, Debbie Leung, Laura Mančinska, et al. “Everything You Always Wanted to Know About LOCC (But Were Afraid to Ask)”. In: *Communications in Mathematical Physics* 328.1 (Mar. 2014), pp. 303–326.

[CN97] Isaac L. Chuang and M. A. Nielsen. “Prescription for experimental determination of the dynamics of a quantum black box”. In: *Journal of Modern Optics* 44.11-12 (1997), pp. 2455–2467.

[Cra&al10] Marcus Cramer et al. “Efficient quantum state tomography”. In: *Nature Communications* 1.1 (Dec. 2010), p. 149.

[CRB&al18] Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, et al. “Neural ordinary differential equations”. In: *Advances in neural information processing systems* 31 (2018).

[CRC&al23] F. A. Cárdenas-López, J. C. Retamal, Xi Chen, et al. *Resilient superconducting-element design with genetic algorithms.* 2023. URL: https://arxiv.org/abs/2302.01837.

[CRS&al21] Lukasz Cincio, Kenneth Rudinger, Mohan Sarovar, et al. “Machine Learning of Noise-Resilient Quantum Circuits”. In: *PRX Quantum* 2 (1 Feb. 2021), p. 010324.

[CS17] Anirban Narayan Chowdhury and Rolando D. Somma. “Quantum algorithms for Gibbs sampling and hitting-time estimation”. In: *Quantum Info. Comput.* 17.1–2 (Feb. 2017), pp. 41–64.

[CT17] Giuseppe Carleo and Matthias Troyer. “Solving the quantum many-body problem with artificial neural networks”. In: *Science* 355.6325 (2017), pp. 602–606.

[CW12] Andrew M. Childs and Nathan Wiebe. “Hamiltonian Simulation Using Linear Combinations of Unitary Operations”. In: *Quantum Information and Computation* 12 (Nov. 2012), pp. 901–924.

[CW19] Shuyang Chen and John T. Wen. “Neural-Learning Trajectory Tracking Control of Flexible-Joint Robot Manipulators with Unknown Dynamics”. In: *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2019, pp. 128–135.

[CZ00] Juan Ignacio Cirac and Peter Zoller. “A scalable quantum computer with ions in an array of microtraps”. In: *Nature* 404.6778 (2000), pp. 579–581.

[CZ95] J. I. Cirac and P. Zoller. “Quantum Computations with Cold Trapped Ions”. In: *Phys. Rev. Lett.* 74 (20 May 1995), pp. 4091–4094.

[Dal&al20] Mogens Dalgaard et al. “Global optimization of quantum dynamics with AlphaZero deep exploration”. In: *npj Quantum Information* 6.1 (Jan. 2020), p. 6.

[Dao&al22] Mohammad Sh. Daoud et al. “Gradient-Based Optimizer (GBO): A Review, Theory, Variants, and Applications”. In: *Archives of Computational Methods in Engineering* (Dec. 2022).

[DB07] W Dür and H J Briegel. “Entanglement purification and quantum error correction”. In: *Reports on Progress in Physics* 70.8 (July 2007), pp. 1381–1424.

[Deb&al16] S. Debnath et al. “Demonstration of a small programmable quantum computer with atomic qubits”. In: *Nature* 536.7614 (Aug. 2016), pp. 63–66.

[DeC&al23] Matthew DeCross et al. “Qubit-Reuse Compilation with Mid-Circuit Measurement and Reset”. In: *Phys. Rev. X* 13 (4 Dec. 2023), p. 041057.

[DEJ&al96] David Deutsch, Artur Ekert, Richard Jozsa, et al. “Quantum Privacy Amplification and the Security of Quantum Cryptography over Noisy Channels”. In: *Phys. Rev. Lett.* 77 (13 Sept. 1996), pp. 2818–2821.

[DiV00] David P. DiVincenzo. “The Physical Implementation of Quantum Computation”. In: *Fortschritte der Physik* 48.9-11 (2000), pp. 771–783.

[DiV95] David P. DiVincenzo. “Two-bit gates are universal for quantum computation”. In: *Phys. Rev. A* 51 (2 Feb. 1995), pp. 1015–1022.

[DJ92] David Deutsch and Richard Jozsa. “Rapid solution of problems by quantum computation”. In: *Proceedings of the Royal Society of London. Series A: Mathematical and Physical Sciences* 439.1907 (1992), pp. 553–558.

[DKM&al08] Michaël Deschamps, Gwendal Kervern, Dominique Massiot, et al. “Superadiabaticity in magnetic resonance”. In: *The Journal of Chemical Physics* 129.20 (Nov. 2008), p. 204110.

[DLL&al24] Zhiyan Ding, Haoya Li, Lin Lin, et al. “Quantum Multiple Eigenvalue Gaussian filtered Search: an efficient and versatile quantum phase estimation method”. In: *Quantum* 8 (Oct. 2024), p. 1487.

[DM22] Mogens Dalgaard and Felix Motzoi. “Fast, high precision dynamics in quantum optimal control theory”. In: *J. Phys. B: At. Mol. Opt. Phys.* 55 (Apr. 2022), p. 085501.

[DMJ&al20] Mogens Dalgaard, Felix Motzoi, Jesper Hasseriis Mohr Jensen, et al. “Hessian-based optimization of constrained quantum control”. In: *Phys. Rev. A* 102 (4 Oct. 2020), p. 042612.

[DMN13] Simon J Devitt, William J Munro, and Kae Nemoto. “Quantum error correction for beginners”. In: *Reports on Progress in Physics* 76.7 (June 2013), p. 076001.

[DMS22] Mogens Dalgaard, Felix Motzoi, and Jacob Sherson. “Predicting quantum dynamical cost landscapes with deep learning”. In: *Phys. Rev. A* 105 (1 Jan. 2022), p. 012402.

[DN06] Christopher M. Dawson and Michael A. Nielsen. “The Solovay-Kitaev Algorithm”. In: *Quantum Info. Comput.* 6.1 (Jan. 2006), pp. 81–95.

[DPS&al24] Alessandro David, Akshay Menon Pazhedath, Lars R. Schreiber, et al. *Long distance spin shuttling enabled by few-parameter velocity optimization*. 2024. URL: `https://arxiv.org/abs/2409.07600`.

[dSG&al11] P. de Fouquieres, S.G. Schirmer, S.J. Glaser, et al. “Second order gradient ascent pulse engineering”. In: *Journal of Magnetic Resonance* 212.2 (2011), pp. 412–417.

[DST&al20] Marc G. Davis, Ethan Smith, Ana Tudor, et al. “Towards Optimal Topology Aware Quantum Circuit Synthesis”. In: *2020 IEEE International Conference on Quantum Computing and Engineering (QCE)*. 2020, pp. 223–234.

[DVB19] D. Devikanniga, K. Vetrivel, and N. Badrinath. “Review of Meta-Heuristic Optimization based Artificial Neural Networks and its Applications”. In: *Journal of Physics: Conference Series* 1362.1 (Nov. 2019), p. 012074.

[DVS01] A. D’Souza, S. Vijayakumar, and S. Schaal. “Learning inverse kinematics”. In: *Proceedings 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems. Expanding the Societal Role of Robotics in the the Next Millennium (Cat. No.01CH37180)*. Vol. 1. Maui, Hawaii, 2001, 298–303 vol.1.

[DW16] Pierre-Luc Dallaire-Demers and Frank K. Wilhelm. “Quantum gates and architecture for the quantum simulation of the Fermi-Hubbard model”. In: *Phys. Rev. A* 94 (6 Dec. 2016), p. 062304.

[DWM22] Mogens Dalgaard, Carrie A. Weidner, and Felix Motzoi. “Dynamical Uncertainty Propagation with Noisy Quantum Parameters”. In: *Phys. Rev. Lett.* 128 (15 Apr. 2022), p. 150503.

[EB01] Jens Eisert and Hans J. Briegel. “Schmidt measure as a tool for quantifying multiparticle entanglement”. In: *Phys. Rev. A* 64 (2 July 2001), p. 022306.

[Eva&al22] Benjamin Eva et al. “How a Minimal Learning Agent can Infer the Existence of Unobserved Variables in a Complex Environment”. In: *Minds and Machines* (Dec. 2022).

[EWP&al19] Alexander Erhard, Joel J. Wallman, Lukas Postler, et al. “Characterizing large-scale quantum computers via cycle benchmarking”. In: *Nature Communications* 10.1 (Nov. 2019).

[FBS&al17] Alex Fout, Jonathon Byrd, Basir Shariat, et al. “Protein Interface Prediction using Graph Convolutional Networks”. In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon et al. Vol. 30. Curran Associates, Inc., 2017.

[Fey82] Richard P. Feynman. “Simulating physics with computers”. In: *International Journal of Theoretical Physics* 21.6 (June 1982), pp. 467–488.

[FGG14] Edward Farhi, Jeffrey Goldstone, and Sam Gutmann. *A quantum approximate optimization algorithm*. 2014. URL: `https://arxiv.org/abs/1411.4028`.

[FGP&al21] Robert Falck, Justin S. Gray, Kaushik Ponnapalli, et al. “dymos: A Python package for optimal control of multidisciplinary systems”. In: *Journal of Open Source Software* 6.59 (2021), p. 2809.

[FHI&al18] Vincent François-Lavet, Peter Henderson, Riashat Islam, et al. “An Introduction to Deep Reinforcement Learning”. In: *Foundations and Trends® in Machine Learning* 11.3-4 (2018), pp. 219–354.

[FJ16] Jianjia Fei and Robert Joynt. “Numerical Computations of Separability Probabilities”. In: *Rep. Math. Phys.* 78.2 (2016), pp. 177–182.

[FNM&al21] Thomas Fösel, Murphy Yuezhen Niu, Florian Marquardt, et al. *Quantum circuit optimization with deep reinforcement learning*. 2021. URL: `https://arxiv.org/abs/2103.07585`.

[Fox&al20] B. Foxen et al. “Demonstrating a Continuous Set of Two-Qubit Gates for Near-Term Quantum Algorithms”. In: *Phys. Rev. Lett.* 125 (12 Sept. 2020), p. 120504.

[Fra&al17] Florian Frank et al. “Autonomous calibration of single spin qubit operations”. In: *npj Quantum Information* 3.1 (Dec. 2017), p. 48.

[Fre98] Ray Freeman. *Spin choreography*. Oxford University Press Oxford, 1998.

[FRS05] Heng Fan, Vwani Roychowdhury, and Thomas Szkopek. “Optimal two-qubit quantum circuits using exchange interactions”. In: *Phys. Rev. A* 72 (5 Nov. 2005), p. 052323.

[FT82] Edward Fredkin and Tommaso Toffoli. “Conservative logic”. In: *International Journal of Theoretical Physics* 21.3–4 (Apr. 1982), pp. 219–253.

[Gam16] Omar Gamel. “Entangled Bloch spheres: Bloch matrix and two-qubit state space”. In: *Phys. Rev. A* 93 (6 June 2016), p. 062320.

[GAN14] I. M. Georgescu, S. Ashhab, and Franco Nori. “Quantum simulation”. In: *Rev. Mod. Phys.* 86 (1 Mar. 2014), pp. 153–185.

[GBC&al15] Steffen J. Glaser, Ugo Boscain, Tommaso Calarco, et al. “Training Schrödinger’s cat: quantum optimal control”. In: *Eur. Phys. J. D* 69.12 (Dec. 2015), p. 279.

[GBC16] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning.* `http://www.deeplearningbook.org`. Cambridge, Massachusetts: MIT Press, 2016.

[GBG&al19] Michael H. Goerz, Daniel Basilewitsch, Fernando Gago-Encinas, et al. “Krotov: A Python implementation of Krotov’s method for quantum optimal control”. In: *SciPost Phys.* 7 (6 2019), p. 80.

[GEK19] Sergey An. Gayvoronskiy, Tatiana Ezangina, and Ivan Khozhaev. “Placing Poles Allocation Areas of Interval Control System with Desired Root Quality Indices”. In: *2019 International Russian Automation Conference (RusAutoCon)*. 2019, pp. 1–6.

[GG24] Guillaume Garrigos and Robert M. Gower. *Handbook of Convergence Theorems for (Stochastic) Gradient Methods.* 2024. URL: `https://arxiv.org/abs/2301.11235`.

[GGD&al21] Élie Genois, Jonathan A. Gross, Agustin Di Paolo, et al. “Quantum-Tailored Machine-Learning Characterization of a Superconducting Qubit”. In: *PRX Quantum* 2 (4 Dec. 2021), p. 040355.

[GKJ16] András Gilyén, Tamás Kiss, and Igor Jex. “Exponential Sensitivity and its Cost in Quantum Physics”. In: *Sci. Rep.* 6.1 (Feb. 2016), p. 20076.

[GLM04] Vittorio Giovannetti, Seth Lloyd, and Lorenzo Maccone. “Quantum-Enhanced Measurements: Beating the Standard Quantum Limit”. In: *Science* 306.5700 (2004), pp. 1330–1336.

[GLM11] Vittorio Giovannetti, Seth Lloyd, and Lorenzo Maccone. “Advances in quantum metrology”. In: *Nature Photonics* 5.4 (Mar. 2011), pp. 222–229.

[GM22] Irene López Gutiérrez and Christian B. Mendl. “Real time evolution with neural-network quantum states”. In: *Quantum* 6 (Jan. 2022), p. 627.

[GMT&al12] J. P. Gaebler, A. M. Meier, T. R. Tan, et al. “Randomized Benchmarking of Multiqubit Gates”. In: *Phys. Rev. Lett.* 108 (26 June 2012), p. 260503.

[GNX&al13] Yilun Guan, Duy Quang Nguyen, Jingwei Xu, et al. “Reexamination of measurement-induced chaos in entanglement-purification protocols”. In: *Phys. Rev. A* 87 (5 May 2013), p. 052316.

[Goe15] Michael Hartmut Goerz. “Optimizing Robust Quantum Gates in Open Quantum Systems”. PhD thesis. Kassel, Universität Kassel, Fachbereich Mathematik und Naturwissenschaften, May 2015.

[Got98] Daniel Gottesman. “Theory of fault-tolerant quantum computation”. In: *Phys. Rev. A* 57 (1 Jan. 1998), pp. 127–137.

[Gre86] John J. Grefenstette. “Optimization of Control Parameters for Genetic Algorithms”. In: *IEEE Transactions on Systems, Man, and Cybernetics* 16.1 (1986), pp. 122–128.

[Gri&al21] Dmitry Grinko et al. “Iterative quantum amplitude estimation”. In: *npj Quantum Information* 7.1 (Mar. 2021), p. 52.

[Gro96] Lov K. Grover. *A fast quantum mechanical algorithm for database search*. 1996. URL: https://arxiv.org/abs/quant-ph/9605043.

[GST24] Manuel Guatto, Gian Antonio Susto, and Francesco Ticozzi. “Improving robustness of quantum feedback control with reinforcement learning”. In: *Phys. Rev. A* 110 (1 July 2024), p. 012605.

[GZB&al13] Simon Gustavsson, Olger Zwier, Jonas Bylander, et al. “Improving Quantum Gate Fidelities by Using a Qubit to Measure Microwave Pulse Distortions”. In: *Phys. Rev. Lett.* 110 (4 Jan. 2013), p. 040502.

[HAB&al14] T. P. Harty, D. T. C. Allcock, C. J. Ballance, et al. “High-Fidelity Preparation, Gates, Memory, and Readout of a Trapped-Ion Quantum Bit”. In: *Phys. Rev. Lett.* 113 (22 Nov. 2014), p. 220501.

[HAM&al20] Timothy Hospedales, Antreas Antoniou, Paul Micaelli, et al. *Meta-Learning in Neural Networks: A Survey*. 2020. URL: https://arxiv.org/abs/2004.05439.

[Han] Ozaner Hansha. *Expressing Boolean Logic with Matrices — ozaner.github.io*. [Accessed 28-10-2024].

[Hay09] Simon S. Haykin. *Neural networks and learning machines*. Third. Upper Saddle River, NJ: Pearson Education, 2009.

[Hay17] Masahito Hayashi. *Quantum Information Theory: Mathematical Foundation*. Springer Berlin Heidelberg, 2017.

[HCS&al23] Zoë Holmes, Nolan Coble, Andrew T. Sornborger, et al. *On nonlinear transformations in quantum computation*. 2023. URL: https://arxiv.org/abs/2112.12307.

[HFW20] Robin Harper, Steven T. Flammia, and Joel J. Wallman. “Efficient learning of quantum noise”. In: *Nature Physics* 16.12 (Aug. 2020), pp. 1184–1188.

[HHH96] Michal Horodecki, Pawel Horodecki, and Ryszard Horodecki. *Distillability of Inseparable Quantum Systems*. 1996. URL: https://arxiv.org/abs/quant-ph/9607009.

[HHL09] Aram W. Harrow, Avinatan Hassidim, and Seth Lloyd. "Quantum Algorithm for Linear Systems of Equations". In: *Phys. Rev. Lett.* 103 (15 Oct. 2009), p. 150502.

[HK21] Kentaro Heya and Naoki Kanazawa. "Cross-Cross Resonance Gate". In: *PRX Quantum* 2 (4 Nov. 2021), p. 040336.

[HKL&al20] Philipp Hauke, Helmut G Katzgraber, Wolfgang Lechner, et al. "Perspectives of quantum annealing: methods and implementations". In: *Reports on Progress in Physics* 83.5 (May 2020), p. 054401.

[HKP20] Hsin-Yuan Huang, Richard Kueng, and John Preskill. "Predicting many properties of a quantum system from very few measurements". In: *Nature Physics* 16.10 (Oct. 2020), pp. 1050–1057.

[HKP21] Hsin-Yuan Huang, Richard Kueng, and John Preskill. "Information-Theoretic Bounds on Quantum Advantage in Machine Learning". In: *Physical Review Letters* 126.19 (May 2021).

[HLO&al20] Jonathan Heek, Anselm Levskaya, Avital Oliver, et al. *Flax: A neural network library and ecosystem for JAX*. https://github.com/google/flax. Version 0.4.1. 2020.

[HN19] Amy Hodler and Mark Needham. *Graph algorithms*. Sebastopol, CA: O'Reilly Media, May 2019.

[HNR68] Peter E. Hart, Nils J. Nilsson, and Bertram Raphael. "A Formal Basis for the Heuristic Determination of Minimum Cost Paths". In: *IEEE Transactions on Systems Science and Cybernetics* 4.2 (1968), pp. 100–107.

[Hoc91] S. Hochreiter. "Untersuchungen zu dynamischen neuronalen Netzen." MA thesis. 1991.

[HRB08] H. Häffner, C.F. Roos, and R. Blatt. "Quantum computing with trapped ions". In: *Physics Reports* 469.4 (2008), pp. 155–203.

[HS97] Sepp Hochreiter and Jürgen Schmidhuber. "Long Short-Term Memory". In: *Neural Computation* 9.8 (1997), pp. 1735–1780.

[HSW89] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. "Multilayer feedforward networks are universal approximators". In: *Neural Networks* 2.5 (1989), pp. 359–366.

[HTF09] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. Springer New York, 2009.

[Hub64] Peter J. Huber. "Robust Estimation of a Location Parameter". In: *The Annals of Mathematical Statistics* 35.1 (1964), pp. 73–101.

[HVS&al18] Robert Heck, Oana Vuculescu, Jens Jakob Sørensen, et al. “Remote optimization of an ultracold atoms experiment by experts and citizen scientists”. In: *Proceedings of the National Academy of Sciences* 115.48 (2018), E11231–E11237.

[HWF&al20] He-Liang Huang, Dachao Wu, Daojin Fan, et al. “Superconducting quantum computing: a review”. In: *Science China Information Sciences* 63.8 (July 2020).

[HWM&al22] William J. Huggins, Kianna Wan, Jarrod McClean, et al. “Nearly Optimal Quantum Algorithm for Estimating Multiple Expectation Values”. In: *Phys. Rev. Lett.* 129 (24 Dec. 2022), p. 240501.

[Hyy&al22] Eric Hyyppä et al. “Unimon qubit”. In: *Nature Communications* 13.1 (Nov. 2022), p. 6895.

[IAA&al21] Raza Imam, Qazi Mohammad Areeb, Abdulrahman Alturki, et al. “Systematic and Critical Review of RSA Based Public Key Cryptographic Schemes: Past and Present Status”. In: *IEEE Access* 9 (2021), pp. 155949–155976.

[IBM22] IBM. *IBM Quantum Composer guide.* [Online; accessed 05-January-2022]. 2022.

[ICJ&al05] G. Ithier, E. Collin, P. Joyez, et al. “Decoherence in a superconducting quantum bit circuit”. In: *Phys. Rev. B* 72 (13 Oct. 2005), p. 134519.

[IH00] Satoshi Ishizaka and Tohya Hiroshima. “Maximally entangled mixed states under nonlocal unitary operations in two qubits”. In: *Phys. Rev. A* 62 (2 July 2000), p. 022310.

[Inc] Wolfram Research Inc. *Mathematica, Version 14.1.* Champaign, IL, 2024. URL: `https://www.wolfram.com/mathematica`.

[JE15] Christopher Jackson and S. J. van Enk. “Detecting correlated errors in state-preparation-and-measurement tomography”. In: *Phys. Rev. A* 92 (4 Oct. 2015), p. 042312.

[Jen06] J. L. W. V. Jensen. “Sur les fonctions convexes et les inégalités entre les valeurs moyennes”. In: *Acta Mathematica* 30.none (1906), pp. 175–193.

[Jer&al23a] Sofiene Jerbi et al. “Quantum machine learning beyond kernel methods”. In: *Nature Communications* 14.1 (Jan. 2023), p. 517.

[Jer&al23b] Sofiene Jerbi et al. *The power and limitations of learning quantum dynamics incoherently.* 2023. URL: `https://arxiv.org/abs/2303.12834`.

[JNN13] J. R. Johansson, Paul D. Nation, and Franco Nori. “QuTiP 2: A Python framework for the dynamics of open quantum systems”. In: *Comput. Phys. Commun.* 184.4 (2013), pp. 1234–1240.

[Jon11] Jonathan A. Jones. “Quantum computing with NMR”. In: *Progress in Nuclear Magnetic Resonance Spectroscopy* 59.2 (2011), pp. 91–120.

[Jos62] B.D. Josephson. “Possible new effects in superconductive tunnelling”. In: *Physics Letters* 1.7 (July 1962), pp. 251–253.

[JT09] Nikolay Jetchev and Marc Toussaint. “Trajectory Prediction: Learning to Map Situations to Robot Trajectories”. In: *Proceedings of the 26th Annual International Conference on Machine Learning*. ICML ’09. Montreal, Quebec, Canada: Association for Computing Machinery, 2009, pp. 449–456.

[JTN&al21] Sofiene Jerbi, Lea M. Trenkwalder, Hendrik Poulsen Nautrup, et al. “Quantum Enhancements for Deep Reinforcement Learning in Large Spaces”. In: *PRX Quantum* 2.1 (Feb. 2021).

[Jum&al21] John Jumper et al. “Highly accurate protein structure prediction with AlphaFold”. In: *Nature* 596.7873 (Aug. 2021), pp. 583–589.

[JW28] P. Jordan and E. Wigner. “Über das Paulische Äquivalenzverbot”. In: *Zeitschrift für Physik* 47.9 (Sept. 1928), pp. 631–651.

[KAJ19] Stefan Krastanov, Victor V. Albert, and Liang Jiang. “Optimized Entanglement Purification”. In: *Quantum* 3 (Feb. 2019), p. 123.

[Kay93] Steven M Kay. *Fundamentals of statistical processing, volume I*. Philadelphia, PA: Prentice Hall, Mar. 1993.

[KB15] Diederik P. Kingma and Jimmy Ba. “Adam: A Method for Stochastic Optimization”. In: *CoRR* abs/1412.6980 (2015).

[KBC&al14] J. Kelly, R. Barends, B. Campbell, et al. “Optimal Quantum Control Using Randomized Benchmarking”. In: *Phys. Rev. Lett.* 112 (24 June 2014), p. 240504.

[KE21] Oleksandr Kyriienko and Vincent E. Elfving. “Generalized quantum circuit differentiation rules”. In: *Phys. Rev. A* 104 (5 Nov. 2021), p. 052417.

[Kel17] Matthew Kelly. “An Introduction to Trajectory Optimization: How to Do Your Own Direct Collocation”. In: *SIAM Review* 59.4 (2017), pp. 849–904.

[KGH&al89] J. R. Kuklinski, U. Gaubatz, F. T. Hioe, et al. “Adiabatic population transfer in a three-level system driven by delayed laser pulses”. In: *Phys. Rev. A* 40 (11 Dec. 1989), pp. 6741–6744.

[KGM&al09] B Khani, J M Gambetta, F Motzoi, et al. “Optimal generation of Fock states in a weakly nonlinear oscillator”. In: *Phys. Scr.* T137 (Dec. 2009), p. 014021.

[KHL&al20] Korbinian Kottmann, Patrick Huembeli, Maciej Lewenstein, et al. “Unsupervised Phase Discovery with Deep Anomaly Detection”. In: *Phys. Rev. Lett.* 125 (17 Oct. 2020), p. 170603.

[Kim03] Gen Kimura. “The Bloch vector for N-level systems”. English. In: *Phys. Lett. A* 314.5-6 (Aug. 2003), pp. 339–349.

[Kir04] Donald E Kirk. *Optimal Control Theory*. Dover Books on Electrical Engineering. Mineola, NY: Dover Publications, Apr. 2004.

[KK23] Korbinian Kottmann and Nathan Killoran. *Evaluating analytic gradients of pulse programs on quantum computers*. 2023. URL: `https://arxiv.org/abs/2309.16756`.

[KKL&al18] Susanna Kirchhoff, Torsten Keßler, Per J. Liebermann, et al. "Optimized cross-resonance gate for coupled transmon systems". In: *Phys. Rev. A* 97 (4 Apr. 2018), p. 042348.

[KKY&al19] P. Krantz, M. Kjaergaard, F. Yan, et al. "A quantum engineer's guide to superconducting qubits". In: *Applied Physics Reviews* 6.2 (June 2019), p. 021318.

[KL13] Christoph Kloeffel and Daniel Loss. "Prospects for Spin-Based Quantum Computing in Quantum Dots". In: *Annual Review of Condensed Matter Physics* 4.Volume 4, 2013 (2013), pp. 51–81.

[KL98] E. Knill and R. Laflamme. "Power of One Bit of Quantum Information". In: *Phys. Rev. Lett.* 81 (25 Dec. 1998), pp. 5672–5675.

[KLF&al23] Mario Krenn, Jonas Landgraf, Thomas Foesel, et al. "Artificial intelligence and machine learning for quantum technologies". In: *Phys. Rev. A* 107 (1 Jan. 2023), p. 010101.

[KLP&al19] Sumeet Khatri, Ryan LaRose, Alexander Poremba, et al. "Quantum-assisted quantum compiling". In: *Quantum* 3 (May 2019), p. 140.

[KLR&al08] E. Knill, D. Leibfried, R. Reichle, et al. "Randomized benchmarking of quantum gates". In: *Phys. Rev. A* 77 (1 Jan. 2008), p. 012307.

[KMM&al12] B. Khani, S. T. Merkel, F. Motzoi, et al. "High-fidelity quantum gates in the presence of dispersion". In: *Phys. Rev. A* 85 (2 Feb. 2012), p. 022306.

[KMO&al23] Fabian Kreppel, Christian Melzer, Diego Olvera Millán, et al. "Quantum Circuit Compiler for a Shuttling-Based Trapped-Ion Quantum Computer". In: *Quantum* 7 (Nov. 2023), p. 1176.

[KMS&al21] Thi Ha Kyaw, Tim Menke, Sukin Sim, et al. "Quantum Computer-Aided Design: Digital Quantum Simulation of Quantum Processors". In: *Phys. Rev. Appl.* 16 (4 Oct. 2021), p. 044042.

[KMW02] David Kielpinski, Chris Monroe, and David J Wineland. "Architecture for a large-scale ion-trap quantum computer". In: *Nature* 417.6890 (2002), pp. 709–711.

[KN98] Tadashi Kadowaki and Hidetoshi Nishimori. "Quantum annealing in the transverse Ising model". In: *Physical Review E* 58.5 (Nov. 1998), pp. 5355–5363.

[Koc&al14] Gary Kochenberger et al. "The unconstrained binary quadratic programming problem: a survey". In: *Journal of Combinatorial Optimization* 28.1 (July 2014), pp. 58–81.

[Koc&al22] Christiane P. Koch et al. "Quantum optimal control in quantum technologies. Strategic report on current status, visions and goals for research in Europe". In: *EPJ Quantum Technology* 9.1 (July 2022), p. 19.

[Kor&al22] Maria Korshunova et al. "Generative and reinforcement learning approaches for the automated de novo design of bioactive compounds". In: *Communications Chemistry* 5.1 (Oct. 2022), p. 129.

[KPB21] Ivan Kobyzev, Simon J.D. Prince, and Marcus A. Brubaker. "Normalizing Flows: An Introduction and Review of Current Methods". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 43.11 (2021), pp. 3964–3979.

[KRK&al05] Navin Khaneja, Timo Reiss, Cindie Kehlet, 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.

[KSB&al20] Morten Kjaergaard, Mollie E. Schwartz, Jochen Braumüller, et al. "Superconducting Qubits: Current State of Play". In: *Annu. Rev. Condens. Matter Phys.* 11.1 (2020), pp. 369–395.

[KSC&al20] Sami Khairy, Ruslan Shaydulin, Lukasz Cincio, et al. "Learning to Optimize Variational Quantum Circuits to Solve Combinatorial Problems". In: *Proceedings of the AAAI Conference on Artificial Intelligence* 34.03 (Apr. 2020), pp. 2367–2375.

[KT18] Shauharda Khadka and Kagan Tumer. "Evolution-Guided Policy Gradient in Reinforcement Learning". In: *Advances in Neural Information Processing Systems*. Ed. by S. Bengio et al. Vol. 31. Curran Associates, Inc., 2018.

[KTL&al21] Sangil Kwon, Akiyoshi Tomonaga, Gopika Lakshmi Bhai, et al. "Gate-based superconducting quantum computing". In: *Journal of Applied Physics* 129.4 (Jan. 2021), p. 041102.

[KUM&al17] Günter Klambauer, Thomas Unterthiner, Andreas Mayr, et al. *Self-Normalizing Neural Networks*. 2017. URL: `https://arxiv.org/abs/1706.02515`.

[KVT&al11] T. Kiss, S. Vym ětal, L. D. Tóth, et al. "Measurement-Induced Chaos with Entangled States". In: *Phys. Rev. Lett.* 107 (10 Aug. 2011), p. 100501.

[KW22] Diederik P Kingma and Max Welling. *Auto-Encoding Variational Bayes*. 2022. URL: `https://arxiv.org/abs/1312.6114`.

[KWS&al21] A. Kandala, K. X. Wei, S. Srinivasan, et al. "Demonstration of a High-Fidelity CNOT Gate for Fixed-Frequency Transmons with Engineered ZZ Suppression". In: *Phys. Rev. Lett.* 127 (13 Sept. 2021), p. 130501.

[KYG&al07] Jens Koch, Terri M. Yu, Jay Gambetta, et al. “Charge-insensitive qubit design derived from the Cooper pair box”. In: *Phys. Rev. A* 76 (4 Oct. 2007), p. 042319.

[Lan61] R. Landauer. “Irreversibility and Heat Generation in the Computing Process”. In: *IBM Journal of Research and Development* 5.3 (1961), pp. 183–191.

[LAS&al22] Boxi Li, Shahnawaz Ahmed, Sidhant Saraogi, et al. “Pulse-level noisy quantum circuits with QuTiP”. In: *Quantum* 6 (Jan. 2022), p. 630.

[LB98] Yann LeCun and Yoshua Bengio. “Convolutional networks for images, speech, and time series”. In: *The Handbook of Brain Theory and Neural Networks*. Cambridge, MA, USA: MIT Press, 1998, pp. 255–258.

[LCM22] Boxi Li, Tommaso Calarco, and Felix Motzoi. “Nonperturbative Analytical Diagonalization of Hamiltonians with Application to Circuit QED”. In: *PRX Quantum* 3 (3 July 2022), p. 030313.

[LCM24] Boxi Li, Tommaso Calarco, and Felix Motzoi. “Experimental error suppression in Cross-Resonance gates via multi-derivative pulse shaping”. In: *npj Quantum Information* 10.1 (July 2024), p. 66.

[LD98] Daniel Loss and David P. DiVincenzo. “Quantum computation with quantum dots”. In: *Phys. Rev. A* 57 (1 Jan. 1998), pp. 120–126.

[Lev44] Kenneth Levenberg. “A method for the solution of certain non-linear problems in least squares”. In: *Quarterly of Applied Mathematics* 2.2 (1944), pp. 164–168.

[LHP&al19] Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, et al. *Continuous control with deep reinforcement learning*. 2019. URL: `https://arxiv.org/abs/1509.02971`.

[Li17] Yuxi Li. *Deep Reinforcement Learning: An Overview*. 2017. URL: `https://arxiv.org/abs/1701.07274`.

[LKB&al20] Sanjaya Lohani, Brian T Kirby, Michael Brodsky, et al. “Machine learning assisted quantum state estimation”. In: *Machine Learning: Science and Technology* 1.3 (July 2020), p. 035007.

[Llo82] S. Lloyd. “Least squares quantization in PCM”. In: *IEEE Transactions on Information Theory* 28.2 (1982), pp. 129–137.

[LMK&al22] Haggai Landa, Dekel Meirom, Naoki Kanazawa, et al. “Experimental Bayesian estimation of quantum state preparation, measurement, and gate errors in multiqubit devices”. In: *Physical Review Research* 4.1 (2022), p. 013199.

[LN89] Dong C. Liu and Jorge Nocedal. “On the limited memory BFGS method for large scale optimization”. In: *Math. Program.* 45.1 (Aug. 1989), pp. 503–528.

[Lov99] László Lovász. “Hit-and-run mixes fast”. In: *Math. Prog.* 86.3 (Dec. 1999), pp. 443–461.

[LPG&al20] Seth Lloyd, Giacomo De Palma, Can Gokler, et al. *Quantum algorithm for nonlinear differential equations.* 2020. URL: `https://arxiv.org/abs/2011.06571`.

[LPQ&al24] Jiaqi Leng, Yuxiang Peng, Yi-Ling Qiao, et al. “Differentiable analog quantum computing for optimization and control”. In: *Proceedings of the 36th International Conference on Neural Information Processing Systems.* NIPS ’22. New Orleans, LA, USA: Curran Associates Inc., 2024.

[LPS15] Siu Kwan Lam, Antoine Pitrou, and Stanley Seibert. “Numba: A llvm-based python jit compiler”. In: *Proceedings of the Second Workshop on the LLVM Compiler Infrastructure in HPC.* 2015, pp. 1–6.

[LTW&al24] Martin Larocca, Supanut Thanasilp, Samson Wang, et al. *A Review of Barren Plateaus in Variational Quantum Computing.* 2024. URL: `https://arxiv.org/abs/2405.00781`.

[LV06] László Lovász and Santosh Vempala. “Hit-and-Run from a Corner”. In: *SIAM J. Comput.* 35.4 (2006), p. 985.

[LYP&al17] Jun Li, Xiaodong Yang, Xinhua Peng, et al. “Hybrid Quantum-Classical Approach to Quantum Optimal Control”. In: *Phys. Rev. Lett.* 118 (15 Apr. 2017), p. 150503.

[LZF&al14] Dong-Hyun Lee, Saizheng Zhang, Asja Fischer, et al. *Difference Target Propagation.* 2014. URL: `https://arxiv.org/abs/1412.7525`.

[Mac98] Chiara Macchiavello. “On the analytical convergence of the QPA procedure”. In: *Phys. Lett. A* 246.5 (1998), pp. 385–388.

[Mad&al22] Lars S. Madsen et al. “Quantum computational advantage with a programmable photonic processor”. In: *Nature* 606.7912 (June 2022), pp. 75–81.

[MAG&al21] Alicia B. Magann, Christian Arenz, Matthew D. Grace, et al. “From Pulses to Circuits and Back Again: A Quantum Optimal Control Perspective on Variational Quantum Algorithms”. In: *PRX Quantum* 2 (1 Jan. 2021), p. 010101.

[Mal21] Moein Malekakhlagh. *How the Cross Resonance Gate Works.* [Online; accessed 06-January-2022]. 2021.

[Mal98] Stephane Mallat. *A wavelet tour of signal processing.* en. San Diego, CA: Academic Press, Mar. 1998.

[Mar&al22] Marco Maronese et al. “Quantum compiling”. In: *Quantum Computing Environments.* Springer, 2022, pp. 39–74.

[MAT&al18] Shai Machnes, Elie Assémat, David Tannor, et al. “Tunable, Flexible, and Efficient Optimization of Control Pulses for Practical Qubits”. In: *Phys. Rev. Lett.* 120 (15 Apr. 2018), p. 150401.

[Mau&al15] Julian Mautner et al. “Projective Simulation for Classical Learning Agents: A Comprehensive Investigation”. In: *New Generation Computing* 33.1 (Jan. 2015), pp. 69–114.

[MBD&al12] Cecilia Muldoon, Lukas Brandt, Jian Dong, et al. “Control and manipulation of cold atoms in optical tweezers”. In: *New Journal of Physics* 14.7 (July 2012), p. 073051.

[McC&al18] Jarrod R. McClean et al. “Barren plateaus in quantum neural network training landscapes”. In: *Nature Communications* 9.1 (Nov. 2018), p. 4812.

[MCD&al21] C. Monroe, W. C. Campbell, L.-M. Duan, et al. “Programmable quantum simulations of spin systems with trapped ions”. In: *Rev. Mod. Phys.* 93 (2 Apr. 2021), p. 025001.

[McE&al22] Matt McEwen et al. “Resolving catastrophic error bursts from cosmic rays in large arrays of superconducting qubits”. In: *Nature Physics* 18.1 (Jan. 2022), pp. 107–111.

[Mel&al17] Alexey A. Melnikov et al. “Projective simulation with generalization”. In: *Scientific Reports* 7.1 (Oct. 2017), p. 14430.

[Men&al21] Tim Menke et al. “Automated design of superconducting circuits and its application to 4-local couplers”. In: *npj Quantum Information* 7.1 (Mar. 2021), p. 49.

[MFF14] Tomoyuki Morimae, Keisuke Fujii, and Joseph F. Fitzsimons. “Hardness of Classically Simulating the One-Clean-Qubit Model”. In: *Physical Review Letters* 112.13 (Apr. 2014).

[MGC18] Matthias M. Müller, Stefano Gherardini, and Filippo Caruso. “Noise-robust quantum sensing via optimal multi-probe spectroscopy”. In: *Scientific Reports* 8.1 (Sept. 2018), p. 14278.

[MGJ&al12] Easwar Magesan, Jay M. Gambetta, B. R. Johnson, et al. “Efficient Measurement of Quantum Gate Error by Interleaved Randomized Benchmarking”. In: *Phys. Rev. Lett.* 109 (8 Aug. 2012), p. 080505.

[MGM&al11] F. Motzoi, J. M. Gambetta, S. T. Merkel, et al. “Optimal control methods for rapidly time-varying Hamiltonians”. In: *Phys. Rev. A* 84 (2 Aug. 2011), p. 022307.

[MGR&al09] F. Motzoi, J. M. Gambetta, P. Rebentrost, et al. “Simple Pulses for Elimination of Leakage in Weakly Nonlinear Qubits”. In: *Phys. Rev. Lett.* 103.11 (Sept. 2009), p. 110501.

[MHB&al22] Alan Morningstar, Markus Hauru, Jackson Beall, et al. “Simulation of Quantum Many-Body Dynamics with Tensor Processing Units: Floquet Prethermalization”. In: *PRX Quantum* 3.2 (May 2022).

[MHW&al16] Felix Motzoi, Eli Halperin, Xiaoting Wang, et al. "Backaction-driven, robust, steady-state long-distance qubit entanglement over lossy channels". In: *Phys. Rev. A* 94 (3 Sept. 2016), p. 032313.

[MJW&al01] W. J. Munro, D. F. V. James, A. G. White, et al. "Maximizing the entanglement of two mixed qubits". In: *Phys. Rev. A* 64 (3 Aug. 2001), p. 030302.

[MKG&al09] Vladimir E. Manucharyan, Jens Koch, Leonid I. Glazman, et al. "Fluxonium: Single Cooper-Pair Circuit Free of Charge Offsets". In: *Science* 326.5949 (2009), pp. 113–116.

[MKH&al09] T. Monz, K. Kim, W. Hänsel, et al. "Realization of the Quantum Toffoli Gate with Trapped Ions". In: *Phys. Rev. Lett.* 102 (4 Jan. 2009), p. 040501.

[MKS&al13] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, et al. *Playing Atari with Deep Reinforcement Learning.* 2013. URL: `https://arxiv.org/abs/1312.5602`.

[MKW17] F. Motzoi, M. P. Kaicher, and F. K. Wilhelm. "Linear and Logarithmic Time Compositions of Quantum Many-Body Operators". In: *Phys. Rev. Lett.* 119 (16 Oct. 2017), p. 160503.

[MM22] Moein Malekakhlagh and Easwar Magesan. "Mitigating off-resonant error in the cross-resonance gate". In: *Physical Review A* 105.1 (Jan. 2022).

[MMD&al21] Zlatko K Minev, Thomas G McConkey, Jeremy Drysdale, et al. *Qiskit Metal: An Open-Source Framework for Quantum Device Design & Analysis.* 2021. URL: `https://doi.org/10.5281/zenodo.4618153`.

[MMM20] Moein Malekakhlagh, Easwar Magesan, and David C. McKay. "First-principles analysis of cross-resonance gate operation". In: *Phys. Rev. A* 102 (4 Oct. 2020), p. 042605.

[MMN&al16] Esteban A Martinez, Thomas Monz, Daniel Nigg, et al. "Compiling quantum algorithms for architectures with multi-qubit gates". In: *New Journal of Physics* 18.6 (June 2016), p. 063029.

[Mni&al15] Volodymyr Mnih et al. "Human-level control through deep reinforcement learning". In: *Nature* 518.7540 (Feb. 2015), pp. 529–533.

[Mor&al21] Lorenzo Moro et al. "Quantum compiling by deep reinforcement learning". In: *Communications Physics* 4.1 (Aug. 2021), p. 178.

[MPK&al18] Alexey A. Melnikov, Hendrik Poulsen Nautrup, Mario Krenn, et al. "Active learning machine learns to create new quantum experiments". In: *Proceedings of the National Academy of Sciences* 115.6 (2018), pp. 1221–1226.

[MRF&al20] Shakir Mohamed, Mihaela Rosca, Michael Figurnov, et al. "Monte Carlo Gradient Estimation in Machine Learning". In: *Journal of Machine Learning Research* 21.132 (2020), pp. 1–62.

[MRR&al14] C. Monroe, R. Raussendorf, A. Ruthven, et al. “Large-scale modular quantum-computer architecture with atomic memory and photonic interconnects”. In: *Phys. Rev. A* 89 (2 Feb. 2014), p. 022317.

[MRS24] Glen Bigan Mbeng, Angelo Russomanno, and Giuseppe E. Santoro. “The quantum Ising chain for beginners”. In: *SciPost Physics Lecture Notes* (June 2024).

[MS14] Simon Milz and Walter T Strunz. “Volumes of conditioned bipartite state spaces”. In: *J. Phys. A: Math. Theor.* 48.3 (Dec. 2014), p. 035306.

[MS99] Klaus Mølmer and Anders Sørensen. “Multiparticle Entanglement of Hot Trapped Ions”. In: *Phys. Rev. Lett.* 82 (9 Mar. 1999), pp. 1835–1838.

[MSG&al11] S. Machnes, U. Sander, S. J. Glaser, et al. “Comparing, optimizing, and benchmarking quantum-control algorithms in a unifying programming framework”. In: *Phys. Rev. A* 84 (2 Aug. 2011), p. 022305.

[MSI&al20] Nina Mazyavkina, Sergey Sviridov, Sergei Ivanov, et al. *Reinforcement Learning for Combinatorial Optimization: A Survey*. 2020. URL: https://arxiv.org/abs/2003.03600.

[MSJ&al22] Matthias M Müller, Ressa S Said, Fedor Jelezko, et al. “One decade of quantum optimal control in the chopped random basis”. In: *Reports on Progress in Physics* 85.7 (June 2022), p. 076001.

[MSZ&al06] Mikko Möttönen, Rogerio de Sousa, Jun Zhang, et al. “High-fidelity one-qubit operations under random telegraph noise”. In: *Phys. Rev. A* 73 (2 Feb. 2006), p. 022332.

[Nau&al22] Hendrik Poulsen Nautrup et al. “Operationally meaningful representations of physical systems in neural networks”. In: *Machine Learning: Science and Technology* 3.4 (Dec. 2022), p. 045025.

[NC10] Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information: 10th Anniversary Edition*. Cambridge University Press, 2010.

[NDD&al19] Hendrik Poulsen Nautrup, Nicolas Delfosse, Vedran Dunjko, et al. “Optimizing Quantum Error Correction Codes with Reinforcement Learning”. In: *Quantum* 3 (Dec. 2019), p. 215.

[NGR&al21] Erik Nielsen, John King Gamble, Kenneth Rudinger, et al. “Gate Set Tomography”. In: *Quantum* 5 (Oct. 2021), p. 557.

[Nie00] Jürg Nievergelt. “Exhaustive Search, Combinatorial Optimization and Enumeration: Exploring the Potential of Raw Computing Power”. In: *Proceedings of the 27th Conference on Current Trends in Theory and Practice of Informatics*. SOFSEM ’00. Berlin, Heidelberg: Springer-Verlag, 2000, pp. 18–35.

[Nie02] Michael A Nielsen. “A simple formula for the average gate fidelity of a quantum dynamical operation”. In: *Physics Letters A* 303.4 (2002), pp. 249–252.

[Niu&al19] Murphy Yuezhen Niu et al. “Universal quantum control through deep reinforcement learning”. In: *npj Quantum Information* 5.1 (Apr. 2019), p. 33.

[NM65] J. A. Nelder and R. Mead. “A Simplex Method for Function Minimization”. In: *The Computer Journal* 7.4 (Jan. 1965), pp. 308–313.

[OGB21] Mateusz Ostaszewski, Edward Grant, and Marcello Benedetti. “Structure optimization for parameterized quantum circuits”. In: *Quantum* 5 (Jan. 2021), p. 391.

[Oli08] César R. de Oliveira. *Intermediate Spectral Theory and Quantum Dynamics (Progress in Mathematical Physics, 54)*. English. Basel, Switzerland: Birkhäuser, Nov. 18, 2008, p. 423.

[OPR&al21] Thomas E. O’Brien, Stefano Polla, Nicholas C. Rubin, et al. “Error Mitigation via Verified Phase Estimation”. In: *PRX Quantum* 2 (2 May 2021), p. 020317.

[ORC&al22] Nimba Oshnik, Phila Rembold, Tommaso Calarco, et al. “Robust magnetometry with single nitrogen-vacancy centers via two-step optimization”. In: *Phys. Rev. A* 106 (1 July 2022), p. 013107.

[Ort&al01] G. Ortiz et al. “Quantum algorithms for fermionic simulations”. In: *Phys. Rev. A* 64 (2 July 2001), p. 022319.

[OSF&al91] T. Ozaki, T. Suzuki, T. Furuhashi, et al. “Trajectory control of robotic manipulators using neural networks”. In: *IEEE Transactions on Industrial Electronics* 38.3 (1991), pp. 195–202.

[Ost&al21] Mateusz Ostaszewski et al. “Reinforcement learning for optimization of variational quantum circuit architectures”. In: *Advances in Neural Information Processing Systems*. Ed. by M. Ranzato et al. Vol. 34. Curran Associates, Inc., 2021, pp. 18182–18194.

[PAB&al20] S. Pirandola, U. L. Andersen, L. Banchi, et al. “Advances in quantum cryptography”. In: *Adv. Opt. Photon.* 12.4 (Dec. 2020), pp. 1012–1236.

[PAO23] Stefano Polla, Gian-Luca R. Anselmetti, and Thomas E. O’Brien. “Optimizing the information extracted by a single qubit measurement”. In: *Phys. Rev. A* 108 (1 July 2023), p. 012403.

[Par06] G. S. Paraoanu. “Microwave-induced coupling of superconducting qubits”. In: *Phys. Rev. B* 74 (14 Oct. 2006), p. 140504.

[Pau03] V. Paulsen. *Completely Bounded Maps and Operator Algebras*. Cambridge: Cambridge University Press, 2003.

[Pau90] Wolfgang Paul. “Electromagnetic traps for charged and neutral particles”. In: *Rev. Mod. Phys.* 62 (3 July 1990), pp. 531–540.

[PB24] Francesco Preti and József Zsolt Bernád. “Statistical evaluation and optimization of entanglement purification protocols”. In: *Phys. Rev. A* 110 (2 Aug. 2024), p. 022619.

[PCM22] Francesco Preti, Tommaso Calarco, and Felix Motzoi. “Continuous Quantum Gate Sets and Pulse-Class Meta-Optimization”. In: *PRX Quantum* 3 (4 Oct. 2022), p. 040311.

[PCT&al22] Francesco Preti, Tommaso Calarco, Juan Mauricio Torres, et al. “Optimal two-qubit gates in recurrence protocols of entanglement purification”. In: *Phys. Rev. A* 106 (2 Aug. 2022), p. 022422.

[Per&al14] Alberto Peruzzo et al. “A variational eigenvalue solver on a photonic quantum processor”. In: *Nature Communications* 5.1 (July 2014), p. 4213.

[Per85] Asher Peres. “Reversible logic and quantum computers”. In: *Phys. Rev. A* 32 (6 Dec. 1985), pp. 3266–3276.

[PGM&al19] Adam Paszke, Sam Gross, Francisco Massa, et al. “PyTorch: An Imperative Style, High-Performance Deep Learning Library”. In: *Advances in Neural Information Processing Systems 32*. Ed. by H. Wallach et al. Curran Associates, Inc., 2019, pp. 8024–8035.

[PKJ&al22] Attila Portik, Orsolya Kálmán, Igor Jex, et al. “Iterated nth order nonlinear quantum dynamics with mixed initial states”. In: *Phys. Lett. A* 431 (2022), p. 127999.

[PMC&al24] Alice Pagano, Matthias M Müller, Tommaso Calarco, et al. *The Role of Bases in Quantum Optimal Control*. 2024. URL: https://arxiv.org/abs/2405.20889.

[Pon18] L.S. Pontryagin. *Mathematical Theory of Optimal Processes*. Routledge, May 2018.

[Por&al19] Riccardo Porotti et al. “Coherent transport of quantum states by deep reinforcement learning”. In: *Communications Physics* 2.1 (June 2019), p. 61.

[Pou&al04] David Poulin et al. “Exponential Speedup with a Single Bit of Quantum Information: Measuring the Average Fidelity Decay”. In: *Phys. Rev. Lett.* 92 (17 Apr. 2004), p. 177906.

[PPM23] Riccardo Porotti, Vittorio Peano, and Florian Marquardt. “Gradient-Ascent Pulse Engineering with Feedback”. In: *PRX Quantum* 4.3 (July 2023).

[Pre&al25] Francesco Preti et al. *Gradients, parallelism, and variance of quantum estimates*. 2025. URL: https://arxiv.org/abs/2509.11214.

[Pre18] John Preskill. “Quantum computing in the NISQ era and beyond”. In: *Quantum* 2 (2018), p. 79.

[Pre20] Francesco Preti. “Deep projective simulation and state preparation”. MA thesis. University of Innsbruck, 2020.

[Pre22] Francesco Preti. *GitHub - franz3105/OptEntDist: Optimal Entanglement Purification — github.com*. [Accessed 24-09-2024]. 2022.

[PRF&al22] Kirill Plekhanov, Matthias Rosenkranz, Mattia Fiorentini, et al. “Variational quantum amplitude estimation”. In: *Quantum* 6 (Mar. 2022), p. 670.

[Pro&al20] Timothy Proctor et al. “Detecting and tracking drift in quantum information processors”. In: *Nature Comm.* 11.1 (Oct. 2020), p. 5396.

[PSF&al19] Daniel K Park, Ilya Sinayskiy, Mark Fingerhuth, et al. “Parallel quantum trajectories via forking for sampling without redundancy”. In: *New Journal of Physics* 21.8 (Aug. 2019), p. 083024.

[PSJ&al24] Francesco Preti, Michael Schilling, Sofiene Jerbi, et al. “Hybrid discrete-continuous compilation of trapped-ion quantum circuits with deep reinforcement learning”. In: *Quantum* 8 (May 2024), p. 1343.

[PV07] Martin B. Plenio and Shashank Virmani. “An introduction to entanglement measures”. In: *Quantum Info. Comput.* 7.1 (Jan. 2007), pp. 1–51.

[Qis23] Qiskit contributors. *Qiskit: An Open-source Framework for Quantum Computing*. 2023.

[RAC&al24] Ben W. Reichardt, David Aasen, Rui Chao, et al. *Demonstration of quantum computation and error correction with a tesseract code*. 2024. URL: `https://arxiv.org/abs/2409.04628`.

[Ral06] L B Rall. *Automatic differentiation*. 1981st ed. Lecture Notes in Computer Science. Berlin, Germany: Springer, Jan. 2006.

[RBL&al22] Robin Rombach, Andreas Blattmann, Dominik Lorenz, et al. *High-Resolution Image Synthesis with Latent Diffusion Models*. 2022. URL: `https://arxiv.org/abs/2112.10752`.

[RBM18] Nicholas C Rubin, Ryan Babbush, and Jarrod McClean. “Application of fermionic marginal constraints to hybrid quantum algorithms”. In: *New Journal of Physics* 20.5 (May 2018), p. 053020.

[RD10] Chad Rigetti and Michel Devoret. “Fully microwave-tunable universal gates in superconducting qubits with linear couplings and fixed transition frequencies”. In: *Phys. Rev. B* 81 (13 Apr. 2010), p. 134507.

[Rie05] Martin Riedmiller. “Neural Fitted Q Iteration – First Experiences with a Data Efficient Neural Reinforcement Learning Method”. In: *Machine Learning: ECML 2005*. Ed. by João Gama et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 317–328.

[RMB19] Katja Ried, Thomas Müller, and Hans J. Briegel. "Modelling collective motion based on the principle of agency: General framework and the case of marching locusts". In: *PLOS ONE* 14.2 (Feb. 2019). Ed. by Iman Borazjani, e0212044.

[RMC&al15] N. Rach, M. M. Müller, T. Calarco, et al. "Dressing the chopped-random-basis optimization: A bandwidth-limited access to the trap-free landscape". In: *Phys. Rev. A* 92 (6 Dec. 2015), p. 062343.

[RMP&al22] Martin Ringbauer, Michael Meth, Lukas Postler, et al. "A universal qudit quantum processor with trapped ions". In: *Nature Physics* 18.9 (July 2022), pp. 1053–1057.

[RN94] Gavin A Rummery and Mahesan Niranjan. *On-line Q-learning using connectionist systems*. Vol. 37. University of Cambridge, Department of Engineering Cambridge, UK, 1994.

[ROM&al20] Phila Rembold, Nimba Oshnik, Matthias M. Müller, et al. "Introduction to quantum optimal control for quantum sensing with nitrogen-vacancy centers in diamond". In: *AVS Quantum Science* 2.2 (June 2020), p. 024701.

[Ros58] F. Rosenblatt. "The perceptron: A probabilistic model for information storage and organization in the brain." In: *Psychological Review* 65.6 (1958), pp. 386–408.

[Rud17] Sebastian Ruder. *An overview of gradient descent optimization algorithms*. 2017. URL: `https://arxiv.org/abs/1609.04747`.

[RZV&al23] Domenico Ribezzo, Mujtaba Zahidy, Ilaria Vagniluca, et al. "Deploying an inter-European quantum network". In: *Advanced Quantum Technologies* 6.2 (2023), p. 2200061.

[SA10] Stefan Schaal and Christopher G. Atkeson. "Learning Control in Robotics". In: *IEEE Robotics Automation Magazine* 17.2 (2010), pp. 20–29.

[SAC&al21] Louis Schatzki, Andrew Arrasmith, Patrick J. Coles, et al. *Entangled Datasets for Quantum Machine Learning*. 2021. URL: `https://arxiv.org/abs/2109.03400`.

[Saf16] M Saffman. "Quantum computing with atomic qubits and Rydberg interactions: progress and challenges". In: *Journal of Physics B: Atomic, Molecular and Optical Physics* 49.20 (Oct. 2016), p. 202001.

[SB18] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. Second. The MIT Press, 2018.

[SB22] A. Sauer and J. Z. Bernád. "Quantitative characterization of several entanglement detection criteria". In: *Phys. Rev. A* 106 (3 Sept. 2022), p. 032423.

[SBM&al21] A Sauer, J Z Bernád, H J Moreno, et al. "Entanglement in bipartite quantum systems: Euclidean volume ratios and detectability by Bell inequalities". In: *J. Phys. A: Math. Theor.* 54.49 (Nov. 2021), p. 495302.

[Sch&al03] Ferdinand Schmidt-Kaler et al. "Realization of the Cirac–Zoller controlled-NOT quantum gate". In: *Nature* 422.6930 (Mar. 2003), pp. 408–411.

[Sch&al22] Sophie G. Schirmer et al. "Robust Control Performance for Open Quantum Systems". In: *IEEE Transactions on Automatic Control* 67.11 (2022), pp. 6012–6024.

[SD12] Paul B Slater and Charles F Dunkl. "Moment-based evidence for simple rational-valued Hilbert–Schmidt generic 2×2 separability probabilities". In: *J. Phys. A: Math. Theor.* 45.9 (Feb. 2012), p. 095305.

[SEL&al22] V. V. Sivak, A. Eickbusch, H. Liu, et al. "Model-Free Quantum Control with Reinforcement Learning". In: *Physical Review X* 12.1 (Mar. 2022).

[SES&al20] Nikitas Stamatopoulos, Daniel J. Egger, Yue Sun, et al. "Option Pricing using Quantum Computers". In: *Quantum* 4 (July 2020), p. 291.

[SFM&al23] Aaron Somoroff, Quentin Ficheux, Raymond A. Mencia, et al. "Millisecond Coherence in a Superconducting Qubit". In: *Phys. Rev. Lett.* 130 (26 June 2023), p. 267001.

[SGD&al21] Ross Shillito, Jonathan A Gross, Agustin Di Paolo, et al. "Fast and differentiable simulation of driven quantum systems". In: *Physical Review Research* 3.3 (2021), p. 033266.

[SHC&al17] Tim Salimans, Jonathan Ho, Xi Chen, et al. *Evolution Strategies as a Scalable Alternative to Reinforcement Learning*. 2017. URL: `https://arxiv.org/abs/1703.03864`.

[She06] Dan Shepherd. *Computation with Unitaries and One Pure Qubit*. 2006. URL: `https://arxiv.org/abs/quant-ph/0608132`.

[Sho94] P.W. Shor. "Algorithms for quantum computation: discrete logarithms and factoring". In: *Proceedings 35th Annual Symposium on Foundations of Computer Science*. 1994, pp. 124–134.

[SHS&al18] Alvaro Sanchez-Gonzalez, Nicolas Heess, Jost Tobias Springenberg, et al. "Graph networks as learnable physics engines for inference and control". In: *International conference on machine learning*. PMLR. 2018, pp. 4470–4479.

[Shy21] Prasanth Shyamsundar. *Non-Boolean Quantum Amplitude Amplification and Quantum Mean Estimation*. 2021. URL: `https://arxiv.org/abs/2102.04975`.

[Sil&al16] David Silver et al. "Mastering the game of Go with deep neural networks and tree search". In: *Nature* 529.7587 (Jan. 2016), pp. 484–489.

[Sil&al17] David Silver et al. “Mastering the game of Go without human knowledge”. In: *Nature* 550.7676 (Oct. 2017), pp. 354–359.

[Sim13] Dan Simon. *Evolutionary optimization algorithms*. Hoboken, NJ: Wiley-Blackwell, Apr. 2013.

[SJ08] Peter W. Shor and Stephen P. Jordan. “Estimating Jones polynomials is a complete problem for one clean qubit”. In: *Quantum Info. Comput.* 8.8 (Sept. 2008), pp. 681–714.

[SLH&al14] David Silver, Guy Lever, Nicolas Heess, et al. “Deterministic Policy Gradient Algorithms”. In: *Proceedings of the 31st International Conference on Machine Learning*. Ed. by Eric P. Xing and Tony Jebara. Vol. 32. Proceedings of Machine Learning Research 1. Bejing, China: PMLR, June 2014, pp. 387–395.

[SLM&al17] John Schulman, Sergey Levine, Philipp Moritz, et al. *Trust Region Policy Optimization*. 2017. URL: `https://arxiv.org/abs/1502.05477`.

[SM00] Anders Sørensen and Klaus Mølmer. “Entanglement and quantum computation with ions in thermal motion”. In: *Phys. Rev. A* 62 (2 July 2000), p. 022311.

[SM99] Anders Sørensen and Klaus Mølmer. “Quantum Computation with Ions in Thermal Motion”. In: *Physical Review Letters* 82.9 (Mar. 1999), pp. 1971–1974.

[Smi84] Robert L. Smith. “Efficient Monte Carlo Procedures for Generating Points Uniformly Distributed over Bounded Regions”. In: *Oper. Res.* 32.6 (1984), pp. 1296–1308.

[SMS&al99] Richard S Sutton, David McAllester, Satinder Singh, et al. “Policy Gradient Methods for Reinforcement Learning with Function Approximation”. In: *Advances in Neural Information Processing Systems*. Ed. by S. Solla, T. Leen, and K. Müller. Vol. 12. MIT Press, 1999.

[SN20] J. J. Sakurai and Jim Napolitano. *Modern Quantum Mechanics*. 3rd ed. Cambridge University Press, 2020.

[SNR&al21] Muhammed Saeed, Mohammed Nagdi, Benjamin Rosman, et al. “Deep Reinforcement Learning for Robotic Hand Manipulation”. In: *2020 International Conference on Computer, Control, Electrical, and Electronics Engineering (ICCCEEE)*. 2021, pp. 1–5.

[SOG&al02] R. Somma, G. Ortiz, J. E. Gubernatis, et al. “Simulating physical phenomena by quantum networks”. In: *Phys. Rev. A* 65 (4 Apr. 2002), p. 042323.

[Son98] Eduardo D. Sontag. *Mathematical Control Theory*. Springer New York, 1998.

[SP19] Sergei Slussarenko and Geoff J. Pryde. “Photonic quantum information processing: A concise review”. In: *Applied Physics Reviews* 6.4 (Oct. 2019).

[SPM&al24] Michael Schilling, Francesco Preti, Matthias M. Müller, et al. *Exponentiation of Parametric Hamiltonians via Unitary interpolation*. 2024. URL: https://arxiv.org/abs/2402.01498.

[SSN&al15a] Yi-Lin Seah, Jiangwei Shang, Hui Khoon Ng, et al. “Monte Carlo sampling from the quantum state space. II”. In: *New J. Phys.* 17.4 (Apr. 2015), p. 043018.

[SSN&al15b] Jiangwei Shang, Yi-Lin Seah, Hui Khoon Ng, et al. “Monte Carlo sampling from the quantum state space. I”. In: *New J. Phys.* 17.4 (Apr. 2015), p. 043017.

[Ste07] D.A. Steck. *Quantum and Atom Optics*. 2007. URL: http://steck.us/teaching.

[Sti55] W. Forrest Stinespring. “Positive Functions on C*-Algebras”. In: *Proceedings of the American Mathematical Society* 6.2 (1955), pp. 211–216.

[Str69] Volker Strassen. “Gaussian elimination is not optimal”. In: *Numerische Mathematik* 13.4 (Aug. 1969), pp. 354–356.

[SUR&al20] Yohichi Suzuki, Shumpei Uno, Rudy Raymond, et al. “Amplitude estimation without phase estimation”. In: *Quantum Information Processing* 19.2 (Jan. 2020).

[SWD&al17] John Schulman, Filip Wolski, Prafulla Dhariwal, et al. *Proximal Policy Optimization Algorithms*. 2017. URL: https://arxiv.org/abs/1707.06347.

[SWM10] M. Saffman, T. G. Walker, and K. Mølmer. “Quantum information with Rydberg atoms”. In: *Rev. Mod. Phys.* 82 (3 Aug. 2010), pp. 2313–2363.

[SZC&al23] Juhi Singh, Robert Zeier, Tommaso Calarco, et al. “Compensating for Nonlinear Distortions in Controlled Quantum Systems”. In: *Phys. Rev. Appl.* 19 (6 June 2023), p. 064067.

[Tak99] Minoru Takahashi. *Thermodynamics of One-Dimensional Solvable Models*. Cambridge University Press, 1999.

[TB06] Andrew G. Taube and Rodney J. Bartlett. “New perspectives on unitary coupled-cluster theory”. In: *International Journal of Quantum Chemistry* 106.15 (2006), pp. 3393–3401.

[TB16] Juan Mauricio Torres and József Zsolt Bernád. “Conditions for entanglement purification with general two-qubit states”. In: *Phys. Rev. A* 94 (5 Nov. 2016), p. 052329.

[TBS02] Todd Tilma, Mark Byrd, and E C G Sudarshan. “A parametrization of bipartite systems based on SU(4) Euler angles”. In: *J. Phys. A: Math. Theor.* 35.48 (Nov. 2002), p. 10445.

[TMM&al18] L. S. Theis, F. Motzoi, S. Machnes, et al. “Counteracting systems of diabaticities using DRAG controls: The status after 10 years”. In: *EPL (Europhysics Letters)* 123.6 (Oct. 2018), p. 60001.

[TMW16] L. S. Theis, F. Motzoi, and F. K. Wilhelm. “Simultaneous gates in frequency-crowded multilevel systems using fast, robust, analytic control shapes”. In: *Phys. Rev. A* 93 (1 Jan. 2016), p. 012324.

[Tof80] Tommaso Toffoli. “Reversible computing”. In: *Automata, Languages and Programming*. Ed. by Jaco de Bakker and Jan van Leeuwen. Berlin, Heidelberg: Springer Berlin Heidelberg, 1980, pp. 632–644.

[Tro59] H. F. Trotter. “On the product of semi-groups of operators”. In: *Proceedings of the American Mathematical Society* 10.4 (Apr. 1959), pp. 545–545.

[TS02] Todd Tilma and E C G Sudarshan. “Generalized Euler angle parametrization for SU(N)”. In: *Journal of Physics A: Mathematical and General* 35.48 (Nov. 2002), p. 10467.

[TSH18] Gao Tang, Weidong Sun, and Kris Hauser. “Learning Trajectories for Real- Time Optimal Control of Quadrotors”. In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Madrid, 2018, pp. 3620–3625.

[VC21] Tyler Volkoff and Patrick J Coles. “Large gradients via correlation in random parameterized quantum circuits”. In: *Quantum Science and Technology* 6.2 (Jan. 2021), p. 025008.

[VD04] G. Vidal and C. M. Dawson. “Universal quantum circuit for two-qubit transformations with three controlled-NOT gates”. In: *Phys. Rev. A* 69 (1 Jan. 2004), p. 010301.

[VDO&al19] Davide Venturelli, Minh Do, Bryan O’Gorman, et al. “Quantum Circuit Compilation: An Emerging Application for Automated Reasoning”. In: *Scheduling and Planning Applications Workshop*. 2019.

[Ver00] Arun Verma. “An introduction to automatic differentiation”. In: *Current Science* (2000), pp. 804–807.

[Vin&al19] Oriol Vinyals et al. “Grandmaster level in StarCraft II using multi-agent reinforcement learning”. In: *Nature* 575.7782 (Nov. 2019), pp. 350–354.

[Vir&al20] Pauli Virtanen et al. “SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python”. In: *Nature Methods* 17 (2020), pp. 261–272.

[VMV&al22] Vipul Jee Verma, Alok Kumar Mishra, D. Vaithiyanathan, et al. “Review of Different Flip-Flop Circuits and a Modified Flip-Flop Circuit for Low Voltage Operation”. In: *2022 IEEE 3rd Global Conference for Advancement in Technology (GCAT)*. 2022, pp. 1–5.

[VRS&al17] Nikolay V. Vitanov, Andon A. Rangelov, Bruce W. Shore, et al. “Stimulated Raman adiabatic passage in physics, chemistry, and beyond”. In: *Rev. Mod. Phys.* 89 (1 Mar. 2017), p. 015006.

[VW04] Farrokh Vatan and Colin Williams. “Optimal quantum circuits for general two-qubit gates”. In: *Phys. Rev. A* 69 (3 Mar. 2004), p. 032315.

[Was10] Larry Wasserman. *All of statistics : a concise course in statistical inference*. New York: Springer, 2010.

[Wat&al15] Paul Watts et al. “Optimizing for an arbitrary perfect entangler. I. Functionals”. In: *Phys. Rev. A* 91 (6 June 2015), p. 062306.

[WD92] Christopher J. C. H. Watkins and Peter Dayan. “Q-learning”. In: *Machine Learning* 8.3–4 (May 1992), pp. 279–292.

[WDD&al19] Re-Bing Wu, Haijin Ding, Daoyi Dong, et al. “Learning robust and high-precision quantum controls”. In: *Phys. Rev. A* 99 (4 Apr. 2019), p. 042327.

[Wer89] Reinhard F. Werner. “Quantum states with Einstein-Podolsky-Rosen correlations admitting a hidden-variable model”. In: *Phys. Rev. A* 40 (8 Oct. 1989), pp. 4277–4281.

[WHB20] Robert Wille, Stefan Hillmich, and Lukas Burgholzer. “Efficient and Correct Compilation of Quantum Circuits”. In: *2020 IEEE International Symposium on Circuits and Systems (ISCAS)*. 2020, pp. 1–5.

[WHT15] Dave Wecker, Matthew B. Hastings, and Matthias Troyer. “Progress towards practical quantum variational algorithms”. In: *Phys. Rev. A* 92 (4 Oct. 2015), p. 042303.

[WHT16] Dave Wecker, Matthew B. Hastings, and Matthias Troyer. In: *Phys. Rev. A* 94 (2 Aug. 2016), p. 022309.

[Wil&al20] Frank K. Wilhelm et al. *An introduction into optimal control for quantum technologies*. 2020. URL: `https://arxiv.org/abs/2003.10132`.

[Wil92] Ronald J. Williams. “Simple statistical gradient-following algorithms for connectionist reinforcement learning”. In: *Machine Learning* 8.3 (May 1992), pp. 229–256.

[Wim94] S. Wimperis. “Broadband, Narrowband, and Passband Composite Pulses for Use in Advanced NMR Experiments”. In: *Journal of Magnetic Resonance, Series A* 109.2 (1994), pp. 221–231.

[WIW&al22] David Wierichs, Josh Izaac, Cody Wang, et al. “General parameter-shift rules for quantum gradients”. In: *Quantum* 6 (Mar. 2022), p. 677.

[WKW&al16] Robert Wille, Oliver Keszocze, Marcel Walter, et al. “Look-ahead schemes for nearest neighbor optimization of 1D and 2D quantum circuits”. In: *2016 21st Asia and South Pacific Design Automation Conference (ASP-DAC)*. 2016, pp. 292–297.

[WLW&al24] Roeland Wiersema, Dylan Lewis, David Wierichs, et al. “Here comes the SU(N): multivariate quantum gates and gradients”. In: *Quantum* 8 (Mar. 2024), p. 1275.

[WMD&al20] Julius Wallnöfer, Alexey A. Melnikov, Wolfgang Dür, et al. In: *PRX Quantum* 1 (1 Sept. 2020), p. 010301.

[Woo98] William K. Wootters. “Entanglement of Formation of an Arbitrary State of Two Qubits”. In: *Phys. Rev. Lett.* 80 (10 Mar. 1998), pp. 2245–2248.

[WRD93] Warren S. Warren, Herschel Rabitz, and Mohammed Dahleh. “Coherent Control of Quantum Dynamics: The Dream Is Alive”. In: *Science* 259.5101 (1993), pp. 1581–1589.

[WSG&al14] Daan Wierstra, Tom Schaul, Tobias Glasmachers, et al. “Natural Evolution Strategies”. In: *Journal of Machine Learning Research* 15.27 (2014), pp. 949–980.

[Yad20] Vivek Yadav. *Direct collocation for optimal control — mec560sbu.github.io*. [Accessed 15-09-2024]. 2020.

[YIL&al21] Ed Younis, Costin C Iancu, Wim Lavrijsen, et al. *Berkeley Quantum Synthesis Toolkit (BQSKit) v1*. Apr. 2021. URL: `https://www.osti.gov/biblio/1785933`.

[YLB21] 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.

[YSK&al20] Fei Yan, Youngkyu Sung, Philip Krantz, et al. *Engineering Framework for Optimizing Superconducting Qubit Designs*. 2020. URL: `https://arxiv.org/abs/2006.04130`.

[YYW18] Xu-Chen Yang, Man-Hong Yung, and Xin Wang. “Neural-network-designed pulse sequences for robust control of singlet-triplet qubits”. In: *Phys. Rev. A* 97 (4 Apr. 2018), p. 042324.

[ZBL23] Ji Zou, Stefano Bosco, and Daniel Loss. *Spatially correlated classical and quantum noise in driven qubits: The good, the bad, and the ugly*. 2023. URL: `https://arxiv.org/abs/2308.03054`.

[ZCH&al21] Jie Zhou, Ganqu Cui, Shengding Hu, et al. *Graph Neural Networks: A Review of Methods and Applications*. 2021. URL: `https://arxiv.org/abs/1812.08434`.

[Zeh02] H. D. Zeh. *Decoherence: Basic Concepts and Their Interpretation*. 2002. URL: `https://arxiv.org/abs/quant-ph/9506020`.

[ŻPN&al11] Karol Życzkowski, Karol A. Penson, Ion Nechita, et al. “Generating random density matrices”. In: *J. Math. Phys.* 52.6 (June 2011), p. 062201.

[ŻS01] Karol Życzkowski and Hans-Jürgen Sommers. “Induced measures in the space of mixed quantum states”. In: *J. Phys. A: Math. Gen.* 34.35 (Aug. 2001), p. 7111.

[Zur03] Wojciech H. Zurek. *Decoherence and the transition from quantum to classical – REVISITED*. 2003. URL: https://arxiv.org/abs/quant-ph/0306072.

[ZZZ&al20] Yuan-Hang Zhang, Pei-Lin Zheng, Yi Zhang, et al. "Topological Quantum Compiling with Reinforcement Learning". In: *Phys. Rev. Lett.* 125 (17 Oct. 2020), p. 170501.

Schlüsseltechnologien / Key Technologies
Band / Volume 302
ISBN 978-3-95806-856-8