

Quantum Error Mitigation for CNOT-gates

Anders Håøy Rokne

December 18, 2020

Abstract

Quantum computing is a rapidly developing field, with the potential to solve problems in real-world applications in the not-so-distant future that are not possible to efficiently solve on any classical computer. In the short term, however, quantum hardware will be limited both in the number of qubits, the basic unit of quantum information, and the length of useful computations due to the influence of noise. These devices are commonly referred to as noisy intermediate-scale quantum (NISQ) devices. While in the long term, quantum error correction codes have the potential to allow arbitrarily long computations performed to arbitrary precision, they require lots of ancillary qubits and are thus not viable for the near-future quantum computation. Our attention therefore turns to Quantum error mitigation (QEM), which encompasses a range of techniques that aim to reduce the impact of noise on quantum computations. This does not require a large amount of ancillary qubits, and is therefore applicable to NISQ-devices.

In this specialization project, we study possible near-term applications of quantum computers. We focus on hybrid quantum-classical algorithms, which are algorithms that combine short quantum computations with classical computation, as well as on QEM techniques. Both are of high importance as they increase the reach of NISQ devices. We focus on a QEM technique based on zero-noise extrapolation. By deliberately amplifying noise, the technique extrapolates the results to the noise-free limit. We propose a new variant of the zero-noise extrapolation technique for noise in the CNOT-gate that utilizes that the gate is its own inverse.

We demonstrate a possible use-case for near-term hybrid algorithms by the quantum approximate optimization algorithm (QAOA). Using a simulator, apply the QAOA to a specific instance of the max-cut problem, and show that the algorithm achieves an approximate solution with a cost of 82% of the optimal value. This decreased to 72% when using a physical quantum device, showing the impact of noise and the need for quantum error mitigation. Furthermore, we apply our variant of the zero-noise extrapolation technique to a SWAP-test circuit with 3 qubits, with promising results. This gave an improvement from the noisy result of 77% of the noise-free value, to a mitigated result of 99%.

Contents

Abstract	iii
Contents	v
1 Introduction	1
2 Preliminary Theory	5
2.1 Linear algebra	5
2.1.1 Inner product spaces	5
2.1.2 Homomorphisms	6
2.2 Quantum Mechanics	7
2.2.1 The State Space	7
2.2.2 Time evolution	7
2.2.3 Measurements	8
2.3 Density operators and Quantum Channels	8
2.3.1 The density operator	8
2.3.2 Trace and partial trace	9
2.3.3 The Bloch sphere	10
2.3.4 Quantum channels	12
2.3.5 Operator-sum representation	13
2.4 Efficiency of algorithms	13
2.4.1 Asymptotic notation	14
2.5 Quantum Computation	14
2.5.1 The Quantum Bit	14
2.5.2 Multi-qubit states	15
2.5.3 Quantum computations and circuit notation	16
2.5.4 Quantum Gates	17
2.5.5 Controlled gates	19
2.5.6 Gate identities	20
2.5.7 Basis set of gates	22
2.5.8 Measurements of Quantum Circuits	23
2.5.9 Expectation values	24
2.5.10 Error estimation	25
3 Error Models for Quantum Computation	27
3.1 Noise channels	27
3.2 Coherent noise	28
3.3 Incoherent noise	28

3.3.1	Stochastic Pauli-noise	28
3.3.2	Amplitude damping	29
3.3.3	Phase Damping	29
3.4	Quantum Error Correction	30
3.4.1	The Classical Repetition code	30
3.4.2	The Quantum Repetition code	31
3.4.3	Digitization of errors	32
3.4.4	Error Thresholds	33
4	Hybrid Algorithms	35
4.1	Variational Quantum Methods	35
4.2	Quantum approximate optimization algorithm	36
4.2.1	The max-cut problem	36
4.2.2	A Quantum Encoding	37
4.2.3	Quantum optimization for max-cut	37
4.2.4	Quantum Circuit for the $p=1$ algorithm	38
4.2.5	Time complexity of the expectation value evaluations	40
5	Quantum Error Mitigation	41
5.1	Zero-noise extrapolation	41
5.1.1	Richardson extrapolation	42
5.1.2	Variance scaling of the Richardson extrapolation	43
5.1.3	Noise amplification by repeating CNOT-gates	43
5.2	Quasi-probability decomposition	44
5.2.1	Decomposition of an ideal circuit	44
5.2.2	Per-gate decomposition	45
6	Implementation	47
6.1	Qiskit	47
6.1.1	Quantum Circuits in <code>qiskit</code>	47
6.1.2	Backend specifications and the transpiler	48
6.1.3	Error models and mock backends	51
6.2	The QAOA for max-cut with $p=1$	52
6.3	Zero-noise amplification for CNOT gates	53
7	Results and discussion	55
7.1	QAOA for Max-Cut with $p=1$	55
7.2	Zero-noise extrapolation for CNOT gates	58
8	Conclusion	63
	Bibliography	65

Chapter 1

Introduction

The onset of quantum computation promises exponential speed up for specific computational tasks when compared to existing classical algorithms. By utilizing the distinct physical properties of quantum mechanics and quantum mechanical systems, quantum computers may soon surpass the capabilities of classical computers for certain applications. As recently as in 2019, a research team at Google announced that they had achieved *quantum advantage*, or *quantum supremacy*, by computing a specific task on a quantum computer faster than what is thought possible on any classical supercomputer [1]; an important milestone on the way to practically applicable quantum computation.

The main idea of quantum computation is to encode information into *qubits* with quantum mechanical properties. In classical computation the most fundamental unit of information is the bit, which is either 0 or 1. The quantum analogue is the qubit. The qubit is physically realized in a quantum mechanical system where the equivalents of the bits 0 and 1 are encoded as *quantum states* [2]. Say for example a spin up/down particle, where we encode the bit value of 0 as spin up, and 1 as spin down. Now we may have a qubit in the state 0, in the state 1, or in any superposition of the two. Quantum computations are most often formulated as quantum circuits. In general, we first initialize some quantum state on a set of qubits. This quantum state is evolved in time by applying a sequence of quantum operations or *quantum gates*, analogous to logical gates of classical logical circuits. Lastly, a set of quantum measurements are performed on the culminating final state, resulting in a set of measurement *outcomes*. Usually, these outcomes are post-processed by classical computation to finalize the algorithm output, which often is an estimation of the expectation value of some operator with respect to a quantum state.

Very oversimplified, the advantages of quantum computation can be understood as follows; on a classical computer with n bits of memory, we can only store and operate on a single bit-string of length n at a time. On a quantum computer with n qubits, however, we can create superpositions of all the 2^n basis states corresponding to each of the n -bit bit-strings, and do operations on all of them at once. This is sometimes referred to as *quantum parallelism*. Such

a state would in general require 2^n bits to be represented on a classical computer, i.e., increasing exponentially with n . The reality is more far complex of course. Still, this offers a rough idea in layman terms of how quantum computation offers something fundamentally distinct from the capabilities of classical computers, with the potential of exponential speed up in certain circumstances.

A first intuitive application for such a quantum-mechanical computer is simulations of quantum systems. Intuitively, representing physical quantum systems using qubits, possessing the same quantum-mechanical properties of superpositions and entanglement, may be more efficient than by using classical bits. By enabling numerical simulations of quantum systems, such as molecules, more efficiently and on a larger scale than what is feasible by classical computation, quantum computation could open new doors in material science and solid state physics. An example algorithm is the *variational quantum eigensolver* (VQE), which utilizes quantum computation to estimate ground state energies of molecules [3].

Other possible applications for quantum computation have been suggested within fields such as classical optimisation. One example is the quantum approximate optimization algorithm (QAOA), proposed in [4]. Perhaps the most well-known quantum algorithm, however, is Shor’s integer factorisation algorithm [5]. This algorithm factorises integers efficiently, more precisely, it factorises an integer encoded as an n -bit bit-string in polynomial time as a function of n . This is a task for which there exists no known efficient classical algorithms. In fact, certain widely used encryption schemes rely on the assumption that integer factorisation for large integers is hard. Quantum computers will eventually break this assumption, though this is still in the distant future.

Further advancements in quantum hardware is nonetheless required before quantum computers may become useful for real-world applications. There are two main restrictions imposed by current quantum hardware. Quantum computers are affected by noise, as quantum systems are notoriously difficult to keep coherent for long times, and they are limited in their scale, by which we mean the number of qubits available and connected in a quantum computer. The near-term era of quantum computation has thus been coined the *noisy intermediate-scale quantum* (NISQ) era.

Quantum error correction by quantum error codes provides means to repeatedly detect and correct errors throughout the run of a quantum computation [6]. This can be done to an arbitrary precision, given that the error rates on physical qubits are below a certain threshold given by the specific error correcting code, and given enough available ancillary qubits. This provides promise for the long-term viability of quantum computation.

For the short-term, however, quantum error-correcting codes are far too demanding in their qubit requirements. This brings us on to the second main limitation of NISQ-era devices; their limited scale. The state-of-the-art quantum processor used to achieve quantum advantage had 53 connected qubits. Mainly, the problem is to have enough qubits both available and connected, such that

they may interact with each other. This interaction is key to the advantages of quantum computers. In the near-future era, a few hundred to a few thousand connected qubits is considered to be realistic. This is still far from the millions of qubits that have been estimated to be needed to factorize integers on the size used in modern encryption schemes with Shor’s factorisation algorithm [7].

The proposed near-term applications are therefore hybrid quantum/classical algorithms. Such algorithms consist of a short-depth quantum computation, combined with classical computation. Both the aforementioned VQE, and the QAOA, are such hybrid algorithms. The underlying idea for both is to encode the solution of the given problem as the ground state of a quantum Hamiltonian operator. A quantum subprocedure prepares some ansatz state parametrized by a set of variational parameters and estimates the expectation value of the state with respect to the Hamiltonian. The classical part consists of a classical optimisation algorithm that minimises this expectation value with respect to the variational parameters. The ground-state eigenvalue forms a lower bound, which one hopes to approach [3, 4].

To combat noise in NISQ-era quantum devices, for which quantum error correction codes are infeasible, we turn to quantum error mitigation, i.e., techniques aiming to reduce the impact of noise. This is in contrast to quantum error correction, which targets exact correction of sets of errors. Error mitigation can often be achieved without similarly large ancillary qubit requirements as needed in quantum error correction.

Several quantum error-mitigation schemes have been proposed in recent years [8–15]. Amongst them the quasi-probability decomposition, with which one mitigates noise by expressing the ideal noise-free quantum circuit as a sum of noisy circuits. We can then sample from the noisy circuits by a Monte Carlo scheme to estimate the noise-free expectation value. Another proposed scheme is the zero-noise extrapolation. The idea is to amplify the noise in our quantum circuits by a set of known factors, then compute the expectation values for each of the noise-amplified circuits. From this, we can then extrapolate the expectation value in question down to the zero-noise case.

The main focus of this specialization project is on quantum error mitigation, with applications to near-term viable quantum algorithms. The goals are to obtain a strong knowledge within the field of theoretical quantum computation and quantum information theory, focusing on near-future applicable algorithms and techniques, to implement and demonstrate some key concepts within hybrid algorithms and quantum error mitigation, and to examine a new variation of the zero-noise extrapolation technique that we propose.

We propose a scheme for amplifying noise in the multi-qubit CNOT operation specifically, by replacing each CNOT-gate in a circuit by a series of repeated, identical CNOT-gates, utilizing that the CNOT is its own inverse. The noise in the CNOT-gates can then be mitigated by the zero-noise extrapolation. This noise amplification scheme has, to the best of our knowledge, not been researched previously.

This project report is laid out in the following way; first, Chapter 2 gives the preliminary theory and prerequisites needed to understand quantum computation, and the further concepts of hybrid algorithms and quantum error mitigation that will be presented. This includes the basics of quantum mechanics, leading up to the density operator and quantum channel formulation of quantum mechanics, which are commonly used in quantum information theory, as well as the quantum computation as quantum circuits. To give a framework to understand and formulate noise present in modern quantum hardware, Chapter 3 presents and defines some useful error models. A brief introduction to quantum error correcting codes is also given, to explain why they are useful for the long-term, but may have limited viability for the short-term.

To understand the possible near-future applications for quantum computers, Chapter 4 presents the main aspects of hybrid algorithms. We focus on a specific, proposed hybrid algorithm, namely the quantum approximate optimization algorithm for the max-cut problem. Chapter 5 further focuses on quantum error-mitigation techniques that are hoped to be applicable to the said near-future algorithms, extending their range of viability on NISQ-era quantum hardware. Specifically, the quasi-probability decomposition and the zero-noise extrapolation schemes are explored in detail. For the latter, our proposed noise amplification scheme is presented.

Chapter 6 explains some concepts important for the later implementations of some specific algorithms. This includes certain crucial aspects of `qiskit`, which is an open-source software development kit for quantum computation with Python, developed by IBM [16, 17]. We have implemented two specific techniques: The quantum approximate optimization algorithm, as an example of a hybrid algorithm, and the zero-noise extrapolation with our proposed noise amplification scheme. The main details of these implementations are presented in this chapter. The code has been made available in an online, public repository at github.com/AndersHR/quantum_error_mitigation.

We have employed our implementations of the quantum approximate optimization algorithm and the zero-noise extrapolation on some chosen example problems, and the results are presented and discussed in Chapter 7. The quantum approximate optimization algorithm was run on an example problem both with a noise-less simulator and a physical 5-qubit quantum device provided by IBM. This was done to demonstrate the capabilities of hybrid algorithms on a small scale, as well as to examine the effect of noise present on the real quantum device. The zero-noise extrapolation was employed on a quantum circuit known as the SWAP-test circuit. This was done to demonstrate the concept of quantum error mitigation, as well as to provide a proof-of-concept for our proposed noise amplification scheme and to examine its potential. Finally, the project is summarized in Chapter 8.

Chapter 2

Preliminary Theory

This chapter explains the preliminary theory needed to understand quantum computation and the concepts later presented. We define a few key concepts of linear algebra, which forms the mathematical foundation of quantum mechanics, and present the basics of quantum mechanics. We further formulate quantum information theory within the density operator formalism.

The theory presented here is well-established. The interested reader can find the subjects explained and explored more in detail in text books such as *Quantum Computation and Quantum Information* by Nielsen and Chuang, and others [2, 18–20].

2.1 Linear algebra

The basis of quantum mechanics is that of linear algebra. We present here some concepts that will be important when defining the postulates of quantum mechanics, and for the density operator and quantum channel formulation of quantum mechanics. We define the inner product space, which will be crucial for the concept of quantum-mechanical state vectors, and the homomorphism. The latter will become important when defining the quantum density operator, and the quantum channel, which are important objects in quantum information theory.

2.1.1 Inner product spaces

An *inner product space* is a vector space V over a field \mathbb{F} (e.g., the real numbers \mathbb{R} or the complex numbers \mathbb{C}), equipped with an inner product

$$\langle \cdot, \cdot \rangle : V \times V \rightarrow \mathbb{F} \tag{2.1}$$

which is a map that for all $x, y, z \in V$ and $\alpha \in \mathbb{F}$ fulfills the following properties:

1. Linearity:

$$\langle \alpha x, y \rangle = \alpha \langle x, y \rangle, \quad (2.2)$$

$$\langle x + z, y \rangle = \langle x, y \rangle + \langle z, y \rangle. \quad (2.3)$$

2. Conjugate symmetry:

$$\overline{\langle x, y \rangle} = \langle y, x \rangle. \quad (2.4)$$

3. Positive semi definite:

$$\langle x, x \rangle > 0 \Leftrightarrow x \neq 0. \quad (2.5)$$

The inner product $\langle \cdot, \cdot \rangle$ of an inner-product space V induces a norm $\|\cdot\|$; that is, a map

$$\|\cdot\| : V \rightarrow \mathbb{R}, \quad (2.6)$$

defined for all $x \in V$ as

$$\|x\| = \sqrt{\langle x, x \rangle}. \quad (2.7)$$

2.1.2 Homomorphisms

A *homomorphism* between two inner product spaces A and B is a linear map $F : A \rightarrow B$ that preserves the inner product. By this we mean the following: denote $\langle \cdot, \cdot \rangle_A$ as the inner product on A , and $\langle \cdot, \cdot \rangle_B$ as the inner product on B . Then

$$\langle F(x), F(y) \rangle_B = \langle x, y \rangle_A \quad (2.8)$$

holds for all $x, y \in A$.

Denote the space of homomorphisms between inner product spaces A and B as $\text{Hom}(A, B)$. A homomorphism from a space V onto itself is called an *endomorphism*, and the space of endomorphisms on an inner product space V is denoted $\text{End}(V)$; i.e., $\text{End}(V) = \text{Hom}(V, V)$.

For a homomorphism $S \in \text{Hom}(A, B)$ on inner product spaces A and B with inner products $\langle \cdot, \cdot \rangle_A$ and $\langle \cdot, \cdot \rangle_B$, respectively, the *adjoint* or *Hermitian conjugate* of S , denoted S^\dagger , is the unique operator such that

$$\langle S(a), b \rangle_B = \langle a, S^\dagger(b) \rangle_A \quad (2.9)$$

for all $a \in A$ and $b \in B$. Note that $(S^\dagger)^\dagger = S$. Furthermore, S is called

- normal if $SS^\dagger = S^\dagger S$,
- unitary if $SS^\dagger = S^\dagger S = \mathbb{I}$, where \mathbb{I} is the identity operator,
- Hermitian if $S^\dagger = S$.

For S being an endomorphism on an inner product space A , S is said to be *positive* if $0 \leq \langle a, S(a) \rangle_A$ for all $a \in A$.

2.2 Quantum Mechanics

The postulates of quantum mechanics provides a mathematical framework for quantum physics. Note that the postulates in themselves do not offer any physics, only a foundation in which to model quantum-mechanical systems. There exists several, equivalent formulations of the postulates. We present an algebraic formulation, that most straight-forwardly leads up to the density operator formalism, which is the commonly used formalism in quantum computation and quantum information theory [2, 19].

2.2.1 The State Space

Postulate 1. For every isolated physical system, there is associated a complex Hilbert space H , known as the *state space*. The state of the system is completely described by a vector $|\psi\rangle \in H$, the *state vector*, with norm equal to 1.

The state vector is unique up to a global phase factor, $|\psi\rangle \rightarrow e^{i\alpha} |\psi\rangle$ for any $\alpha \in \mathbb{R}$. This will become clear later, when we consider quantum measurements later in this chapter. For every state vector $|\psi\rangle$, which is called a *ket* vector, there exists a corresponding *bra* vector $\langle\psi|$, which is a vector in its dual space. The inner product between two state vectors $|\psi\rangle$ and $|\phi\rangle$ is denoted $\langle\phi, \psi\rangle = \langle\phi| \cdot |\psi\rangle = \langle\phi|\psi\rangle$. This notation is referred to as the *Bra-Ket notation*.

2.2.2 Time evolution

Postulate 2. Time evolution of a closed quantum system in state $|\psi\rangle \in H$ is described by a unitary operator $U : H \rightarrow H$ derived from the time-dependent Schrödinger equation

$$i\hbar \frac{d}{dt} |\psi\rangle = \mathcal{H} |\psi\rangle, \quad (2.10)$$

where \mathcal{H} is the Hamiltonian of the system and \hbar is the Heisenberg's reduced constant.

The Hamiltonian is a Hermitian operator, whose eigenvalues are energies $\{E_i\}_i$ with eigenvectors $\{|i\rangle\}_i$. The eigenvectors form a complete basis for the Hilbert space, and the state vector can thus be expressed as a linear combination $|\psi\rangle = \sum_i c_i |i\rangle$ for a set of complex coefficients c_i .

The time evolution of a state vector $|\psi\rangle$ from time t' to t , given a time-independent Hamiltonian $\mathcal{H}(t) = \mathcal{H}$, is thus given as

$$U(t, t') |\psi\rangle = e^{-i\hbar(t-t')\mathcal{H}} |\psi\rangle = \sum_i c_i e^{-i\hbar E_i(t-t')} |i\rangle. \quad (2.11)$$

It is common practice in literature to set $\hbar = 1$ by rescaling the units, in order to ease notation.

2.2.3 Measurements

Postulate 3. For every measurable physical quantity of a system, there is associated an Hermitian operator A with eigenvalues a and eigenvectors $|a\rangle$. A quantum measurement will probabilistically return a measurement result equal to one of the eigenvalues a . As A is Hermitian operator there exist a spectral decomposition

$$A = \sum_a a P_a \quad (2.12)$$

where P_a is the projection operator onto the eigenspace associated with the eigenvalue a . If the system is in state $|\psi\rangle$, then the probability of a measurement obtaining the measurement result a is

$$p_a = \langle \psi | P_a | \psi \rangle, \quad (2.13)$$

and the state will afterwards be left in the post-measurement state

$$|\psi'\rangle = \frac{U P_a |\psi\rangle}{\sqrt{\langle \psi | P_a^\dagger P_a | \psi \rangle}}, \quad (2.14)$$

where U is some possible time evolution operator. An equivalent formulation of quantum measurements can be written in terms of the set of measurement operators $\{M_a\}_a$, where $M_a = U P_a$.

2.3 Density operators and Quantum Channels

Until now we have considered the description of a quantum system in terms of a wave function $|\psi\rangle$ as a vector in a Hilbert space H . The density operator formalism, where the quantum system is described by a density operator ρ , offers a more general description, that also let us describe *open* quantum systems, quantum *subsystem*, and *ensembles* of state. We will further define general maps between density operators in terms of *completely-positive trace-preserving maps*, referred to as *quantum channels*.

2.3.1 The density operator

A *density operator*, also referred to as a *density matrix*, is an operator $\rho \in \text{End}(H)$ for a Hilbert space H such that

$$\rho = \sum_k p_k |\psi_k\rangle \langle \psi_k|, \quad (2.15)$$

where $|\psi_k\rangle \in H$ are orthogonal quantum states and the coefficients p_k form a probability distribution; i.e., $p_k \geq 0$ for all k and

$$\sum_k p_k = 1. \quad (2.16)$$

Note that the term *state* is often used interchangeably for quantum states $|\psi\rangle$ and density operators ρ . Whether we mean one or the other becomes clear from the context. A density operator is said to be *pure* if it can be written as

$$\rho = |\psi\rangle\langle\psi|, \quad (2.17)$$

for some state vector $|\psi\rangle$. A density operator which is not pure is said to be *mixed*.

A closed, isolated quantum system described by the quantum state $|\psi\rangle$ is in the density-operator formalism described by the pure density operator $|\psi\rangle\langle\psi|$. As an example, take the composite state $|\psi_{AB}\rangle \in H_A \otimes H_B$ given by

$$|\psi_{AB}\rangle = \frac{1}{\sqrt{2}}(|0\rangle_A \otimes |0\rangle_B + |1\rangle_A \otimes |1\rangle_B) \equiv \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle), \quad (2.18)$$

that gives the corresponding density operator as the pure density operator

$$\rho_{AB} = |\psi_{AB}\rangle\langle\psi_{AB}| = \frac{1}{2}(|00\rangle\langle 00| + |00\rangle\langle 00| 11 + |11\rangle\langle 11| 00 + |11\rangle\langle 11|) \quad (2.19)$$

for $|0\rangle$ and $|1\rangle$ being two orthonormal basis states for H_A and H_B . Compare this to the following example of a mixed state

$$\rho' = \frac{1}{2}(|00\rangle\langle 00| + |11\rangle\langle 11|). \quad (2.20)$$

Density operators are particularly useful for describing ensembles of states. As an example, assume that we start out with a quantum state $|\psi\rangle$ and an error occurs with probability p that transforms the state into $|\psi'\rangle = M|\psi\rangle$. We do not know if an error has occurred, only the probability that it has done so. The state afterwards can then be described by the mixed state

$$\rho' = (1-p)|\psi\rangle\langle\psi| + p|\psi'\rangle\langle\psi'|. \quad (2.21)$$

2.3.2 Trace and partial trace

The *trace* of a density operator, and later the *partial trace*, are concepts that will become useful for both analysis, and for later definitions. We define the trace of a density operator $\rho \in \text{End}(H)$ as

$$\text{Tr}[\rho] = \sum_l \langle e_l | \rho | e_l \rangle \quad (2.22)$$

for $\{|e_L\rangle\}_l$ being some orthonormal basis of H . The requirement given in (2.16) is then equivalent to requiring $\text{Tr}[\rho] = 1$. This becomes evident by choosing the basis such that $|e_l\rangle = |\psi_l\rangle$. Furthermore, an operator ρ is pure if and only if $\text{Tr}[\rho^2] = 1$.

For a composite system AB and $\rho \in \text{End}(H_A \otimes H_B)$, define the *partial trace* of ρ onto A as

$$\text{Tr}_B[\rho] = \sum_l \langle e_l | \rho | e_l \rangle \quad (2.23)$$

for $\{|e_L\rangle\}_l$ being some orthonormal basis of H_B . The resulting operator $\rho_A \equiv \text{Tr}_B[\rho]$ is a density operator in $\text{End}(H_A)$.

For any density operator $\rho_A \in \text{End}(H_A)$ one can always find a density operator $\rho_{AB} \in \text{End}(H_{AB})$ on some larger composite system AB such that ρ_{AB} is pure and $\text{Tr}_B[\rho_{AB}] = \rho_A$. ρ_{AB} is then called a *purification* of ρ_A .

As an example, let us look at the pure state $\rho_{AB} = |\psi_{AB}\rangle \langle \psi_{AB}|$ as given in Equation (2.19). Taking the partial trace of ρ_{AB} onto A we can describe the state on

$$\rho_A = \text{Tr}_B[|\psi_{AB}\rangle \langle \psi_{AB}|] = \frac{1}{2}(|0\rangle \langle 0|_A + |1\rangle \langle 1|_A). \quad (2.24)$$

The expectation value of an observable M with respect to some density operator ρ is given as

$$\langle M \rangle_\rho = \text{Tr}[M\rho]. \quad (2.25)$$

2.3.3 The Bloch sphere

The *Bloch sphere* representation provides a geometrical representation for density operators on a certain class of quantum-mechanical systems; quantum systems of dimension equal to 2, e.g., a qubit system.

Theorem 2.1. Any density operator ρ on a quantum system of dimension 2 can be written on the form

$$\rho = \frac{1}{2}(\mathbb{I} + \vec{r} \cdot \vec{\sigma}) \quad (2.26)$$

for a real vector $\vec{r} = (r_x, r_y, r_z)$, such that $\|\vec{r}\| \leq 1$ and $\vec{\sigma} = (\sigma_x, \sigma_y, \sigma_z)$, where $\{\sigma_i\}_{i=x,y,z}$ are the Pauli matrices defined as

$$\sigma_z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}, \quad \sigma_y = \begin{bmatrix} 0 & i \\ -i & 0 \end{bmatrix}, \quad \sigma_x = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}. \quad (2.27)$$

Thus the Bloch sphere representation gives a geometrical representation for general density operators ρ on 2-dimensional quantum systems, in terms of the vector $\vec{r} \in \mathbb{R}^3$. The vector \vec{r} is referred to as the *Bloch vector* of ρ . This can be represented as in Figure 2.1.

The Bloch sphere representation of general 2-level density operators can be proved in the following way. First, write the density operator ρ for a 2-level system as a 2×2 complex-valued general matrix

$$\rho = \begin{bmatrix} a & b \\ c & d \end{bmatrix}. \quad (2.28)$$

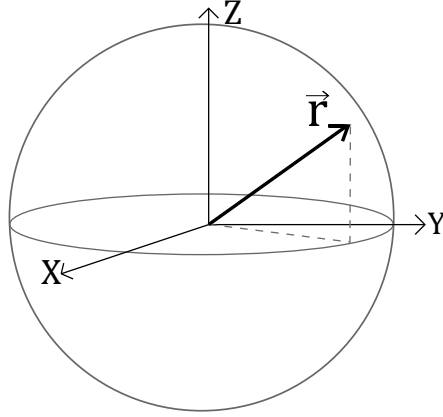


Figure 2.1: A geometrical representation of a density operator $\rho = \frac{1}{2}(\mathbb{I} + \vec{r} \cdot \vec{\sigma})$ by its Bloch vector \vec{r} on the Bloch sphere.

We will use the properties of density matrices to show that ρ can always be written on the form of Equation (2.26). From $\text{Tr}[\rho] = 1$ we get

$$\text{Tr}[\rho] = a + d = 1, \quad (2.29)$$

where we have used the fact that the trace of matrices is the sum of its diagonal elements. Furthermore, observe that

$$\rho^\dagger = \left(\sum_k p_k |\psi_k\rangle \langle \psi_k| \right)^\dagger = \sum_k p_k (|\psi_k\rangle \langle \psi_k|)^\dagger = \sum_k p_k |\psi_k\rangle \langle \psi_k| = \rho, \quad (2.30)$$

such that

$$\rho^\dagger = \begin{bmatrix} a^* & c^* \\ b^* & d^* \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} = \rho \quad (2.31)$$

where we have used $A^\dagger = (A^*)^T$ for A a complex matrix. From this we see that $a^* = a$ and $d^* = d$, implying $a, d \in \mathbb{R}$. This, together with Equation (2.29), implies that we can write $a = \frac{1}{2}(1 + r_z)$ and $d = \frac{1}{2}(1 - r_z)$ for some $r_z \in \mathbb{R}$.

From Equation (2.31) it also follows that $c^* = b$ and $b^* = c$, such that we may write $c = \frac{1}{2}(r_x - ir_y)$, $b = \frac{1}{2}(r_x + ir_y)$ for some $r_x, r_y \in \mathbb{R}$. Collecting the terms,

$$\rho = \begin{bmatrix} a & b \\ c & d \end{bmatrix} = \frac{1}{2} \begin{bmatrix} 1 + r_z & r_x - ir_y \\ r_x + ir_y & 1 - r_z \end{bmatrix} = \frac{1}{2}(\mathbb{I} + \vec{r} \cdot \vec{\sigma}) \quad (2.32)$$

Observe that $\text{Tr}[\rho^2] = \frac{1}{2}(1 + |\vec{r}|^2)$. From the fact that $\text{Tr}[\rho^2] \leq 1$, with equality if and only if ρ is pure, we conclude that the bloch vector \vec{r} is contained within the unit sphere in \mathbb{R}^3 . The states represented by the vectors on the surface of the unit sphere are exactly the pure states.

2.3.4 Quantum channels

We define a *quantum channel*, or *quantum operation*, \mathcal{E} as a *trace-preserving completely positive linear map* from density operators $\rho \in \text{End}(H_A)$ to density operators $\rho' \in \text{End}(H_B)$,

$$\mathcal{E} : \text{End}(H_A) \rightarrow \text{End}(H_B). \quad (2.33)$$

With trace-preserving we mean that the trace of ρ is preserved under \mathcal{E} , i.e.,

$$\text{Tr}[\mathcal{E}(\rho)] = \text{Tr}[\rho]. \quad (2.34)$$

A map $\mathcal{E} : \text{End}(H_A) \rightarrow \text{End}(H_B)$ is said to be *positive* if it fulfills the property that if $\rho \in \text{End}(H_A)$ is a positive operator, then $\mathcal{E}(\rho) \in \text{End}(H_B)$ is also a positive operator.

A map $\mathcal{E} : \text{End}(H_A) \rightarrow \text{End}(H_B)$ is said to be *completely positive* if it fulfills the slightly more restrictive restriction that if $\rho \in \text{End}(H_R \otimes H_A)$ a positive operator, then $(\mathbb{I} \otimes \mathcal{E})(\rho) \in \text{End}(H_R \otimes H_B)$ a positive operator.

The motivation behind defining quantum channels as trace-preserving and completely positive maps is to preserve the properties of quantum density operators, which are positive operators with trace equal to one. The stricter requirement of complete positivity ensures that a quantum channel acting locally on a part of a system still preserves positivity of the total, composite state.

A special class of quantum channels are the unitary channels $\mathcal{N}(\rho) = U\rho U^\dagger$ for U a unitary operator. The unitary evolution of a quantum state vector $|\psi\rangle$ by a unitary operator U , as described in Section 2.2.2, can be written within the density operator formalism as the unitary quantum channel $\mathcal{N}(|\psi\rangle\langle\psi|) = U|\psi\rangle\langle\psi|U^\dagger$ on the pure state $|\psi\rangle\langle\psi|$. Note that a unitary channel preserves purity, as

$$\rho' = \mathcal{N}(|\psi\rangle\langle\psi|) = U|\psi\rangle\langle\psi|U^\dagger = |\psi'\rangle\langle\psi'|, \quad (2.35)$$

where we have defined $|\psi'\rangle = U|\psi\rangle$.

Theorem 2.2 (The Stinespring Dilation). For any quantum channel $\mathcal{E} : \text{End}(H_A) \rightarrow \text{End}(H_B)$ there exists a unitary $U \in \text{Hom}(\text{End}(H_A), \text{End}(H_B) \otimes \text{End}(H_C))$, and a unitary $\tilde{U} \in \text{Hom}(\text{End}(H_A) \otimes \text{End}(H_C), \text{End}(H_B) \otimes \text{End}(H_C))$, such that

$$\mathcal{E}(\rho) = \text{Tr}_C[\tilde{U}(\rho \otimes |e\rangle\langle e|_C)\tilde{U}^\dagger] = \text{Tr}_C[U\rho U^\dagger] \quad (2.36)$$

for any $\rho \in \text{End}(H_A)$; $|e\rangle_C$ is a unit vector in an ancillary Hilbert space H_C .

The Stinespring dilation gives the interpretation that any quantum channel \mathcal{E} can be seen as a unitary channel, with unitary \tilde{U} , on a larger composite system $\rho \otimes |e\rangle\langle e|_C \in \text{End}(H_A \otimes H_C)$. The channel acting locally on the subsystem of interest is found by tracing out the system C .

2.3.5 Operator-sum representation

An efficient representation of general quantum channels is the *operator-sum representation*. By (2.36) we can write any arbitrary quantum channel \mathcal{E} as

$$\mathcal{E}(\rho) = \text{Tr}_C[U \rho U^\dagger] = \sum_i \langle e_i | U \rho U^\dagger | e_i \rangle_C = \sum_i E_i \rho E_i^\dagger, \quad (2.37)$$

where $\{|e_i\rangle_C\}_i$ is some basis for the ancillary Hilbert space H_C , and we have defined the operators $E_i \equiv U |e_i\rangle_C$. The operators $\{E_i\}_i$ are called the *Kraus operators* of \mathcal{E} .

Note the following restriction, which must hold for any valid set of Kraus operators describing a quantum channel,

$$\sum_i E_i^\dagger E_i = \sum_i U^\dagger |e_i\rangle \langle e_i|_C U = U^\dagger \left(\sum_i |e_i\rangle \langle e_i| \right) U = U^\dagger \mathbb{I} U = \mathbb{I}. \quad (2.38)$$

We have used that $\sum_i |e_i\rangle \langle e_i| = \mathbb{I}$, for $\{|e_i\rangle\}_i$ being a complete, orthonormal basis, and \mathbb{I} being the identity operator, and that U is unitary. Note that the freedom in choosing the basis $\{|e_i\rangle\}_i$ implies that the Kraus operators are not uniquely defined.

As an example, take the error channel as described in Equation (2.21), where the pure state $|\psi\rangle \langle \psi|$ is transformed into a mixed state after an error M may have occurred with probability p . This case can be described as a quantum channel by defining the Kraus operators

$$E_0 = \sqrt{1-p} \mathbb{I}, \quad E_1 = \sqrt{p} M. \quad (2.39)$$

These completely describe the given quantum error channel \mathcal{E} , as

$$\begin{aligned} \mathcal{E}(\rho) &= E_0 |\psi\rangle \langle \psi| E_0^\dagger + E_1 |\psi\rangle \langle \psi| E_1^\dagger \\ &= (1-p) \mathbb{I} |\psi\rangle \langle \psi| \mathbb{I} + p M |\psi\rangle \langle \psi| M^\dagger \\ &= (1-p) |\psi\rangle \langle \psi| + p |\psi'\rangle \langle \psi'| = \rho'. \end{aligned} \quad (2.40)$$

They also fulfill the requirement given in Equation (2.38), as

$$E_0^\dagger E_0 + E_1^\dagger E_1 = (1-p) \mathbb{I} + p M^\dagger M = (1-p+p) \mathbb{I} = \mathbb{I}, \quad (2.41)$$

where we use that M is unitary.

2.4 Efficiency of algorithms

The *complexity* of an algorithm is a metric of the resources, typically either the amount of time or memory needed for an algorithm to finish as a function of the *input size*. A typical example is a classical algorithm with an input encoded in a bit string of length n . The integer n is then taken to be the input size.

The *running time* of an algorithm is the number of primitive steps needed for the algorithm to finish, for a given input. It is common practice to write the running time in *leading order* in the input size, as this is most often the most informative measure. Say an algorithm runs in time $f(n) = an^2 + bn + c$. For large n , the leading order term an^2 dominates, and other terms become negligible.

2.4.1 Asymptotic notation

Categorizing algorithms by the leading order growth of the running time is captured by use of *asymptotic notation* [20].

For a function $g(n)$, we denote by $\Theta(g(n))$ the set of functions defined by

$$\Theta(g(n)) = \{f(n) \text{ such that there exists constants } c_0, c_1 \geq 0 \text{ and } n_0 \text{ such that } c_0g(n) \leq f(n) \leq c_1g(n) \forall n \geq n_0\}. \quad (2.42)$$

The function $f(n) = an^2 + bn + c$ is in the set $\Theta(n^2)$, but not in $\Theta(n^3)$ or $\Theta(n)$. We may write $f(n) \in \Theta(n^2)$, but it is, however, common in literature to write $f(n) = \Theta(n^2)$. We say that $g(n) = n^2$ is an *asymptotically tight bound* for $f(n)$.

Furthermore, for a function $g(n)$, we denote by $\mathcal{O}(g(n))$ the set of functions defined by

$$\mathcal{O}(g(n)) = \{f(n) \text{ such that there exists constants } c \geq 0 \text{ and } n_0 \text{ such that } f(n) \leq cg(n) \forall n \geq n_0\} \quad (2.43)$$

A function $f(n) = an^2 + bn + c$ is thus in both $\mathcal{O}(n^2)$, and in $\mathcal{O}(n^3)$, but not in $\mathcal{O}(n)$.

A function $f(n)$ is said to be *polynomially bounded* if $f(n) = \mathcal{O}(n^m)$ for some constant m . We define the set $\text{poly}(n)$ as the set of all polynomially bounded functions.

An algorithm is said to be *efficient* if it runs in a polynomially bound running time. As an example, take the function $f(n) = 2^n$. This function grows *exponentially* with n , and cannot be asymptotically bound by any polynomial function. It is therefore *not* said to be efficient.

2.5 Quantum Computation

So far, we have presented the basics of quantum mechanics, and the density operator and quantum channel formalism. This chapter employs the aforementioned theory to build a basis in which to model *quantum computations*.

2.5.1 The Quantum Bit

In classical computation the fundamental unit of information is the bit, which has a value of either 0 or 1. Analogously, the quantum bit, or qubit, forms the

most fundamental unit of information for quantum computation. The qubit is encoded into some quantum system, taking advantage of the distinct quantum-mechanical properties of superposition and entanglement. An arbitrary qubit state may be written

$$|\psi\rangle = \alpha |0\rangle + \beta |1\rangle, \quad (2.44)$$

for $\alpha, \beta \in \mathbb{C}$ and $\{|0\rangle, |1\rangle\}$ being the *computational basis*, which is some orthonormal basis spanning the Hilbert space $H \cong \mathbb{C}^2$ of the qubit system. When measuring the state $|\psi\rangle$ in the computational basis, we get a measurement result of 0 with probability $|\langle 0|\psi\rangle|^2 = |\alpha|^2$ and 1 with probability $|\langle 1|\psi\rangle|^2 = |\beta|^2$. Normalization of the state $|\psi\rangle$ requires that $|\alpha|^2 + |\beta|^2 = 1$.

Note that there is nothing inherently special about the computational basis and other bases may be constructed. For example the $\{|+\rangle, |-\rangle\}$ basis can be constructed as

$$|+\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle), \quad |-\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle), \quad (2.45)$$

which is an equivalent orthonormal, complete basis for the same Hilbert space. Still, it is useful to define a canonical basis for which to write out qubit states.

2.5.2 Multi-qubit states

To construct the space of multi qubit states, we take the tensor product of single-qubit state spaces. For a 2-qubit system with qubits A and B, we take the Hilbert space of the combined system to be

$$H_{AB} = H_A \otimes H_B, \quad (2.46)$$

on which a general state can be written as

$$|\psi\rangle = \sum_{y \in \{0,1\}} \sum_{x \in \{0,1\}} \alpha_{xy} |xy\rangle. \quad (2.47)$$

It is common in literature to write the tensor product of two qubit states $|x\rangle \otimes |y\rangle$ as $|xy\rangle$. A complete, orthonormal basis for the 2-qubit space is then $\{|00\rangle, |01\rangle, |10\rangle, |11\rangle\}$.

We extend this to an arbitrary number of qubits. An n -qubit system will be of dimension 2^n , and we denote the computational basis states as $|z\rangle$, where $z \in \{0,1\}^n$ are the n -bit bitstrings, which can be written $z = z_0 z_1 z_2 \dots z_{2^n-1}$ for $z_k \in \{0,1\}$. An arbitrary state $|\psi\rangle$ on an n -qubit system is thus written

$$|\psi\rangle = \sum_{z \in \{0,1\}^n} c_z |z\rangle \quad (2.48)$$

with coefficients c_z such that $\sum_z |c_z|^2 = 1$. A measurement in the computational basis gives outcome z with probability $|c_z|^2$.

An alternative way to represent n -qubit states is by integers $i = 0, 1, \dots, 2^n - 1$, such that

$$|z\rangle = |z_0 z_1 \dots z_{2^n-1}\rangle = |i\rangle, \quad i = \sum_{k=0}^{2^n-1} z_k \cdot 2^k, \quad (2.49)$$

where z is a binary string $z = z_0 z_1 \dots z_{2^n-1}$, $z_k \in \{0, 1\} \forall k$. This eases notation in some circumstances. We will write multi-qubit states in the bit string representation, unless otherwise specified.

2.5.3 Quantum computations and circuit notation

A *quantum computation*, or *quantum algorithm*, consists of the preparation of a multi-qubit initial state $|\psi\rangle_0$, the evolution of this state by a sequence of unitary operators acting on one or more of the qubits at a time, and projective measurements of some or all of the qubits.

We will write quantum computations in terms of *quantum circuits*. The notation is analogous to that of classical *logical circuits*. Logical circuits model classical computation on the lowest level of abstractions, in terms of wires, representing bits, and logical operators acting on them. Some examples of classical logical operators are AND-gates and OR-gates.

At the leftmost end of a quantum circuit, we write the initial state of each qubit. Each qubit is assigned a wire. We follow the evolution of a qubit state by following its wire from left to right. Thus, the circuit

$$|\psi\rangle \text{ —————} \quad (2.50)$$

represents a single qubit initialized in state $|\psi\rangle$.

The states are evolved by applying *quantum gates*, representing unitary operators acting on one or several qubits, in sequence. These are applied in time in order from left to right. A bare wire represents no evolution, or equivalently an application of the identity operator \mathbb{I} . The quantum circuit

$$\begin{array}{c} |\psi\rangle \text{ — } \boxed{U} \text{ — } \\ |\phi\rangle \text{ ————— } \boxed{V} \text{ — } \end{array}, \quad (2.51)$$

is thus equivalent to acting on the state $|\psi\rangle \otimes |\phi\rangle$ with the operator $V_{12}(U_1 \otimes \mathbb{I}_2)$.

Measurements are represented by a special class of gates, denoted by a meter. Doubled wires represents classical bits. The measurement of a single qubit is represented by the circuit

$$\text{———— } \boxed{\text{meter}} = x, \quad (2.52)$$

where the measurement outcome is stored in the classical bit x . All measurements are performed in the computational basis $\{|0\rangle, |1\rangle\}$, unless otherwise specified.

As an example, combining all the aspects of quantum circuits that we have defined, take the quantum circuit

$$\begin{array}{c}
 |q_0\rangle \text{ --- } \boxed{U} \text{ --- } \boxed{M} \text{ --- } \text{meter} = a \\
 |q_1\rangle \text{ --- } \boxed{U} \text{ --- } \text{---} \\
 |q_2\rangle \text{ --- } \boxed{V} \text{ --- } \boxed{E} \text{ --- } \text{meter} = b.
 \end{array} \tag{2.53}$$

This circuit is equivalent to the unitary evolution of the initial state $|q_0q_1q_2\rangle = |q_0\rangle \otimes |q_1\rangle \otimes |q_2\rangle$, to the state $|\psi\rangle = \mathcal{U} |q_0q_1q_2\rangle$. The operator \mathcal{U} is written as

$$|\psi\rangle = \mathcal{U} |q_0q_1q_2\rangle = (M \otimes \mathbb{I} \otimes E)(U \otimes \mathbb{I})(\mathbb{I} \otimes \mathbb{I} \otimes V) |q_0q_1q_2\rangle. \tag{2.54}$$

Then followed by a measurement of qubits 0 and 2, in the computational basis $\{|0\rangle, |1\rangle\}$, for which the measured values, 0 or 1, are stored in the classical bits a and b .

Note that, whereas a quantum gate normally represents a unitary operator, sometimes it will be convenient to write general quantum channels, acting on the qubit states, in the same quantum circuit notation, for example, in the case of a noise-afflicted gate that is not necessarily unitary anymore. When doing so, this will be clearly specified, and the following analysis will be done within the density operator formalism.

When quantifying the complexity of a quantum algorithm, a common metric is the circuit *depth*. This is defined as the longest path from initialisation to the circuit end, moving in time along the qubit wires. Each gate execution gives a contribution of one. This is roughly equivalent to the number of time steps needed for the circuit execution to terminate. As an example, take the circuit in Equation (2.53). Counting the measurement gates, this circuit has a depth of four.

2.5.4 Quantum Gates

We will now have a closer look at quantum gates, and define some useful examples of quantum gates. A *quantum gate* is a unitary operation, acting on one or several qubits. We write the computational basis states of a qubit system as vectors in \mathbb{C}^2 .

$$|0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad |1\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix}. \tag{2.55}$$

By taking the tensor product of the single-qubit computational basis states, the computational basis state for an n -qubit system can analogously be written as unit vectors in \mathbb{C}^{2^n} . As an example, take an arbitrary state $|\psi\rangle$ on a 2-qubit system. We now write $|\psi\rangle$ as

$$|\psi\rangle = \sum_{z \in \{0,1\}^2} c_z |z\rangle = [c_{00} \ c_{10} \ c_{01} \ c_{11}]^T. \tag{2.56}$$

Any unitary operator on an n -qubit system will now map 2^n -dimensional complex vectors to 2^n -dimensional complex vectors, and can thus be expressed as a $2^n \times 2^n$ matrix.

An example of an often used quantum single-qubit gate is the Hadamard gate, defined as

$$H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} = -\boxed{H}. \quad (2.57)$$

Notably, the Hadamard acts on the single-qubit computational basis states as $H|0\rangle = |+\rangle$ and $H|1\rangle = |-\rangle$. The Hadamard thus gives a basis transformation, from the computational basis $\{|0\rangle, |1\rangle\}$, to the basis $\{|+\rangle, |-\rangle\}$ given by $|\pm\rangle = \frac{1}{\sqrt{2}}(|0\rangle \pm |1\rangle)$.

Furthermore, consider an n -qubit system starting out in the 0-state, i.e., in $|00\dots 0\rangle$. If we apply a single Hadamard gate to each of the n -qubits, we obtain an equal superposition of all the computational basis states, as

$$H^{\otimes n} |00\dots 0\rangle = \frac{1}{\sqrt{2^n}} \sum_{z \in \{0,1\}^n} |z\rangle. \quad (2.58)$$

Here, we have used $H^{\otimes n}$ to denote the tensor product $H \otimes \dots \otimes H$, of a total of n Hadamard gates.

The T -gate, and its inverse, the T^\dagger -gate, are two other useful quantum gates. These are defined as

$$T = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{bmatrix} = -\boxed{T}, \quad T^\dagger = \begin{bmatrix} 1 & 0 \\ 0 & e^{-i\pi/4} \end{bmatrix} = -\boxed{T^\dagger}. \quad (2.59)$$

Furthermore, we define the phase gate S , and its inverse S^\dagger , as

$$S = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\pi/2} \end{bmatrix} = -\boxed{S}, \quad S^\dagger = \begin{bmatrix} 1 & 0 \\ 0 & e^{-i\pi/2} \end{bmatrix} = -\boxed{S^\dagger}, \quad (2.60)$$

and the Pauli gates $X = \sigma_x$, $Z = \sigma_z$, and $Y = \sigma_y = iXZ$, which are defined as

$$X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} = -\boxed{X}, \quad (2.61)$$

$$Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} = -\boxed{Z}, \quad (2.62)$$

$$Y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix} = -\boxed{Y}. \quad (2.63)$$

The Pauli gates further define the *rotation gates*, $R_x(\theta)$, $R_z(\theta)$, and $R_y(\theta)$, all

parametrized by an angle θ , as

$$R_x(\theta) = e^{i\theta X/2} = \begin{bmatrix} \cos(\frac{\theta}{2}) & -i \sin(\frac{\theta}{2}) \\ -i \sin(\frac{\theta}{2}) & \cos(\frac{\theta}{2}) \end{bmatrix} = -\boxed{R_x(\theta)}, \quad (2.64)$$

$$R_z(\theta) = e^{i\theta Z/2} = \begin{bmatrix} e^{-i\frac{\theta}{2}} & 0 \\ 0 & e^{i\frac{\theta}{2}} \end{bmatrix} = -\boxed{R_z(\theta)}, \quad (2.65)$$

$$R_y(\theta) = e^{i\theta Y/2} = \begin{bmatrix} \cos(\frac{\theta}{2}) & -\sin(\frac{\theta}{2}) \\ \sin(\frac{\theta}{2}) & \cos(\frac{\theta}{2}) \end{bmatrix} = -\boxed{R_y(\theta)}. \quad (2.66)$$

Representing an arbitrary single-qubit state in the Bloch sphere representation, given in Equation (2.26), these gates have the following interpretation: The gates $R_x(\theta)$, $R_z(\theta)$, and $R_y(\theta)$ act on $\rho = \frac{1}{2}(\mathbb{I} + \vec{r} \cdot \vec{\sigma})$ by rotating the Bloch vector by θ degrees around the x , z , or y axis respectively.

In fact, any unitary mapping between single-qubit pure quantum states can be decomposed into a sequence of the three rotation operators, possibly multiplied by a global phase factor. This follows from the rotation operators giving three independent rotations of a three-dimensional unit vector, i.e., the Bloch vector. Thus, any arbitrary unitary operator U can be written as

$$U = e^{i\alpha} R_x(\theta) R_z(\phi) R_y(\lambda). \quad (2.67)$$

An important set of single-qubit gates, sometimes referred to as the *physical gates*, are the U_1 , U_2 , and U_3 gates. These are the physically realized single-qubit gates in the quantum computers publicly available, at present time, through the IBM Quantum Experience [21]. The $U_3(\theta, \phi, \lambda)$ gate, parameterized by the angles θ , ϕ , and λ , is given as

$$U_3(\theta, \phi, \lambda) = \begin{bmatrix} \cos(\theta/2) & -e^{i\lambda} \sin(\theta/2) \\ e^{i\phi} \sin(\theta/2) & e^{i\lambda+i\phi} \cos(\theta/2) \end{bmatrix} = -\boxed{U_3(\theta, \phi, \lambda)}. \quad (2.68)$$

With three independent angles, the U_3 gate can generate any arbitrary rotation of the Bloch vector, and thus any arbitrary single-qubit unitary. The U_1 and U_2 gates are derived from the U_3 gate, as $U_1(\lambda) = U_3(0, 0, \lambda)$ and $U_2(\phi, \lambda) = U_3(\frac{\pi}{2}, \phi, \lambda)$.

2.5.5 Controlled gates

An important class of multi-qubit gates are the *controlled* gates. Focusing on the two-qubit case, a controlled gate acts on one qubit called the *control*, and one qubit called the *target*. In general, a controlled gate may act on multiple control and target qubits. In circuit notation, a two-qubit *controlled- U* gate, denoted cU , is defined as

$$cU = \begin{array}{c} \text{---} \bullet \text{---} \\ | \\ \boxed{U} \end{array}, \quad (2.69)$$

for U being a unitary operator. The black dot denotes the control, and the other qubit is the target. The action of a controlled-U gate on a state $|x\rangle|y\rangle$, where $|x\rangle$ is the control and $|y\rangle$ is the target, is written as a quantum operator as

$$cU = |0\rangle\langle 0| \otimes \mathbb{I} + |1\rangle\langle 1| \otimes U. \quad (2.70)$$

A controlled gate of high practical importance is the *controlled-NOT* gate, or *controlled-X* gate, denoted CNOT or cX . The CNOT-gate is in circuit notation denoted as

$$\text{CNOT} = \begin{array}{c} \bullet \\ | \\ \oplus \end{array}. \quad (2.71)$$

Note the notation on the lower qubit, the target qubit. Writing out the quantum operator, and representing the two-qubit state as a vector in \mathbb{C}^4 , as given in Equation (2.56), the CNOT-gate can be expressed as a 4×4 matrix as

$$\text{CNOT} = |0\rangle\langle 0| \otimes \mathbb{I} + |1\rangle\langle 1| \otimes X = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}. \quad (2.72)$$

The reason for the *controlled-NOT* term becomes apparent when inspecting the action of the CNOT on computational basis states $|x\rangle, |y\rangle \in \{|0\rangle, |1\rangle\}$. We get

$$\text{CNOT} |x\rangle |y\rangle = |x\rangle |y \oplus x\rangle, \quad (2.73)$$

where \oplus is addition modulo 2. For $|x\rangle, |y\rangle$ computational basis states, the CNOT acts as a logical NOT gate on y if $x = 1$, and as the identity gate otherwise.

2.5.6 Gate identities

The Pauli-gates interact with the CNOT-gates to give effective, total operators according to

$$\text{CNOT} (X \otimes \mathbb{I}) \text{CNOT} = X \otimes X, \quad \text{CNOT} (\mathbb{I} \otimes X) \text{CNOT} = \mathbb{I} \otimes X, \quad (2.74)$$

and

$$\text{CNOT} (Z \otimes \mathbb{I}) \text{CNOT} = Z \otimes \mathbb{I}, \quad \text{CNOT} (\mathbb{I} \otimes Z) \text{CNOT} = Z \otimes Z, \quad (2.75)$$

where the operators act on 2-qubit states where the first qubit is the control, and the second is the target. In terms of quantum circuits, the Pauli- X and Z gates are said to be "propagated" over a CNOT-gate, according to the following identities

$$\begin{array}{c} \boxed{X} \text{---} \bullet \\ | \\ \oplus \end{array} = \begin{array}{c} \bullet \text{---} \boxed{X} \\ | \\ \oplus \end{array} \boxed{X}, \quad (2.76)$$

$$\begin{array}{c}
 \text{---} \bullet \text{---} \\
 | \\
 \text{---} \boxed{X} \oplus \text{---}
 \end{array}
 =
 \begin{array}{c}
 \text{---} \bullet \text{---} \\
 | \\
 \text{---} \oplus \boxed{X} \text{---}
 \end{array},
 \quad (2.77)$$

$$\begin{array}{c}
 \text{---} \boxed{Z} \text{---} \bullet \text{---} \\
 | \\
 \text{---} \oplus \text{---}
 \end{array}
 =
 \begin{array}{c}
 \text{---} \bullet \boxed{Z} \text{---} \\
 | \\
 \text{---} \oplus \text{---}
 \end{array},
 \quad (2.78)$$

$$\begin{array}{c}
 \text{---} \bullet \text{---} \\
 | \\
 \text{---} \boxed{Z} \oplus \text{---}
 \end{array}
 =
 \begin{array}{c}
 \text{---} \bullet \boxed{Z} \text{---} \\
 | \\
 \text{---} \oplus \boxed{Z} \text{---}
 \end{array}.
 \quad (2.79)$$

In the context of quantum errors, this means that a single Pauli- X error on the control qubit will be propagated into a *double* Pauli- X error, one on both qubits, by a CNOT-gates. A single Pauli- X error on the target is left alone. And similarly, a single Pauli- Z error on the target propagates into two errors, whereas a Pauli- Z error on the control stays as a single error. This has implications for the construction of *fault-tolerant* quantum circuit.

The Pauli- Y gate identities can be found from the above, by using $Y = iXZ$. The factor i becomes a global phase factor that can be ignored.

A CNOT gate with opposite control and target qubits can be constructed by applying single-qubit Hadamard gates, according to

$$\begin{array}{c}
 \text{---} \boxed{H} \text{---} \bullet \text{---} \boxed{H} \text{---} \\
 | \\
 \text{---} \boxed{H} \oplus \text{---} \boxed{H} \text{---}
 \end{array}
 =
 \begin{array}{c}
 \text{---} \oplus \text{---} \\
 | \\
 \text{---} \bullet \text{---}
 \end{array}.
 \quad (2.80)$$

Furthermore, the other controlled-Pauli gates, cZ and cY , can be constructed from the $cX = \text{CNOT}$ gate and single-qubit gates, as

$$\begin{array}{c}
 \text{---} \bullet \text{---} \\
 | \\
 \text{---} \boxed{H} \oplus \boxed{H} \text{---}
 \end{array}
 =
 \begin{array}{c}
 \text{---} \bullet \text{---} \\
 | \\
 \text{---} \boxed{Z} \text{---}
 \end{array},
 \quad (2.81)$$

$$\begin{array}{c}
 \text{---} \bullet \text{---} \\
 | \\
 \text{---} \boxed{S^\dagger} \oplus \boxed{S} \text{---}
 \end{array}
 =
 \begin{array}{c}
 \text{---} \bullet \text{---} \\
 | \\
 \text{---} \boxed{Y} \text{---}
 \end{array}.
 \quad (2.82)$$

In general, any controlled- U gate, for a single-qubit unitary operator U , can be constructed with CNOT-gates and single-qubit rotations. To do so, one has to find three rotations, A , B , and C , such that

$$ABC = \mathbb{I}, \quad AZBZC = e^{i\alpha}U, \quad (2.83)$$

for $e^{i\alpha}$ some global phase that can be ignored. The controlled- U gate is then constructed as

$$\begin{array}{c}
 \text{---} \bullet \text{---} \bullet \text{---} \boxed{U_1(\alpha)} \text{---} \\
 | \quad | \\
 \text{---} \boxed{A} \boxed{Z} \boxed{B} \boxed{Z} \boxed{C} \text{---}
 \end{array}
 =
 \begin{array}{c}
 \text{---} \bullet \text{---} \\
 | \\
 \text{---} \boxed{C} \text{---}
 \end{array}.
 \quad (2.84)$$

$$\text{SWAP} |\psi\rangle |\phi\rangle = |\phi\rangle |\psi\rangle. \quad (2.85)$$
$$\begin{array}{|c|} \hline \times \\ \hline \\ \hline \times \\ \hline \end{array} = \begin{array}{|c|c|c|} \hline \bullet & \bigcirc & \bullet \\ \hline \bigcirc & \bullet & \bigcirc \\ \hline \end{array} . \quad (2.86)$$
$$\begin{array}{ccc}
 |x\rangle & \text{---} \bullet & |x\rangle \\
 |y\rangle & \text{---} \bullet & |y\rangle \\
 |z\rangle & \text{---} \oplus & |z \oplus (x \cdot y)\rangle .
 \end{array} \quad (2.87)$$

(2.88)

A native gate set that is universal, and that can not be efficiently simulated in any known way using a classical computer, is $\{\text{CNOT}, H, T\}$. Since this set

is universal, it follows that $\{\text{CNOT}, U_3\}$ is also universal for quantum computation. Here, U_3 is the general single-qubit gate, as defined in Equation (2.68).

But, what about initialization and measurements? Assume that the native gate set of a quantum device is universal and that we want to execute a quantum computation starting from initial state $|\psi_0\rangle$. As the gate set is universal, we can construct by a sequence of gates from the native gate set the unitary operator U such that $|\psi_0\rangle = U|00\dots 0\rangle$. Thus, initialisation to the 0-state, or any other single, initial state, is universal for quantum computation. Similarly, for any measurement in an arbitrary basis $\{|\xi_i\rangle\}_i$, we can construct the basis-change operator from the basis states $|\xi_i\rangle$ to the computational basis states $|z\rangle$. Therefore, measurements in the computational basis is universal for quantum computation.

We focus now on the native gate set of $\{\text{CNOT}, U_3\}$. Assume that we have a 3-qubit quantum device, with qubits denoted by 0, 1, and 2. Do we need to have CNOT-gates physically realized between all qubits? Assume that we have physically realized CNOT-gates between qubits 0 and 1, and between 1 and 2. A CNOT-gate between the states on qubit 0 and qubit 2 can then be performed by first applying a SWAP-gate between qubits 1 and 2, which by Equation (2.86) can be constructed from CNOT-gates. Then, performing the CNOT-gate between qubit 0 and qubit 1, the latter on which the state originally on qubit 2 is now located. Thus, it is universal for quantum computing with a gate set consisting of single-qubit gates, and a network of CNOT-gates such that there exists a path of CNOT-gates between any two qubits, i.e., this network need not be fully connected.

2.5.8 Measurements of Quantum Circuits

A measurement in a quantum circuit is treated as a special type of gate, as denoted in Equation (2.52). Measurements in the computational basis is sufficient for quantum computation, as one can construct measurements in any arbitrary single-qubit basis $\{|\xi_0\rangle, |\xi_1\rangle\}$ by applying a basis change operator U , such that $U|\xi_0\rangle = |0\rangle$ and $U|\xi_1\rangle = |1\rangle$, and then measuring in the computational basis.

As $H|+\rangle = |0\rangle$ and $H|-\rangle = |1\rangle$, for $|\pm\rangle = \frac{1}{\sqrt{2}}(|0\rangle \pm |1\rangle)$, which forms an orthonormal basis for the qubit state space, one can construct a measurement in the $\{|\pm\rangle\}$ basis as

$$\text{---} \boxed{H} \text{---} \boxed{\text{meter}} = x = \text{---} \boxed{\text{meter}}_{|\pm\rangle} = x, \quad (2.89)$$

where the meter on the right hand side denotes a measurement in the $\{|\pm\rangle\}$ basis. A measurement outcome of $x = 0$ on the classical bit is to be understood as the measurement of the $|+\rangle$ state, and $x = 1$ as the $|-\rangle$ state.

A non-destructive measurement of a qubit can be done by use of ancillary

qubits. Take the circuit

$$\begin{array}{ccc}
 |\psi\rangle & \text{---} & \bullet & \text{---} & |\psi'\rangle \\
 & & | & & \\
 |0\rangle & \text{---} & \oplus & \text{---} & \boxed{\text{Measurement}} = x.
 \end{array} \quad (2.90)$$

For a general qubit state $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ on the input, the full state of the two-qubit system after the CNOT gate will be

$$\text{CNOT } |\psi\rangle |0\rangle = \text{CNOT } (\alpha|0\rangle + \beta|1\rangle) |0\rangle = \alpha|0\rangle|0\rangle + \beta|1\rangle|1\rangle. \quad (2.91)$$

A measurement of the ancillary qubit in the computational basis will thus yield an outcome of $x = 0$ with probability $|\alpha|^2$, and $x = 1$ with probability $|\beta|^2$. The measurement is still projective, such that the upper qubit is left in a post-measurement state depending on the outcome, which in this case will be $|\psi'\rangle = |x\rangle$.

2.5.9 Expectation values

Expectation values of operators are measured by combining simultaneous single-qubit measurement, on one or several qubits, with classical post-processing. Assume that we want to measure the expectation value of an operator A with respect to some state $|\psi\rangle = \mathcal{U}|00\dots 0\rangle$. The unitary operator \mathcal{U} is implemented as a quantum circuit. The state $|\psi\rangle$ can be expressed as

$$|\psi\rangle = \sum_i c_i |\phi_i\rangle \quad (2.92)$$

for $\{|\phi_i\rangle\}_i$ a set of orthonormal eigenstates of A , such that $A|\phi_i\rangle = a_i|\phi_i\rangle$ for a $a_i \in \mathbb{R}$.

Measurements in other bases than the computational basis can be performed if we find a basis transformation by an unitary operator U such that

$$U\phi_i = |z_i\rangle, \quad (2.93)$$

where z_i are the unique bit-string representations of the integers $i = 0, \dots, 2^n - 1$. When measuring in the computational basis, after the basis change, the measurement outcomes will be the bit string z_i with probability $|c_i|^2$.

To estimate the expectation value of A with respect to $|\psi\rangle$, as each outcome is probabilistic, we repeat the experiment of state preparation and measurement several times. Then, from the set of measurement outcomes, we compute the average observed value.

As an example, assume we prepare the two-qubit state

$$|\psi\rangle = a_{00}|0\rangle \otimes |0\rangle + a_{01}|0\rangle \otimes |1\rangle + a_{10}|1\rangle \otimes |0\rangle + a_{11}|1\rangle \otimes |1\rangle, \quad (2.94)$$

by a quantum circuit. Say we want to measure its expectation value with respect to the operator $Z \otimes Z$. The eigenvalues and eigenstates are

$$Z|0\rangle = +1|0\rangle, \quad Z|1\rangle = -1|1\rangle. \quad (2.95)$$

We get the eigenvalue $+1$ for eigenstates $|0\rangle \otimes |0\rangle$ and $|1\rangle \otimes |1\rangle$, and -1 for the eigenstates $|0\rangle \otimes |1\rangle$ and $|1\rangle \otimes |0\rangle$. The expectation value $E^* = \langle Z \rangle_{|\psi\rangle}$ of a measurement of this operator, with respect to the state $|\psi\rangle$, is thus

$$\begin{aligned} E^* &= (+1)(p_{00} + p_{11}) + (-1)(p_{01} + p_{10}) \\ &= (+1)(|a_{00}|^2 + |a_{11}|^2) + (-1)(|a_{01}|^2 + |a_{10}|^2). \end{aligned} \quad (2.96)$$

Here, $p_{00} = |a_{00}|^2$ is the probability of measuring the outcome 00, and analogously for the other eigenstates.

Let $n_{z_0 z_1}$ be the number of times the measurement outcome $z = z_0 z_1 \in \{0, 1\}^2$, corresponding to the state $|z_0\rangle \otimes |z_1\rangle$ for $z_0, z_1 \in \{0, 1\}$, and $N = \sum_{z_0, z_1} n_{z_0 z_1}$ be the total number of experiments. Then, $n_{z_0 z_1}/N$ serves as an estimate for $p_{z_0 z_1} = |a_{z_0 z_1}|^2$. Furthermore, the expectation value E^* is then estimated by

$$E = \frac{1}{N}((n_{00} + n_{11}) - (n_{01} + n_{10})). \quad (2.97)$$

2.5.10 Error estimation

Assume that N shots of a circuit, measuring some observable, are performed, each time with a measurement outcome of μ_j , $j = 1, \dots, N$. Let E^* be the expectation value and σ^2 the variance of the measurement outcomes, which are assumed to be independently and equally distributed. An estimator for the expectation value is then

$$E = \frac{1}{N} \sum_j \mu_j, \quad (2.98)$$

and an estimator for the variance σ^2 in each measurement is the *mean square error* s^2 , given as

$$s^2 = \frac{1}{N-1} \sum_j (\mu_j - E)^2 = \frac{1}{N} \sum_j \mu_j^2 - \left(\frac{1}{N} \sum_j \mu_j\right)^2. \quad (2.99)$$

By the central limit theorem, an estimator for the error ϵ in E is

$$\epsilon = \frac{\sigma}{\sqrt{N}}, \quad (2.100)$$

where E is the average over N measurement outcomes, i.e., with the circuit executed for N shots.

The variance in a quantum measurement is in general hard to know without knowing the exact state $|\psi\rangle$ of the system at the time of measurements. Therefore, we do not in general know how many shots is necessary to estimate E^* to within some tolerance error ϵ_{tol} .

A solution to this problem is thus to *sample* the circuit for some initial number of shots N , obtaining measurement outcomes μ_j . Then, we estimate the variance σ^2 by the mean-square error s^2 , as shown in Equation (2.100).

The required amount of shots N' to achieve an error below the threshold is then estimated to be

$$N' = \left(\frac{s}{\epsilon_{\text{tol}}} \right)^2. \quad (2.101)$$

If $N' > N$, we repeat the experiment for an additional number $N' - N$ of shots. The average over all measurement outcomes now gives an estimator of the expectation value, with an estimated error $\epsilon \leq \epsilon_{\text{tol}}$.

Chapter 3

Error Models for Quantum Computation

Near-term NISQ-era devices are noisy. A main concern is thus to model and combat the noise present in these devices. In this chapter, we present noise in quantum computers in terms of quantum channels, and we give a few examples of particular useful noise channels.

Based on the quantum channel formulation of noise in quantum computers, we give a brief introduction to the concept of quantum error correction by error correcting codes. We define the 3-qubit quantum repetition code, which is analogue to the classical 3-bit repetition code, as an example. Using this example, we explore the reasons why quantum error correction is useful, by allowing correction to arbitrary precision, given an error rate below a certain error threshold, and why it might not be feasible in the short-term, because of the scaling of the ancillary qubit requirements.

3.1 Noise channels

Any ideal quantum gate that applies a unitary operator U , can be described by a quantum channel $\mathcal{N}(\rho) = U\rho U^\dagger$. When a gate is *noisy*, its action on any density operator ρ is still in general given by a quantum channel \mathcal{E} , but this noisy quantum channel may potentially no longer preserve purity of density operators.

Furthermore, say we want to apply a quantum channel \mathcal{N} corresponding to a unitary gate, but due to noise, the channel that is applied is instead \mathcal{E} . By writing

$$\mathcal{E} = \mathcal{E} \circ \mathcal{N}^{-1} \circ \mathcal{N} = \tilde{\mathcal{E}} \circ \mathcal{N}, \quad (3.1)$$

where $\tilde{\mathcal{E}} \circ \mathcal{N}(\rho) = \tilde{\mathcal{E}}(\mathcal{N}(\rho))$, we can regard a faulty quantum channel or gate afflicted by noise as the ideal channel \mathcal{N} followed by the effective noise channel $\tilde{\mathcal{E}}$ on the qubits between gates.

Similarly, we can describe a noisy initialization of qubits to the faulty initial state ρ'_0 as an ideal initialization to the state ρ_0 followed by an error channel such that $\mathcal{N}_{init}(\rho_0) = \rho'_0$. Furthermore, noise in measurements can be described by an ideal measurement preceded by a noise channel.

3.2 Coherent noise

Coherent noise is noise that can be expressed as a unitary error operator \tilde{U} . The quantum operation acting on density operators then becomes a unitary operation $\tilde{\mathcal{E}}(\rho) = \tilde{U}\rho\tilde{U}^\dagger$. By Equation (2.35), such noise preserves the purity of a pure input state $\rho = |\psi\rangle\langle\psi|$.

An example is an error in a quantum single-qubit gate that causes a perturbation in the rotation of the Bloch sphere. Any single qubit gate can be expressed as a rotation of the Bloch vector as in Equation (2.67). Say one wants to perform a rotation about the z -axis by an angle ϕ , i.e., $U = R_z(\phi)$, but due to an error the applied operator is instead $U' = R_z(\phi + \delta)$ for some small angle δ . This can be described by the ideal rotation followed by the perturbation $R_z(\delta)$ as $U' = R_z(\delta)R_z(\phi) = R_z(\delta)U$. The corresponding quantum channel becomes $\tilde{\mathcal{E}}(\rho) = U'\rho U'^\dagger$, preserving purity.

3.3 Incoherent noise

Incoherent noise is noise that does not necessarily preserve purity, for example noise stemming from an interaction with the environment. As described by the Stinespring dilation and Equation (2.36), a quantum operation unitary on a larger system, for example the qubit in interaction with the environment, can be described locally on the qubit as a quantum channel.

3.3.1 Stochastic Pauli-noise

A simple example of an incoherent noise channel is the stochastic Pauli channel. Consider the Pauli operators X , Z , and Y . Say that each of them are independently, probabilistically applied to a single qubit with probabilities p_x , p_z , and p_y . This is described by the Kraus operators

$$E_0 = \sqrt{1 - p_x - p_y - p_z} \mathbb{I}, \quad E_1 = \sqrt{p_x} X, \quad E_2 = \sqrt{p_y} Y, \quad E_3 = \sqrt{p_z} Z. \quad (3.2)$$

Then, the condition of Equation (2.38) is fulfilled by these Kraus operators, as

$$\sum_i E_i^\dagger E_i = (1 - p_x - p_y - p_z) \mathbb{I} + p_x X X + p_y Y Y + p_z Z Z = \mathbb{I}, \quad (3.3)$$

where we use that the Pauli gates are their own adjoints, and their own inverses.

3.3.2 Amplitude damping

Let us assume that we have a qubit system in which the $|0\rangle$ state is encoded as the ground-state of the system, and $|1\rangle$ as an excited state. A qubit in the $|1\rangle$ state may then fall to the $|0\rangle$ state by the process of spontaneous emission, but not the other way around.

This is described by the *amplitude damping* channel. Assume that the excited $|1\rangle$ state may collapse down to a $|0\rangle$ state with probability p . This case is described by the Kraus operators

$$E_0 = \begin{bmatrix} 1 & 0 \\ 0 & \sqrt{1-p} \end{bmatrix}, \quad E_1 = \begin{bmatrix} 0 & \sqrt{p} \\ 0 & 0 \end{bmatrix}. \quad (3.4)$$

These Kraus operators give exactly the described case, where a $|0\rangle$ -state is kept in the $|0\rangle$ state as

$$\tilde{\mathcal{E}}(|0\rangle\langle 0|) = E_0 |0\rangle\langle 0| E_0^\dagger + E_1 |0\rangle\langle 0| E_1^\dagger = |0\rangle\langle 0|, \quad (3.5)$$

whereas a $|1\rangle$ -state is transformed to a probabilistic mixture of the state collapsing into the $|0\rangle$ -state with probability p , or not collapse with probability $1-p$, as

$$\tilde{\mathcal{E}}(|1\rangle\langle 1|) = E_0 |1\rangle\langle 1| E_0^\dagger + E_1 |1\rangle\langle 1| E_1^\dagger = p|0\rangle\langle 0| + (1-p)|1\rangle\langle 1|. \quad (3.6)$$

The action on a general qubit state, $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$, further follows from linearity.

3.3.3 Phase Damping

A uniquely quantum-mechanical noise process is the process of *phase damping*, describing loss of information in the partial phases of the eigenstates. Take the case in which the qubit state on system A does not transition between states but couples to the environment in the following way,

$$\tilde{\mathcal{E}}(|0\rangle_A \otimes |0\rangle_E) = \sqrt{1-p}|0\rangle_A \otimes |0\rangle_E + \sqrt{p}|0\rangle_A \otimes |1\rangle_E, \quad (3.7)$$

and

$$\tilde{\mathcal{E}}(|1\rangle_A \otimes |0\rangle_E) = \sqrt{1-p}|1\rangle_A \otimes |0\rangle_E + \sqrt{p}|1\rangle_A \otimes |2\rangle_E, \quad (3.8)$$

where $\{|i\rangle_E\}_i$ form some basis for the environment. An interpretation could be that the environment in some way would scatter off of the qubit with probability p , ending up in a state dependent on the qubit state.

By tracing out the environment, we obtain a quantum channel with Kraus operators

$$E_0 = \begin{bmatrix} \sqrt{1-p} & 0 \\ 0 & \sqrt{1-p} \end{bmatrix}, \quad E_1 = \begin{bmatrix} \sqrt{p} & 0 \\ 0 & 0 \end{bmatrix}, \quad E_2 = \begin{bmatrix} 0 & 0 \\ 0 & \sqrt{p} \end{bmatrix}. \quad (3.9)$$

The effect of this channel on a general density operator

$$\rho = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \quad (3.10)$$

becomes

$$\tilde{\mathcal{E}}(\rho) = \begin{bmatrix} a & (1-p)b \\ (1-p)c & d \end{bmatrix}. \quad (3.11)$$

Thus, for repeated applications of this channel, the off-diagonal terms b and c , the relative phases, are suppressed.

3.4 Quantum Error Correction

Quantum error correction using quantum error-correcting codes provides means to detect and exactly correct errors during a quantum computation by making use of *redundancy*. By encoding virtual, or *logical*, qubits into states on multiple physical qubits, errors that move the states out of the assigned code subspace can be detected by doing measurements. Based on the measurement outcomes, recovery operations are then done to recover the original logical qubit state.

3.4.1 The Classical Repetition code

To illustrate the concept, first consider the classical 3-bit repetition code with syndrome decoding. In this code, a bit $x \in \{0, 1\}$ is encoded into an encoded *logical* bit according to

$$0 \rightarrow 000, \quad 1 \rightarrow 111. \quad (3.12)$$

Say now that a bit flip error occurs with probability p , flipping a 0 into a 1 or vice versa. The encoded bit $\tilde{x} \in \{000, 111\}$ can be recovered and decoded even if a single bit-flip error has happened on any of the three bits.

This can be done as follows. For an encoded bit $\tilde{x} = x_0x_1x_2$, assign the two *syndromes* s_0 and s_1 as

$$s_0 = x_0 \oplus x_1, \quad s_1 = x_1 \oplus x_2, \quad (3.13)$$

where \oplus is addition modulo 2. The decoding process proceeds as follows; check the two syndromes s_0 and s_1 . Assume that at most a single bit-flip error has occurred. If $s_0 = 1$, then $x_0 \neq x_1$ and a bit-flip error has occurred on either x_0 or x_1 . If $s_1 = 1$, then $x_1 \neq x_2$ and a bit-flip error has occurred on either x_1 or x_2 .

If $s_0 = 1$ and $s_1 = 0$, flip x_0 . If $s_0 = 1$ and $s_1 = 1$, flip x_1 . If $s_0 = 0$ and $s_1 = 1$, flip x_2 . The encoded bit is now either a string of three zeros or three ones, coinciding with either the encoded 0-bit, or the encoded 1-bit.

This procedure ensures that a single bit-flip error on one of the *physical* bits does not result in an error in the *logical* bit. If bit-flip errors may occur independently with probability p on any bit, the 3-bit repetition code lowers the probability of a *logical* error to $O(p^2)$, as two or more bit-flip errors must happen for the logical bit to retain an error.

3.4.2 The Quantum Repetition code

Now, we want to construct an analogous error-correction code for use on quantum bits. A first, naive idea would be to simply repeat the qubit state in question, $|\psi\rangle$ three times, giving

$$|\psi\rangle \rightarrow |\psi\rangle |\psi\rangle |\psi\rangle. \quad (3.14)$$

This is, however, not allowed by the *no-cloning* theorem. The no-cloning theorem states that no quantum operator C , such that

$$C(|\psi\rangle |\phi\rangle) = |\psi\rangle |\phi\rangle, \quad (3.15)$$

can exist. The theorem can be derived from the unitary nature of quantum operators, or as a consequence of *no-signalling*, which states that information cannot be transferred faster than the speed of light.

Thus, we need a different scheme. We first consider the analogue to a classical bit-flip error, which is a Pauli- X error. Consider the noise channel \mathcal{E} on a single qubit, defined by Kraus operators

$$E_0 = \sqrt{1-p} \mathbb{I}, \quad E_1 = \sqrt{p} X. \quad (3.16)$$

Extend the noise model to several qubits as $\mathcal{E}_3 = \mathcal{E} \otimes \mathcal{E} \otimes \mathcal{E}$.

A better proposal for a quantum repetition code is the following; encode the computational basis states $\{|0\rangle, |1\rangle\}$ as

$$|0\rangle \rightarrow |000\rangle, \quad |1\rangle \rightarrow |111\rangle. \quad (3.17)$$

The states $|\bar{0}\rangle \equiv |000\rangle$ and $|\bar{1}\rangle \equiv |111\rangle$ are the *code words* of this code. In the same way that $\{|0\rangle, |1\rangle\}$ form a basis for the state space of a single qubit, $\{|000\rangle, |111\rangle\}$ form a basis of the state space of the encoded *logical* qubit. This space is called the *code space*. The encoding of a general qubit state then becomes

$$|\psi\rangle = \alpha |0\rangle + \beta |1\rangle \rightarrow |\bar{\psi}\rangle = \alpha |\bar{0}\rangle + \beta |\bar{1}\rangle = \alpha |000\rangle + \beta |111\rangle. \quad (3.18)$$

This encoding of a general qubit state $|\psi\rangle$ can be performed by the quantum circuit sub-procedure

$$\begin{array}{c} |\psi\rangle \text{---} \bullet \text{---} \bullet \text{---} \\ |0\rangle \text{---} \oplus \text{---} \text{---} \\ |0\rangle \text{---} \oplus \text{---} \end{array} . \quad (3.19)$$

Analogous to the syndromes s_0 and s_1 of the classical repetition code, the syndrome operators, or *stabiliser operators*, of the quantum repetition code are defined as

$$Z_0 Z_1, \quad Z_1 Z_2, \quad (3.20)$$

where Z_i is the Pauli- Z operator acting on qubit i , for $i = 0, 1, 2$. The eigenvalues of these operators are ± 1 . Notice how the general encoded state $|\bar{\psi}\rangle$ is a $+1$ eigenstate of both stabiliser operators. This is by design.

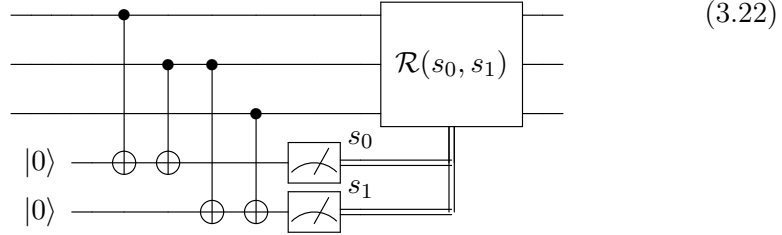
The syndromes s_0 and s_1 are obtained by measuring the stabiliser operators Z_0Z_1 and Z_2Z_3 , respectively. If an X -error have occurred, on say qubit 1, the error-afflicted encoded state becomes a -1 eigenstate of the Z_1 operator, as

$$Z_0Z_1X_1|\bar{\psi}\rangle = -X_1Z_0Z_1|\bar{\psi}\rangle = -X_1|\bar{\psi}\rangle. \quad (3.21)$$

The faulty state $X_1|\bar{\psi}\rangle$ is thus a -1 eigenstate of both stabiliser operators. Here, we use that X_i anticommutes with Z_i , i.e., $X_iZ_i = -Z_iX_i$. X_i , however, commutes with any Z_j for $i \neq j$.

After obtaining the measurement outcomes, stored in s_0 and s_1 , the recovery proceeds in the same way as for the classical repetition code. Now, $s_0 = -1$ indicates an X error on physical qubit 0 or 1, $s_1 = -1$ indicates an X error on physical qubit 1 or 2, and so on. From this example, the measurements return -1 for both syndromes, indicating an X error on qubit 1

The recovery operation $\mathcal{R}(s_0, s_1)$ is done by applying an X -gate to the physical qubit on which the error has been detected. The non-destructive measurements of the stabilizers, followed by recovery operation, is done by the circuit sub-procedure as follows



The single-qubit X -errors, $\{X_i\}_{i=1,2,3}$, is the set of *correctable errors* for the 3-qubit quantum repetition code. These are the errors this code can exactly detect and correct.

In general, a *stabilizer code* is a quantum error-correcting code where the logical qubit states are encoded in the simultaneous +1 eigenstate of some set of stabilizer operators. Errors are detected by doing measurements of the stabilizer operators. In general, the set of correctable errors are the error operators which anticommutes with one or more of the stabilizer operators.

3.4.3 Digitization of errors

Consider an error-channel Kraus operator

$$E = \alpha\mathbb{I} + \beta X + \gamma Z + \delta XZ, \quad (3.23)$$

where $\alpha, \beta, \gamma, \delta$ are complex coefficients.

Assume that we have a quantum error-correcting code in which Z is a stabilizer operator, and with an encoded state $|\bar{\psi}\rangle$. Then, $Z|\bar{\psi}\rangle = |\bar{\psi}\rangle$. By using the anticommutation relation $ZX = -XZ$, we get

$$ZE|\bar{\psi}\rangle = -1(\beta X + \delta XZ)|\bar{\psi}\rangle + (+1)(\alpha\mathbb{I} + \gamma Z)|\bar{\psi}\rangle. \quad (3.24)$$

A measurement outcome of $+1$ thus leaves the logical qubit in the post-measurement state $(\alpha\mathbb{I} + \gamma Z)|\bar{\psi}\rangle$, which is a $+1$ eigenstate of Z , where no X error have occurred. An outcome of -1 leaves the post-measurement state $(\beta X + \delta XZ)|\bar{\psi}\rangle$, where an X error definitely has occurred.

In this way, the measurement of the stabiliser operator has collapsed the state, from some arbitrary superposition of error-free and error-afflicted states, into a state that can be said to definitely be either error-free or error-afflicted. This is with respect to the errors that are correctable by the given stabiliser. We say that the error has been *digitized*.

By concatenating the 3-qubit repetition code with the 3-qubit phase code, with stabilizers X_0X_1 and X_1X_2 , and code words $|\tilde{0}\rangle \equiv |+++\rangle$ and $|\tilde{1}\rangle \equiv |--\rangle$, we get the 9-qubit *Shor code*. The correctable errors for the Shor code are thus any X , Z , and joint XZ errors occurring on only a single qubit at a time. As all single-qubit Kraus operators can be written in the form of Equation (3.23), the Shor code can correct any arbitrary single one-qubit error [6].

3.4.4 Error Thresholds

Consider the quantum repetition code and an error model with Pauli- X error operators occurring on the qubits independently, on each with a probability p . The quantum repetition code allows any case in which at most one X -error occurs to be corrected. However, two or more X -errors happening at the same time will result in an error on the logical qubit, i.e., a *logical error*. The probability of a logical error p_{logical} , given a *physical* error probability p on the physical qubits, thus becomes

$$p_{\text{logical}} = 3p^2(1-p) + p^3 = 3p^2 + \mathcal{O}(p^3). \quad (3.25)$$

Assume the physical error rate p is such that $p = p^*$ for some p^* fulfilling

$$p_{\text{logical}} = p^* \Rightarrow 3(p^*)^2 + \mathcal{O}((p^*)^3) = p^* \Rightarrow p^* = \frac{1}{3}. \quad (3.26)$$

Thus $p^* = \frac{1}{3}$ is the error threshold for the 3-qubit quantum repetition code, up to order $\mathcal{O}(p^3)$. If $p \leq p^*$, the encoding leaves the logical qubit with a logical error rate that is improved compared to the physical error rate.

If the physical error rate is below the threshold value, the logical error rate can be lowered to arbitrary precision by *concatenation*. One concatenation of the 3-qubit repetition code gives the 9-qubit code

$$|\tilde{\psi}\rangle = \alpha|\bar{0}\bar{0}\bar{0}\rangle + \beta|\bar{1}\bar{1}\bar{1}\rangle = \alpha|000000000\rangle + \beta|111111111\rangle, \quad (3.27)$$

with logical error rate $p_{l=2} = (p_{l=1})^2 = 9p^4 + \mathcal{O}(p^5)$, where p_l is the logical error rate of a level l concatenated code. This can be repeated.

Note that the number of physical qubits needed grows as 3^l , i.e., exponentially. The saving grace is that the logical error rate $p_l = (p_{l-1})^2 = p^{2^l}$ drops super-exponentially with l . The total qubit requirement can thus be shown to increase polylogarithmically as a function of the inverse, target precision ϵ^{-1} , i.e., as $\text{polylog}(\epsilon^{-1})$.

Chapter 4

Hybrid Algorithms

Near-term quantum computers are noisy, limiting the depth of quantum circuits that can be executed within reasonable precision. They are also very limited in their number of qubits available. In near-future quantum devices, it is considered realistic to expect from a few hundred to thousands of qubits.

For comparison, let us look at Shor’s factorization algorithm, which has been hailed as a future application of quantum computation. Shor’s algorithm finds the prime factors of an n -bit integer in efficient time. This is a problem for which all known classical algorithms runs in exponential time. An estimate for the number of noisy qubits needed to perform Shor’s factorization on 2048-bit integers, which would break certain modern encryption schemes, has been found to be over 20 million qubits [7]. The large qubit requirements are due to the quantum error correcting codes, which are needed to obtain the required precision.

Our attention for possible application of NISQ-era devices therefore turns to *hybrid algorithms*. With this, we mean algorithms combining classical and quantum computation, where the quantum part involves some short-depth quantum circuits. Promising hybrid algorithms have been proposed to solve problems within optimization and simulation of quantum systems. Prominent examples are the variational quantum eigensolver [3], and the quantum approximate optimization algorithm [4].

4.1 Variational Quantum Methods

Assume we have a Hamiltonian \mathcal{H} with real eigenvalues $\{\lambda_n\}_n$ and eigenvectors $\{|\psi_n\rangle\}_n$. Order the terms such that $\lambda_0 \leq \lambda_1 \leq \lambda_2 \leq \dots$. The ground state of \mathcal{H} is the eigenstate $|\psi_0\rangle$ corresponding to the minimal eigenvalue λ_0 .

The eigenstate $|\psi_0\rangle$ is referred to as the *ground state*, and in the case of energy eigenvalues, the eigenvalue λ_0 is referred to as the *ground state energy*. If more than one eigenvalue equals λ_0 , the state is said to be *degenerate*.

We want to prepare a quantum state $|\psi(\vec{\theta})\rangle$, parametrized by a set of variational parameters $\vec{\theta}$. We can reformulate the preparation of a quantum

state $|\psi(\vec{\theta})\rangle$ to the execution of a unitary operator $U(\vec{\theta})$, s.t.

$$|\psi(\vec{\theta})\rangle = U(\vec{\theta})|\psi_0\rangle, \quad (4.1)$$

where $|\psi_0\rangle$ is some initial state that can be easily prepared on the quantum device. Commonly, the quantum device is reset to the 0-state at the start of a quantum computation, i.e., $|\psi_0\rangle = |00\dots 0\rangle$.

As λ_0 is the smallest eigenvalue, we know that it gives a lower bound on the expectation value of \mathcal{H} with respect to the state $|\psi(\vec{\theta})\rangle$, s.t.

$$\lambda_0 \leq F(\vec{\theta}) \equiv \langle \psi(\vec{\theta}) | \mathcal{H} | \psi(\vec{\theta}) \rangle. \quad (4.2)$$

This is referred to as the variational principle. For a non-degenerate ground state, equality holds if and only if $|\psi(\vec{\theta})\rangle = |\psi_0\rangle$.

Classical optimization methods can then be used to minimize $F(\vec{\theta})$ with respect to the variational parameters $\vec{\theta}$. The quantum part of each step in the optimization process is then the preparation of $|\psi(\vec{\theta})\rangle$, with the following estimation of the expectation value $F(\vec{\theta})$ by repeated circuit executions, and averaging over the measurement outcomes.

To utilize the variational method on a problem, quantum or classical, we need to find a mapping of the problem to a quantum operator \mathcal{H} such that the solution is given by its ground state. Note that, assuming a finite, discrete spectrum of eigenstates, one can transform this into the problem of finding the *largest* eigenvalue instead by substituting $\mathcal{H} \rightarrow -\mathcal{H}$, s.t. $F(\vec{\theta}) \rightarrow -F(\vec{\theta})$.

4.2 Quantum approximate optimization algorithm

Quantum approximate optimization algorithms aim to approximate solutions to hard computational problems that have no known polynomial-time algorithmic solutions, by use of quantum resources. The basis is the variational principle, as shown in Equation (4.2). In quantum optimization, we encode a *cost function* as a quantum Hamiltonian operator C , and aim to find a state $|\psi(\vec{\theta})\rangle$ with a minimal or maximal expectation value $\langle \psi(\vec{\theta}) | C | \psi(\vec{\theta}) \rangle$. We here present the quantum approximate optimization algorithm (QAOA) for the max-cut problem.

4.2.1 The max-cut problem

The max-cut problem has applications in a variety of areas, including layouts of electronic circuits and combinatorial optimization. The problem is stated as follows: Given a graph $G = (V, E)$, with a set V of $n = |V|$ vertices and a set E of $m = |E|$ edges, find a partition of the vertices V into two complementary subsets, S_0 and S_1 , such that the number of edges between the two subsets is maximized.

A partition of the vertices is encoded into a bitstring $z \in \{0, 1\}^n$. With $\{0, 1\}^n$ we mean the set of all n -bit bitstrings. If the j -th bit is equal to 0, then vertex j belongs subset S_0 , whereas if it is 1, the vertex belongs to S_1 .

The *cost* of a certain solution or partition S_0, S_1 is the number of edges going from a vertex in one subset, S_0 , to the other, S_1 . A cost function for the max-cut problem can then be formulated as a function of $z = z_{n-1} \dots z_1 z_0$ as

$$C(z) = \sum_{(i,j) \in E} C_{(i,j)}(z), \quad (4.3)$$

where (i, j) denotes an edge from vertex i to vertex j and $C_{(i,j)}(z) = 1$ if $z_i \neq z_j$, and $C_{(i,j)}(z) = 0$ if $z_i = z_j$. Solving the max-cut problem now surmounts to finding a bitstring z such that $C(z)$ is maximal.

The max-cut problem is known to be a NP-complete problem, and there are no known solutions which run in polynomial time except for in specific cases [22]. Instead, one aims to find approximate solutions, i.e., solutions z with cost $C(z)$ reasonably close to $C_{max} = \max_z C(z)$. We define $r(z)$ as the *approximation ratio* of z as

$$r(z) = \frac{C(z)}{C_{max}}. \quad (4.4)$$

4.2.2 A Quantum Encoding

A quantum encoding for the max-cut problem is achieved by assigning one qubit to each graph vertex. The state $|0\rangle_j$ on qubit j represents the vertex j being in partition 0. The computational basis set for the full n -qubit Hilbert space needed for the encoding is $\{|z\rangle\}$ for $z \in \{0, 1\}^n$. The basis state $|z\rangle$ represents the partition denoted by the classical bitstring z .

The cost function is represented by an operator C , such that the basis states $|z\rangle$ are its eigenstates. The corresponding eigenvalues are the costs $C(z)$, as given in Equation (4.3). This is achieved by the operator

$$C = \sum_{(k,l) \in E} C_{(k,l)}, \quad (4.5)$$

which is simply a sum over all the edges of the graph. Here, the cost associated with each graph is given by the operator

$$C_{(k,l)} = \frac{1}{2}(\mathbb{I} - Z_k Z_l). \quad (4.6)$$

With Z_i , we here mean the Pauli- Z operator acting on qubit i .

4.2.3 Quantum optimization for max-cut

We have now formulated the max-cut problem as a quantum operator C , given in Equation (4.5). We then need a parametrized state ansatz for which to maximize its expectation value of C with respect to the variational parameters.

The choice proposed by Farhi et al. [4] is, for an integer $p > 0$ and $2p$ angles $\vec{\gamma} = (\gamma_p, \dots, \gamma_1)$ and $\vec{\beta} = (\beta_p, \dots, \beta_1)$, to define the state

$$|\vec{\gamma}, \vec{\beta}\rangle = U(\vec{\gamma}, \vec{\beta}) |s\rangle, \quad (4.7)$$

where

$$|s\rangle = \frac{1}{2^n} \sum_{z \in \{0,1\}^n} |z\rangle, \quad (4.8)$$

is an equal superposition of all computational basis states. The unitary $U(\vec{\gamma}, \vec{\beta})$ is decomposed into one unitary for each angle γ_i , and each β_i , as

$$U(\vec{\gamma}, \vec{\beta}) = U(\beta_p, B)U(\gamma_p, C) \dots U(\beta_1, B)U(\gamma_1, C). \quad (4.9)$$

For any $j \leq p$, these are given as

$$U(\gamma_j, C) = e^{-i\gamma_j C/2} = \prod_{(k,l) \in E} e^{-i\gamma_j C_{(k,l)}/2} \quad (4.10)$$

for C the cost function operator, and

$$U(\beta_j, B) = e^{-i\beta_j B/2} = \prod_{k \in V} e^{-i\beta_j B_k/2}, \quad (4.11)$$

where the choice of Farhi et al. is $B_k = X_k$. The expectation value,

$$F(\vec{\gamma}, \vec{\beta}) = \langle \vec{\gamma}, \vec{\beta} | C | \vec{\gamma}, \vec{\beta} \rangle, \quad (4.12)$$

is then maximized with respect to the angles $\vec{\gamma}$ and $\vec{\beta}$ by use of a classical optimization algorithm.

Observe from Equations (4.10) and (4.11) that by setting $\gamma_k = 0$, we retrieve $U(\gamma_k = 0, C) = \mathbb{I}$, and similarly $U(\beta_k = 0, B) = \mathbb{I}$. Thus, by setting $\gamma_p = \beta_p = 0$ we retrieve the algorithm for $p \rightarrow p-1$. Optimizing the cost function with respect to the variational parameters $\gamma_{p-1}, \dots, \gamma_1$ and $\beta_{p-1}, \dots, \beta_1$ thus gives a lower bound for the solution found by the level- p algorithm, as $\gamma_p = \beta_p = 0$ may not be the optimal choice.

4.2.4 Quantum Circuit for the $p=1$ algorithm

Focusing on the case for $p = 1$, the parameterized state of (4.7) becomes

$$|\gamma, \beta\rangle = \prod_{k \in V} e^{-i\beta X_k/2} \prod_{(k,l) \in E} e^{-i\gamma(\mathbb{I} - Z_k Z_l)/2} |s\rangle, \quad (4.13)$$

where Z_k means the Z operator acting on qubit k .

The state $|s\rangle$ is given in Equation (4.8) and is prepared by applying one Hadamard gate to each qubit, such that

$$|s\rangle = H^{\otimes n} |00 \dots 0\rangle \quad (4.14)$$

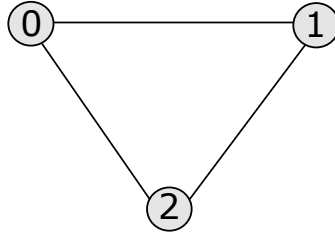


Figure 4.1: A simple undirected graph with vertices $V = \{0, 1, 2\}$ and edges $E = \{(0, 1), (1, 2), (0, 2)\}$.

for n vertices and n qubits, where $H^{\otimes n} = H \otimes \cdots \otimes H$ is the tensor product of n Hadamard gates.

The factors $e^{-i\beta X_k/2}$ of $U(\beta, B_k)$, where $B_k = X_k$ is the chosen ansatz, can be expressed as an operator acting on the k -th qubit as

$$e^{-i\beta X/2} = R_x(\beta), \quad (4.15)$$

where $R_x(\beta)$ is rotation gate about the x axis as defined in Equation (2.64). Similarly, for the factors $e^{-i\gamma C_{(k,l)}/2}$ of $U(\gamma, C)$, we can express

$$e^{-i\gamma C_{(k,l)}/2} = e^{-i\gamma(\mathbb{I} - Z_k Z_l)/2} = e^{-i\gamma/2} e^{i\gamma Z_k Z_l/2}, \quad (4.16)$$

where the factor $e^{-i\gamma/2}$ adds only a global phase factor and can be ignored. As $e^{-i\gamma Z/2} = R_z(\gamma)$, with $R_z(\gamma)$ being the rotation gate about the z -axis, we can implement the $e^{-i\gamma C_{(k,l)}/2}$ operator in a quantum circuit as

$$q_k \text{ --- } \bullet \text{ --- } \bullet \text{ --- } , \quad (4.17)$$

$$q_l \text{ --- } \oplus \text{ --- } \boxed{R_z(\gamma)} \text{ --- } \oplus$$

where we have used the identity $\text{CNOT}(R_z(\gamma) \otimes \mathbb{I})\text{CNOT} = R_z(\gamma) \otimes R_z(\gamma)$.

Take the example graph given in Figure 4.1, with vertices $V = \{0, 1, 2\}$, and undirected edges $E = \{(0, 1), (1, 2), (0, 2)\}$. The quantum circuit the quantum approximate optimization algorithm (QAOA) with $p = 1$, for this specific graph, is given as

$$\begin{aligned} |0\rangle & \text{---} \boxed{H} \text{---} \bullet \text{---} \bullet \text{---} \bullet \text{---} \bullet \text{---} \boxed{R_x(\beta)} \text{---} \text{Measurement} = x_0 \\ |1\rangle & \text{---} \boxed{H} \text{---} \oplus \text{---} \boxed{R_z(\gamma)} \text{---} \oplus \text{---} \bullet \text{---} \bullet \text{---} \bullet \text{---} \boxed{R_x(\beta)} \text{---} \text{Measurement} = x_1 \\ |2\rangle & \text{---} \boxed{H} \text{---} \oplus \text{---} \boxed{R_z(\gamma)} \text{---} \oplus \text{---} \oplus \text{---} \boxed{R_z(\gamma)} \text{---} \oplus \text{---} \boxed{R_x(\beta)} \text{---} \text{Measurement} = x_2 \end{aligned} \quad (4.18)$$

where β and γ are the variational parameters. The measurements are done in the computational basis $\{|0\rangle, |1\rangle\}$, with measurement outcomes 0 or 1 on each qubits, which are stored in the classical bits x_0 , x_1 and x_2 . From these, the measurement outcome, with respect to the cost function operator C , can be calculated. Averaging the measurement outcomes over a series of N shots gives an estimate for the expectation value $F(\gamma, \beta)$.

4.2.5 Time complexity of the expectation value evaluations

The input to the QAOA is a graph $G = (V, E)$ with $n = |V|$ vertices and $m = |E|$ edges. With the measurements done in the computational basis on n qubits, one might be concerned by the fact that there are 2^n possible measurement outcomes. If one needs to sample $\mathcal{O}(2^n)$ states to compute $F(\vec{\gamma}, \vec{\beta})$, this would certainly not be an efficient algorithm. Fortunately, this is not the case.

Let the variance σ^2 in the expectation value $F(\vec{\gamma}, \vec{\beta})$ be estimated by the mean square error. The eigenvalues of the cost function C as shown in Equation (4.5) are bounded above by m , the number of edges in the graph, and bounded below by 0. It follows that the expectation value $F(\vec{\gamma}, \vec{\beta})$ of the cost function with respect to the state $|\vec{\gamma}, \vec{\beta}\rangle$ is equally bounded. Thus we may write

$$\sigma^2 \leq \frac{1}{N-1} \sum_{i=1}^N (\mu_i - F(\vec{\gamma}, \vec{\beta}))^2 \leq \frac{1}{N} \sum_{i=1}^N (m - 0)^2 = m^2. \quad (4.19)$$

Fix some error tolerance ϵ_{tol} . We now estimate the number of shots needed to sample the expectation value $F(\vec{\gamma}, \vec{\beta})$, within the given error tolerance, as

$$N = \frac{\sigma^2}{\epsilon_{\text{tol}}^2} \leq \frac{m^2}{\epsilon_{\text{tol}}^2}. \quad (4.20)$$

In the worst-case scenario of a fully connected graph, where each vertex is connected to every other vertex, the number of edges is $m = \mathcal{O}(n^2)$. This gives $N = \mathcal{O}(m^4)$, i.e., a polynomially asymptotic upper bounded. Because of this scaling, the evaluation of $F(\vec{\gamma}, \vec{\beta})$ for the given parameters $\vec{\gamma}, \vec{\beta}$ is still efficient.

Chapter 5

Quantum Error Mitigation

As discussed so far, the near-future NISQ-era of quantum computation is characterized by two main challenges; noise and limited scaling in terms of available, connected qubits. In Chapter 4, we looked at hybrid algorithms, hoped to be applicable to real-world tasks in the not-so-distant future. These are algorithms in which the quantum part is a short-depth quantum circuit, viable for execution on near-future devices. Still, we may hope to increase the range of viability, and the maximum viable circuit depth, if we can combat the noise present in near-term quantum devices in some meaningful way. In Chapter 3, we presented the basics of quantum error correction and why the associated requirements for ancillary qubits may be prohibitively large for near-term applications. Instead, we turn to *quantum error mitigation*. By quantum error mitigation, we mean techniques that aim to reduce the impact of noise, i.e., to mitigate the noise, in a quantum circuit. This contrasts with quantum error correction that aims to exactly correct sets of correctable errors. This can often be done in practice using no, or only limited, additional quantum resources.

In this chapter, we will present two specific examples of state-of-the-art quantum error-mitigation techniques. There are the zero-noise extrapolation and the quasi-probability decomposition. Additionally, for the zero-noise extrapolation, we introduce a proposal for a new noise-amplification scheme.

A common task of a quantum computation is to prepare a state ρ by some quantum circuit, and then measure its expectation value with respect to some operator A . The expectation value is given as $E = \text{Tr}[A\rho]$. If only the expectation value E is of interest, and not the specifics of the final state ρ , the goal of error mitigation becomes to estimate the true expectation value E^* of the noiseless case as closely as possible [8, 9].

5.1 Zero-noise extrapolation

The idea behind the zero-noise extrapolation scheme for error mitigation is to execute several runs of the noisy quantum circuit, but with the noise amplified by a set of known noise amplification factors. Each time, the expectation

value is evaluated, and from the results of the noise-amplified circuits, the expectation value is extrapolated to the zero-noise case [9].

The preparation of ρ is done by applying a quantum circuit on some initial state ρ_0 . Equivalently, this procedure can be described by a time-dependent Hamiltonian operator $H(t)$ representing the circuit, which evolves the state $\rho(t)$ from the initial point $\rho(t=0) = \rho_0$ to the final state $\rho(t=T)$ at time T . The time evolution of $\rho(t)$ can then be written as

$$\frac{\partial}{\partial t}\rho(t) = -i[H(t), \rho(t)] + \lambda\mathcal{L}(\rho(t)), \quad (5.1)$$

where we have added an additional noise term $\lambda\mathcal{L}(\rho(t))$. We ask for the noise to be weak, i.e., $\lambda \ll 1$.

We can express the Hamiltonian for the ideal circuit as $K(t) = \sum_{\alpha} J_{\alpha}(t)P_{\alpha}$, for $J_{\alpha}(t)$ being a set of time-dependent coefficients and $\{P_{\alpha}\}_{\alpha}$ being the n -qubit Pauli group. The n -qubit Pauli group consists of the tensor products of the single-qubit Pauli operators $\{\mathbb{I}, X, Z, Y\}$, acting on each of the n qubits.

The expectation value can then be expanded around the expectation value of the ideal case as

$$E_K(\lambda) = E^* + \sum_{k=1}^n a_k \lambda^k + O(\lambda^{n+1}), \quad (5.2)$$

where $E_K(\lambda) = \text{Tr}[A\rho(T)]$ is the expectation value of the operator A as a function of the noise parameter λ , $E^* = \text{Tr}[A\rho^*(T)]$ is the expectation value in the ideal case, and the coefficients a_k give the expansion of $E_k(\lambda)$ in terms of λ [8].

5.1.1 Richardson extrapolation

Assume that we, by execution of some quantum circuit, want to evaluate some expectation value $E_K(\lambda)$. We can find an extrapolation to the zero-noise limit if we are able to alter the circuit to run with the error "amplified", such that $\lambda \rightarrow c_i \lambda$. The coefficients $\{c_i\}_i$ are here a set of so-called noise amplification factors. We then evaluate the noise-amplified circuits to obtain associated expectation value estimators $\tilde{E}_K(c_i \lambda)$. Using these, we can utilize *Richardson extrapolation*, [23], which gives an extrapolation to the $\lambda \rightarrow 0$ limit as

$$\tilde{E}_K^n = \sum_{i=0}^n \gamma_i \tilde{E}_K(c_i \lambda), \quad (5.3)$$

where the coefficients γ_i are determined such as to fulfill the restrictions

$$\sum_i \gamma_i = 1, \quad \sum_i \gamma_i c_i^k = 0, \quad (5.4)$$

for all $k = 1, \dots, n$. The problem of finding the coefficients γ_i thus reduces to solving a set of linear equations. Now, \tilde{E}_K^n serves as an estimator for the noise-free expectation value E^* [8].

5.1.2 Variance scaling of the Richardson extrapolation

In the following, we have a look at how the variance in the mitigated expectation value scales with the γ_i -coefficients, when compared to the bare-circuit expectation value. For two uncorrelated random variables X, Y with variances σ_X^2 and σ_Y^2 , the variance of the function $f(X, Y) = a + bX + cY$ becomes $\sigma_{a+bX+cY}^2 = b^2\sigma^2 + c^2\sigma^2$.

Assume $\sigma_{K, c_i \lambda}^2$ are the variances for the estimators $\tilde{E}_K(c_i \lambda)$, estimating the expectation value of the noise amplified circuit with amplification factor c_i . The variance $\sigma_{K, \text{ext}}^2$ of the mitigated expectation value estimator \tilde{E}_K^n , is then found to be

$$\sigma_{K, \text{ext}}^2 = \sum_i \gamma_i^2 \sigma_{K, c_i \lambda}^2, \quad (5.5)$$

i.e., the variance $\sigma_{K, c_i \lambda}^2$ gives a contribution to the total variance scaled by a factor γ_i^2 . This contribution is larger than the bare-circuit variance if $|\gamma_i| \geq 1$.

As an example, solving the system of equations from Equation (5.4) for noise amplification factors $\{1, 2, 3, 4, 5\}$ gives γ_i -values $\{5, -10, 10, -5, 1\}$. Thus, in this specific case, the variance of the mitigated expectation value will be bounded below by the lowest variance in the amplified circuit expectation value estimators, and in the worst case be amplified by a factor of order 10^2 .

The variance in the estimators $\tilde{E}_K(c_i \lambda)$ for the circuit expectation values can be controlled by adjusting the number of shots for which the circuit is executed, and that the measurement outcomes are averaged over. The per-circuit error estimation is detailed in Section 2.5.10. To control the variance and error in the mitigated expectation value, however, the influence of the γ_i -coefficients must be taken into account as just explained. This give a means to determine the number of shots for each specific, noise-amplified circuit, such that the error in the final mitigated expectation value estimators are within our specified tolerance.

5.1.3 Noise amplification by repeating CNOT-gates

To employ the zero-noise extrapolation technique, we first need to specify the noise we want to mitigate. In modern quantum hardware, the error rate in two-qubit gates, such as the CNOT gate, are in most cases significantly larger than in single-qubit gates [24]. For a device with a native gate set consisting of single-qubit gates and CNOT-gates, a reasonable choice is thus to aim to mitigate noise in CNOT gates.

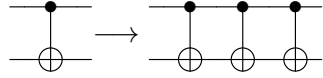
We here propose an approach to amplifying general noise in the CNOT-gate, which to the best of our knowledge has not been researched before. We first note that the CNOT-gate is its own inverse, i.e.,

$$(\text{CNOT})^2 = \text{CNOT} \circ \text{CNOT} = \mathbb{I}, \quad (5.6)$$

where both CNOT gates act on the same control and target qubits. Consider the gate $[\text{CNOT} \circ (\text{CNOT})^{2m}]$, $m \in \mathbb{N}$, where each gate act on the same control

and target qubits. In the noiseless case, this gate then has the same action as a single CNOT-gate. For a noisy gate, however, that by Equation (3.1) can be expressed as $\mathcal{E} \circ \text{CNOT}$, with CNOT being the ideal gate and \mathcal{E} the associated noise channel, the noise channel \mathcal{E} will be applied $2m + 1$ times to the quantum state throughout the execution. Thus, by replacing each CNOT-gate in a circuit by a series of $2m + 1$ identical CNOT-gates, we obtain an circuit in which the general noise in the CNOT-gates has been amplified by a factor $c_m = 2m + 1$.

By constructing and executing quantum circuits with noise amplification factors $c_m = 2m + 1$, $m = 1, 2, 3, \dots$, we obtain measured, noise-amplified expectation values $\tilde{E}_K(c_m)$. As an example, take the noise-amplification factor $c_1 = 3$, for which this substitution becomes


(5.7)

By obtaining the noise-amplified, estimated expectation values for circuits with noise amplification factors $1, 3, 5, \dots, 2n + 1$, we obtain the estimated *mitigated* expectation value \tilde{E}_K by use of the Richardson extrapolation, as shown in Equation (5.3).

5.2 Quasi-probability decomposition

The quasi-probability decomposition is a technique for mitigating errors by expressing the noise-free circuit as a sum of noise-afflicted circuits, then sampling from those by a Monte Carlo scheme. In this way, we obtain an unbiased estimator for the noise-free expectation value, with the trade-off of an increased variance [8, 9]. To be able to employ the technique of quasi-probability decomposition, we need a reasonably accurate model of the noise present in the quantum device in question. This can be obtained by use of *gate set tomography* [25], a concept that we will not go into details on in this project.

5.2.1 Decomposition of an ideal circuit

Assume that a quantum circuit, in the noise free case, may be described by the initialization of a state $\rho^{(0)}$, a sequence of quantum channels $O_1^{(0)}, \dots, O_N^{(0)}$, representing unitary gates, followed by the measurement of an observable $A^{(0)}$. The expectation value then becomes

$$E^* = \text{Tr}[A^{(0)} O_{\text{tot}}^{(0)}(\rho^{(0)})], \quad (5.8)$$

where $O_{\text{tot}}^{(0)} = O_N^{(0)} \dots O_1^{(0)}$. Assume that the actual noise-afflicted circuit is described by the initial state ρ , a sequence of quantum channels O_1, \dots, O_N , and the noisy measurement described by the observable A , such that

$$E = \text{Tr}[A O_{\text{tot}}(\rho)]. \quad (5.9)$$

Assume further that we may construct a set of noise-afflicted circuits $\{O_{\text{tot}}^{(l)}\}_l$, with associated initial state $\rho^{(l)}$ and observables $A^{(l)}$, such that

$$E^* = \sum_l q_l \text{Tr}[A^{(l)} O_{\text{tot}}^{(l)}(\rho^{(l)})], \quad (5.10)$$

for a set of coefficients $q_l \in \mathbb{R}$. This can be done for specific error models, given that one has full knowledge of the error model describing the noise [9]. The expectation values of the noisy circuits are

$$E_l = \text{Tr}[A^{(l)} O_{\text{tot}}^{(l)}(\rho^{(l)})], \quad (5.11)$$

which are estimated by executing the circuit and averaging over the measurement outcomes for a sufficient number of shots, as to obtain the desired precision. Then, the noise-free expectation value is expressed as

$$E^* = \sum_l |q_l| (\text{sgn}(q_l) E_l) = \sum_l |q_l| E_{\text{eff},l} = \sum_l p_l C E_{\text{eff},l}. \quad (5.12)$$

Defining $C = \sum_l |q_l|$, this becomes

$$E^* = \sum_l p_l C E_{\text{eff},l}, \quad (5.13)$$

where $\{p_l\}_l$ forms a probability distribution, i.e., $p_l \geq 0$ and $\sum_l p_l = 1$.

Sampling from the set of circuits l with associated probabilities p_l , and calculating the estimated, rescaled expectation values $C E_{\text{eff},l}$, thus, by Equation (5.13) [8], gives an unbiased estimator for E^* .

5.2.2 Per-gate decomposition

Errors in each gate, in addition to initialisation and measurement, may be corrected by decomposing the ideal gate $O_i^{(0)}$ into a set of sub-circuits $O_i^{(l_i)}$ such that

$$O_i^{(0)} = \sum_{l_i} q_{l_i}^{[i]} O_i^{(l_i)}, \quad (5.14)$$

and similarly for state initialisations and measurements. The decomposition of the full circuit then gives the noise-free expectation value E^* as

$$E^* = \sum_{l_{\text{meas}}} \sum_{l_N} \cdots \sum_{l_1} \sum_{l_{\text{init}}} q_{l_{\text{meas}}}^{[\text{meas}]} q_{l_N}^{[N]} \cdots q_{l_1}^{[1]} q_{l_{\text{init}}}^{[\text{init}]} \text{Tr}[A^{(l_{\text{meas}})} O_N^{(l_N)} \cdots O_1^{(l_1)}(\rho^{l_{\text{init}}})] \quad (5.15)$$

If we assume that the noise in multi-qubit gates, such as the CNOT-gate, is dominant, then we take advantage of the technique of Pauli-twirling that transforms general errors into stochastic Pauli-errors [26]. This may ease the analysis of the error models for the quantum device, which are crucial to be able to employ the quasi-probability decomposition. In this project, we will not go further in detail into the technique of Pauli-twirling.

Chapter 6

Implementation

In this project, we have implemented two choice algorithms. As an example of a hybrid algorithm, and to demonstrate on a small scale the possible near-future applications of quantum computers, we have implemented the quantum approximate optimization algorithm (QAOA) for the max-cut problem. For a quantum error-mitigation technique, we have implemented the zero-noise extrapolation, but with our proposed noise-amplification scheme for CNOT-gates. Both have been implemented using `qiskit`, an open-source software development kit for quantum computation using Python, which is developed by IBM. With `qiskit`, we can access a range of publicly available physical quantum computers, provided by the IBM Quantum Experience, as well as simulators for both noise-free and noisy quantum computation [16, 21].

In this chapter, we will present some aspects of `qiskit` that are important for our implementations of said algorithms. We further present the most important details of our implementation, both of the QAOA, which is presented in Chapter 4, and the zero-noise extrapolation, which is described in Chapter 5. The code has been made available in the following, public repository at: github.com/AndersHR/quantum_error_mitigation.

6.1 Qiskit

This section presents some important aspects of the `qiskit` software development kit that are important in order to understand our implementations of the aforementioned quantum algorithms.

6.1.1 Quantum Circuits in `qiskit`

The central object in `qiskit` is the *quantum circuit*, which describes quantum computations as detailed in Section 2.5.3. A `qiskit` quantum circuit object contains quantum registers of qubits, classical registers for storage of measurement results, and a sequence of quantum gates.

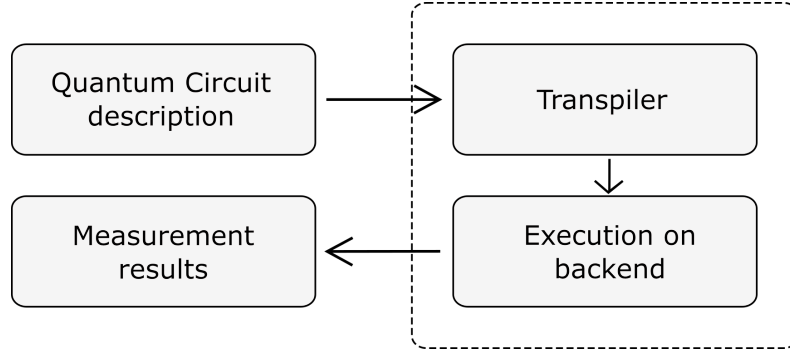


Figure 6.1: The main flow of a quantum computation executed using `qiskit`. The input is a description of a quantum circuit, said circuit is transpiled for and executed on the specified backend, and the measurement results is outputted.

The qubits are usually prepared in the 0-state $|00\dots0\rangle$. This state is evolved by the specified sequence of quantum gates, up until some final state, and then measurements on some, or all of the qubits are done in the computational basis. Measurements are handled as a special type of gate. Note that `qiskit` only supports measurements done at the end of any given quantum circuit. To extract information about the final state, as each measurement outcome is probabilistic, the experiment is repeated for a specified number of *shots*.

The measurement result for each experiment is a bit-string, specifying the measured computational basis state on each of the measured qubits. The output of the full circuit execution is a set of *measurement counts*, specifying how many times each possible measurement result was recorded.

Before execution, the input circuit is *transpiled* to be optimized, and to be compatible with the specifications of the specified backend. The basic flow of a quantum computation performed in `qiskit` is sketched in Figure 6.1.

Figure 6.2 shows an example circuit constructed in `qiskit`. Assuming a noiseless computation, the state at the last vertical dotted line right before measurement is $|q_0q_1q_2q_3\rangle = \frac{1}{2}(|0000\rangle - |0111\rangle + |1011\rangle - |1100\rangle)$. With measurements done on qubits q_0 , q_1 , and q_2 , the possible measurement outcomes are 000, 011, 101, and 110, each with a probability of $\frac{1}{4}$. Say 8192 shots of this circuit are performed. A possible final measurement count is then $\{000 : 2085, 011 : 2014, 101 : 2004, 110 : 2089\}$.

6.1.2 Backend specifications and the transpiler

The physical realization of quantum hardware imposes restrictions on the circuits that can be executed on the given quantum devices. An important restriction is the *CNOT connectivity* of quantum devices. Quantum devices with native gate sets consisting of single-qubit gates and CNOT gates are sufficient

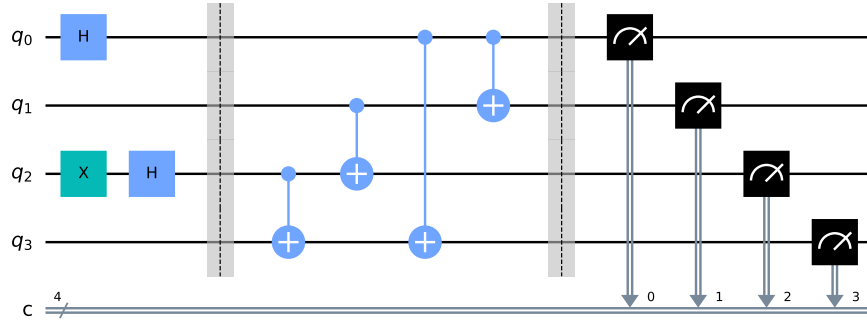


Figure 6.2: A quantum circuit created in `qiskit`, with a quantum register of size 3 and a classical register of size 3. The vertical dotted lines are *barriers*, whose only purpose is separating gates for readability.

for general quantum computing. But physical realizations of such devices seldom have physically realized CNOT gates between every pair of qubits. The physically implemented CNOT gates can be described by a *connectivity map* or *coupling map*. The connectivity map for the IBMQ Vigo device with 5 qubits is shown in Figure 6.3.

General quantum circuits can still be executed on any quantum hardware as long as they fit within the available number of qubits provided by the device, but they may have to be transformed. This is in `qiskit` done by the *transpiler*. The transpiler employs different circuit transformations, known as *transpiler passes* [17].

A first technique for adapting a given circuit for execution on a specific backend is a mapping from *virtual qubits* to *physical qubits*. In the example circuit shown in Figure 6.2 there are 4 virtual qubits q_0 , q_1 , q_2 , and q_3 . Say we want to execute this circuit on the IBMQ Vigo device with 5 physical qubits, marked with indices $i = 1, 2, 3, 4, 5$, and CNOT connectivity as given in Figure 6.3. The naive mapping $q_0 \rightarrow 0$, $q_1 \rightarrow 1$ and so on leaves us unable to perform the first CNOT gate in Figure 6.2 between q_2 and q_3 , as the IBMQ Vigo device does not have CNOT connectivity between physical qubits 2 and 3. An alternative mapping, with $q_3 \rightarrow 0$, $q_2 \rightarrow 1$, takes care of this problem.

After optimizing the mapping of virtual qubits to physical qubits, there may still be CNOT gates that cannot be performed due to connectivity restrictions. Any *logical* CNOT gate on two virtual qubits can however be performed by adding SWAP-gates, as defined in Equation (2.85). Note that a SWAP gate is implemented using CNOT gates.

As an example, assume that we for a circuit to be executed on a device with a connectivity map as shown in Figure 6.3, have the mapping $q_0 \rightarrow 3$, $q_1 \rightarrow 2$, $q_2 \rightarrow 1$, $q_3 \rightarrow 0$, and that this mapping was chosen to enable a set of CNOT-gates to be executable. Assume further that we later in the quantum

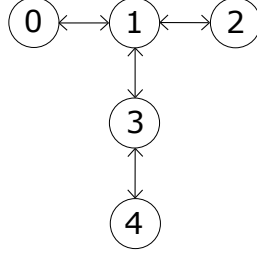


Figure 6.3: Connectivity map for the IBMQ Vigo and IBMQ Valencia NISQ backends. Nodes denote qubits, marked by indices $i = 1, 2, 3, 4, 5$, and arrows denote which qubits are connected by CNOTgates [21].

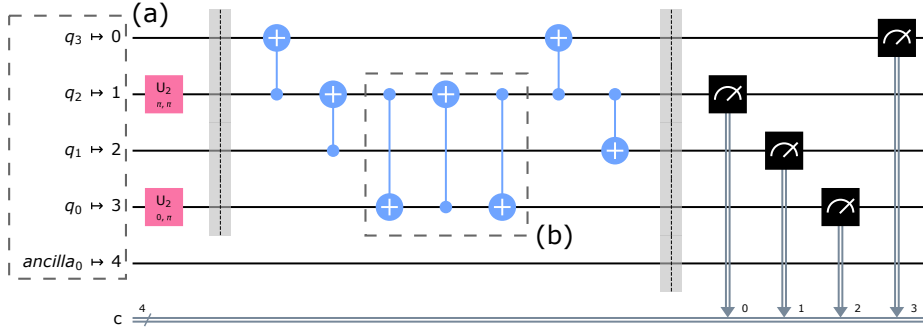


Figure 6.4: A quantum circuit transpiled to the specifications of the IBMQ Vigo 5-qubit device with optimization level 1. The original circuit is shown in Figure 6.2. (a) gives the mapping from the virtual qubits q_0, q_1, q_2 and q_3 of the original circuit to the physical qubits of the IBMQ Vigo device. An ancilla qubit is assigned to the leftover physical qubit. (b) is an added SWAP gate which swaps q_0 and q_3 between their assigned physical qubits 1 and 3.

circuit want to perform a CNOT gate with q_0 as control and q_3 as target. There is no CNOT-connectivity between physical qubit 3 and 0. There is, however, CNOT connectivity between physical qubits 0 and 1, and between 1 and 3. A SWAP gate between physical qubits 3 and 1 swaps the states on the two physical qubits, which reassigns q_0 to qubit 1, and q_2 to qubit 3. After the SWAP, the CNOT between q_3 and q_0 can be performed by a CNOT between physical qubits 1 and 3, which are connected.

The SWAP gates can be thought of as adjusting the virtual-physical qubit mapping along the duration of the quantum circuit. Note that this does add additional circuit depth and CNOT complexity to the circuit, which on a physical device adds noise.

Qiskit allows circuits to be expressed using a wide variety of gates. Before execution, all gates in a circuit must be decomposed, or *unrolled*, to the native gate set of the quantum device in question. For the IBMQ devices publicly

available at the time of writing, this is the U_1 , U_2 , U_3 , and CNOT gates. In principle, other native gate sets are fully possible and compatible with `qiskit`, for example the U_3 gate and controlled- Z gate. This is handled by the transpiler, and the `Unroller` transpiler pass.

Furthermore, the transpiler optimizes the circuit to minimize noise. On a physical quantum device, each gate has an associated noise. Minimizing the number of gates needed, and thus minimizing the circuit depth, is important for minimizing noise. A simple gate optimization is collapsing adjacent single-qubit gates. As each single-qubit gate can be seen as a rotation of the Bloch vector, adjacent gates on the same qubit can be collapsed into one single, total rotation. Adjacent CNOT gates with the same control and target qubits can be removed, as $\text{CNOT} \circ \text{CNOT} = \mathbb{I}$. More powerful gate optimizations can be done by taking commutation rules between CNOT and U_3 gates into account.

6.1.3 Error models and mock backends

With `qiskit`, we can build our own, custom error models for noisy simulation. `Qiskit` provides functionality both for building error models using standard errors, such as phase errors, phase-damping errors, amplitude-damping errors and others, as well as for building general quantum error channels by defining sets of Kraus operators.

The desired noise model can be created by composing different errors, e.g., $\mathcal{E}(\rho) = \mathcal{E}_2(\mathcal{E}_1(\rho))$, and taking tensor products between different errors, e.g., $\mathcal{E}(\rho) = \mathcal{E}_1 \otimes \mathcal{E}_2(\rho)$. Errors can be assigned to initialization of qubits, to quantum gates, including the identity gate, and to measurements. Recall that by Equation 3.1, faulty gates, state initializations, and measurements can be defined as the ideal, noiseless gate, state initialization or measurement either followed by, or preceded by, a noise channel.

A useful, related feature offered in `qiskit` is *mock backends*. For any physical quantum backend provided by IBMQ, an associated mock backend that emulates the physical backend is implemented. For example, for the IBMQ Vigo backend, the associated mock backend is implemented through the `FakeVigo` class. The `FakeVigo` mock backend is a simulator backend with the exact same configurations as the IBMQ Vigo backend, including connectivity map and native gate set. And it is embedded with an error model created by IBM, to emulate the error on the physical backend. The errors are formulated in terms of sets of Kraus operators. Different errors are applied on each gate in the native gate-set, including the identity operator, as well as on initialisations and on measurements.

6.2 The QAOA for max-cut with $p=1$

The QAOA for the max-cut problem has been implemented as detailed in Section 4.2. The QAOA was then run on the specific max-cut problem defined by the graph given in Figure 6.5.

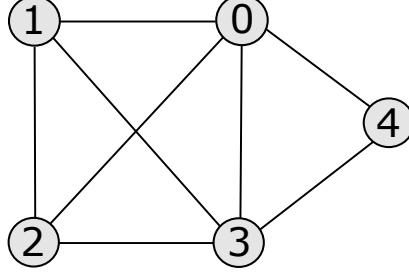


Figure 6.5: An undirected graph with vertices $V = \{0, 1, 2, 3, 4\}$ and edges $E = \{(0, 1), (0, 2), (0, 3), (0, 4), (1, 2), (1, 3), (2, 3), (3, 4)\}$.

The optimal partition of the vertices of the graph, up to symmetry, is $S_0 = \{0, 3\}$, $S_1 = \{1, 2, 4\}$, with a cost of six. This partition, not accounting for symmetry, is encoded into the bit strings 01101 and 10010. The equivalent quantum encodings are the computational basis states $|01101\rangle$ and $|10010\rangle$. There are 12 further "good" solutions with a cost of five, one of which is $S_0 = \{0, 1\}$, $S_1 = \{2, 3, 4\}$ encoded in the bit string 00111.

Figure 6.6 shows the quantum circuit as created in `qiskit` for the QAOA, for the graph given in Figure 6.5 and $p = 1$. The QAOA was implemented using the implementation of the Nelder–Mead classical optimization algorithm from the `scipy` library for Python. The function used was `scipy.optimize.minimize` [27, 28]. The absolute tolerances for termination of the optimization procedure was set to $x_{atol} = 10^{-2}$ in the variational parameters and $f_{atol} = 10^{-1}$ in the cost function value.

The cost function as implemented takes in the variational parameters γ and β , estimates the expectation value $F(\gamma, \beta) = \langle \gamma, \beta | C | \gamma, \beta \rangle$ by executing the quantum circuit preparing $|\gamma, \beta\rangle$ and subsequent measurements for 8192 shots on a specified backend, either a physical backend provided by IBM or a simulator backend with or without a noise model. Before execution, the quantum circuit was optimized for the chosen backend by the `qiskit` transpiler with the `optimization_level = 1` preset.

The value $-F(\gamma, \beta)$ is then returned to be used in the classical optimization procedure. The negative of the estimated expectation value was returned as the `scipy` optimization implementation minimizes the given cost function with respect to the variational parameters. Minimizing $-F(\gamma, \beta)$ is equivalent to maximizing $F(\gamma, \beta)$.

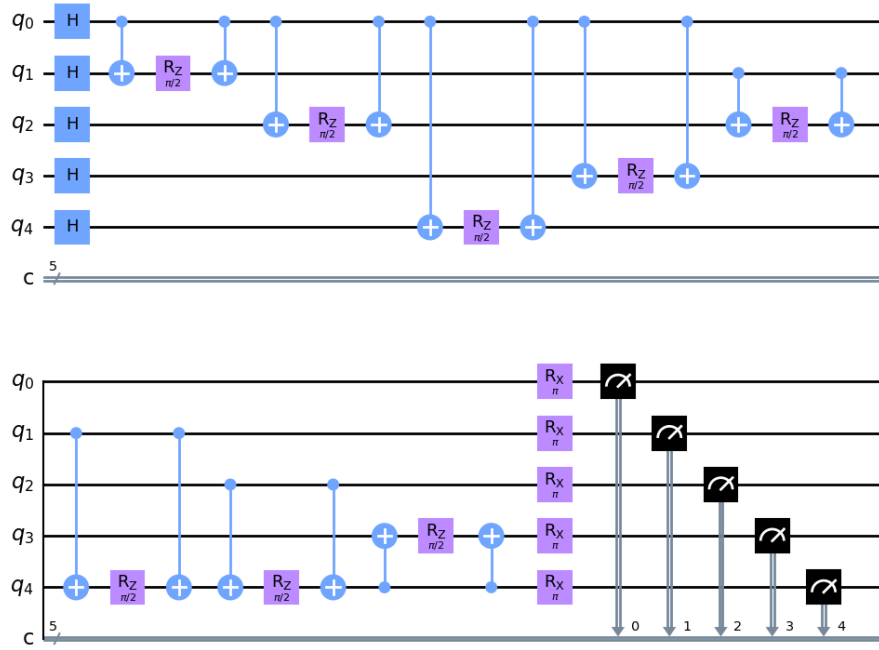


Figure 6.6: Quantum circuit created in `qiskit` for the QAOA, for the graph given in Figure 6.5, with $p = 1$, $\gamma = \pi/2$ and $\beta = \pi$. q_i for $i = 1, 2, 3, 4, 5$ are the virtual qubits representing the 5 vertices of the graph. The two rows are taken to loop around in such a way that the rightmost side of the upper row is equal to the leftmost side of the lower row.

6.3 Zero-noise amplification for CNOT gates

The zero-noise extrapolation was implemented using noise amplification by the proposed method of repeating CNOT-gates. This approach, to the best of our knowledge, has not been researched previously.

The so-called SWAP-test circuit has been previously employed to examine different error mitigation techniques in an article [9]. The circuit implements a SWAP gate, as defined in Equation (2.85), but by replacing the CNOT-gates with TOFFOLI gates, and adding an ancillary *probe qubit*, the overlap between the swapped states $|\psi\rangle$ and $|\phi\rangle$ is measured by a Z -measurement on the probe qubit. The SWAP-test circuit for two single-qubit states $|\psi\rangle$ and $|\phi\rangle$ is given as

$$\begin{array}{c}
 |0\rangle \text{ --- } [H] \text{ --- } \bullet \text{ --- } \bullet \text{ --- } \bullet \text{ --- } [H] \text{ --- } \text{Measurement} = x \\
 |\psi\rangle \text{ --- } \bullet \text{ --- } \oplus \text{ --- } \bullet \\
 |\phi\rangle \text{ --- } \oplus \text{ --- } \bullet \text{ --- } \oplus
 \end{array} \quad (6.1)$$

where the probe qubit is the upper qubit, which is initialised to the $|0\rangle$ -state. The expectation value of the Z -type measurement on the probe qubit becomes

Chapter 7

Results and discussion

This chapter presents and discusses the results found from the algorithms we have implemented; the QAOA, and the zero-noise extrapolation with our proposed noise-amplification scheme. The QAOA was implemented for the case of $p = 1$, and employed on the example graph given in Figure 6.5. The zero-noise extrapolation was applied on the so-called SWAP-test circuit, with 3 qubits, as shown in Figure 6.7.

7.1 QAOA for Max-Cut with $p=1$

Figure 7.1 shows the values of $F(\gamma, \beta)$ obtained with the $p = 1$ algorithm for $0 \leq \gamma, \beta \leq 2\pi$, evaluating the quantum circuit on a noiseless simulator backend. The QAOA with the specified parameters was run from the starting point $\gamma_0 = 1.3, \beta = 2.6$. The backend used to evaluate $F(\gamma, \beta)$ was a noise-free simulator backend. The optimized variational parameters was found to be $\gamma^* = 0.52$ and $\beta^* = 2.57$ after 19 iteration steps. The cost function expectation value of this state was estimated to be $F(\gamma^*, \beta^*) = 4.95$.

The QAOA was then run similarly, but using the IBMQ Valencia NISQ backend to execute the quantum circuit that evaluates $F(\gamma, \beta)$. This backend has a connectivity map as shown in Figure 6.3. The optimized variational parameters were then found to be $\gamma' = 1.11, \beta' = 2.72$ after 20 iteration steps with the maximum cost function expectation value found to be $F(\gamma', \beta') = 4.30$.

The optimized states $|\gamma^*, \beta^*\rangle$ and $|\gamma', \beta'\rangle$ were then prepared and measured in the computational basis using the noise-free simulator backend and the IBMQ Valencia backend, respectively; each time for a total of $N = 8192$ shots. The normalized measurement counts n_z/N are shown in Figure 7.2. Here, n_z denotes the number of times the outcome z occurred, for $z \in \{0, 1\}^n$ associated with the computational basis state $|z\rangle$, whereas n_z/N gives an estimation of the probabilities $p_z = |\langle \gamma, \beta | z \rangle|^2$ of measuring outcome z .

Considering the state $|\gamma^*, \beta^*\rangle$ as found by the QAOA with the noise-free simulator backend, we can see from Figure 7.2 that the computational basis states with a cost of 5 or above are very prominent. Measuring the state $|\gamma^*, \beta^*\rangle$

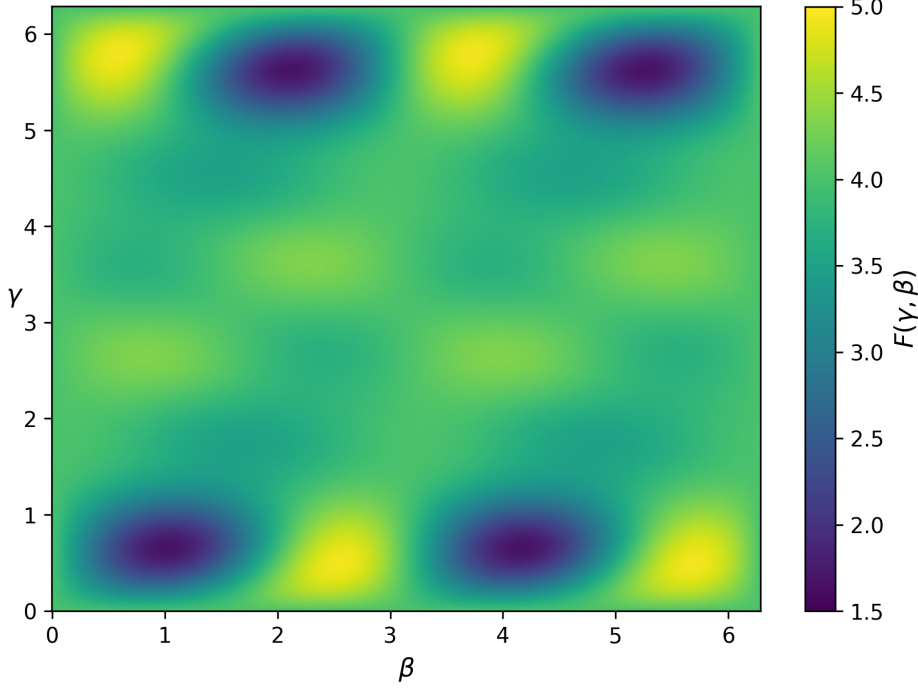


Figure 7.1: Values of the cost function expectation value $F(\gamma, \beta) = \langle \gamma, \beta | C | \gamma, \beta \rangle$ of the QAOA with $p = 1$ and the graph as shown in Figure 6.5. For each γ and β the expectation value was estimated by performing 8192 shots of the associated quantum circuit using a simulator with zero noise. This was done for a 1000 by 1000 square lattice of equally spaced points (γ, β) where $0 \leq \gamma, \beta \leq 2\pi$.

in the computational basis in fact yields a computational basis state with cost $C \geq 5$ with probability 0.90. Thus, given the optimized values of the variational parameters γ and β , one can obtain a partition of the graph with a “good” max-cut cost $C \geq 5$ with very high probability by repeating the measurement a couple of times. The probability of obtaining one of the optimal basis states, for which $C = 6$, is 0.19.

For the QAOA executed on the physical `ibmq_valencia` quantum backend, we can from Figure 7.2 observe the effect of noise. The probability distribution for the measurement outcome seems to be more uniform. The a measurement outcome has a probability of yielding a state with cost $C \geq 5$ of 0.58. This is only a slight improvement over the random-pick strategy, of randomly assigning each vertex either 0 or 1 with equal probability, which could give such a state with probability $\frac{14}{2^5} = 0.4375$. This highlights the need for error mitigation for the QAOA and similar algorithms.

The obtained optimized expectation value in the noise-free case is similar to what has been found in a recent implementation of the QAOA [29], where

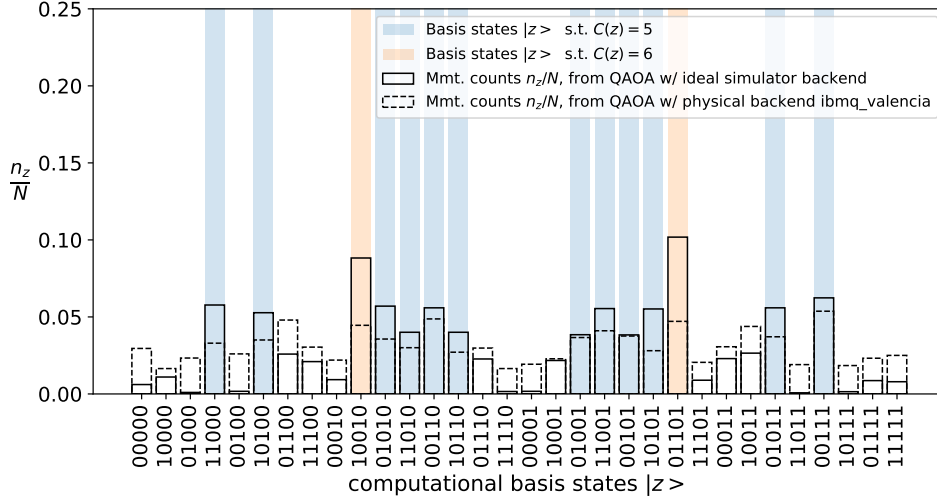


Figure 7.2: Normalized measurement counts $\frac{n_z}{N}$ estimating the probabilities $p_z = |\langle \gamma, \beta | z \rangle|^2$ of outcomes z for the states $|\gamma^*, \beta^*\rangle$, prepared and measured on a noise free simulator, shown by the full line and the state $|\gamma', \beta'\rangle$, prepared and measured on the IBMQ Valencia physical backend, shown by the dotted line. The states are the optimized solutions when running the QAOA for the graph given in Figure 6.5, with $p = 1$, starting points $\gamma_0 = 1.3, \beta_0 = 2.6$, and with the cost function expectation value $F(\gamma, \beta)$ evaluated for each optimization iteration by executing 8192 shots of the quantum circuits on a noiseless simulator and the IBMQ Valencia NISQ backend respectively. The orange and blue bars respectively indicate the computational basis states with cost function eigenvalue $C(z) = 6$, which is the optimal and with cost function eigenvalues $C(z) = 5$.

an optimized expectation value was found as $F(\gamma, \beta) = 4.77$ with $p = 1$ and other initial values for γ and β . Here, this has been slightly exceeded using other initial points.

Possible future improvements on this experiment is to employ error mitigation to the procedure while executing the QAOA on a noisy backend. NISQ devices are noisy, and for the QAOA to be of practical use, one needs to reduce the impact of that noise. Note that the point of interest for the QAOA is the composition of the optimized state $|\gamma^*, \beta^*\rangle$ itself, not necessarily the expectation value $F(\gamma^*, \beta^*)$. We want to measure the state to obtain a computational basis state $|z\rangle$ representing a partition of the graph vertices V , and for that state to have a high cost C with high probability. Thus, techniques like the zero-noise extrapolation likely may have limited usability for the QAOA.

7.2 Zero-noise extrapolation for CNOT gates

The zero-noise extrapolation using the proposed method of repeating CNOT gates was implemented as presented in Section 6.3, then employed on the SWAP-test circuit shown in Figure 6.7. The Z -operator measurement on the probe qubit done in the SWAP-test circuit has a known expectation value of $E^* = 0.5$ in the noise-free case [9]. The SWAP-test circuit on three qubits was chosen due to its reasonable CNOT complexity. If the circuit depth is such that the end state due to compounded noise decoheres into a fully mixed state, with computational basis state measurement probabilities close to a uniform distribution, we would not expect the zero-noise extrapolation to be of any considerable use. This was not found to be the case for the SWAP-test circuit.

Each noise-amplified quantum circuit for ten noise amplification factors $1, 3, 5, \dots, 19$ was executed using the **FakeVigo** simulator backend, emulating the **ibmq_vigo** backend, for a total of $N^* = 5000 \cdot 8192 = 4.096 \cdot 10^7$ shots. We chose the **FakeVigo** mock backend to give a noise as realistic as possible while still running the quantum circuits simulated locally. We chose the total number of shots to be approximately one order of magnitude above the required number of shots to achieve at least an estimated precision of 0.01 in all mitigated expectation values. We will touch more on this later on in this section.

Based on a sample of 8192 shots for each circuit i with amplification factor $c_i = 2i - 1$, the variances σ_i^2 were estimated by the mean square error. Table 7.1 shows the estimated variances, as well as the expectation value estimates $E[(2i - 1)\lambda]_{N=N^*}$ from averaging over all the $N^* \approx 4 \cdot 10^7$ shots. From the table it becomes clear that the noise-amplification procedure skews

Table 7.1: Estimated expectation values $E[(2i-1)\lambda]$, variances σ_i^2 and circuit depths for the SWAP-test circuit noise-amplified by noise amplification factors $c_i = 2i - 1$, $i = 1, 2, \dots, 10$. The expectation values are estimated based on $N^* \approx 4 \cdot 10^7$ shots executed for each circuit. The variances σ_i^2 are estimated by the mean square error from a sample of 8192 shots from each circuit. The circuit depths are measured after the SWAP circuit first was transpiled for the specifications of the **FakeVigo** backend, emulating the **ibmq_vigo** quantum backend, with the highest optimization preset, and then noise-amplified by repeating each CNOT-gate $c_i = 2i - 1$ times.

	$i = 1$	$i = 2$	$i = 3$	$i = 4$	$i = 5$
$E[(2i - 1)\lambda]$	0.387	0.239	0.149	0.0960	0.0652
σ_i^2	0.850	0.942	0.977	0.991	0.997
Circuit depth	50	104	158	212	266
	$i = 6$	$i = 7$	$i = 8$	$i = 9$	$i = 10$
$E[(2i - 1)\lambda]$	0.0475	0.0376	0.0317	0.0284	0.0264
σ_i^2	0.999	0.999	0.998	1.00	1.00
Circuit depth	320	374	428	482	536

the noise-amplified expectation values further and further from the ideal expectation value of $E^* = 0.5$. Without error mitigation, the best estimate for this expectation value is the bare circuit expectation value $E[1\lambda] = 0.387$. The relative deviation from the noise-free expectation value in this estimate becomes $\Delta^* = \frac{|0.5-0.387|}{0.5} = 0.226$. This is the error we here hope to mitigate by use of the zero-noise extrapolation.

Figure 7.3 shows the estimated expectation values $E[c_i\lambda]_N = \frac{1}{N} \sum_{k=1}^N \mu_k^{(c_i\lambda)}$ for $i = 1, \dots, 5$, as well as the mitigated expectation value $E[1, \dots, 2n-1]_N$ for $n = 5$, as a function of the number of shots N included in the averages.

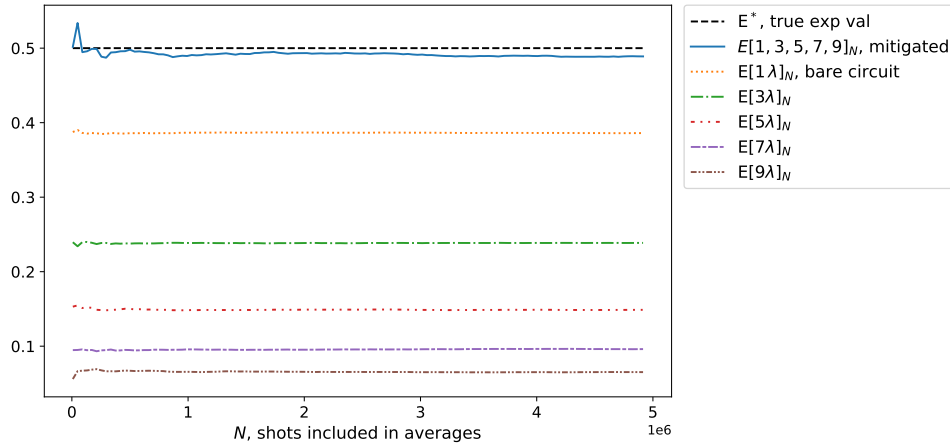


Figure 7.3: Estimated expectation values $E[(2i-1)\lambda]$ and variances σ_i^2 , as well as circuit depths, for noise-amplified SWAP-test circuits with noise amplification factors $c_i = 2i-1$ for $i = 1, 2, \dots, 10$. The circuits are transpiled with maximal optimization level by the `qiskit` transpiler for the specifications of the `FakeVigo` mock backend, emulating the `ibmq_vigo` quantum backend, and noise amplified by the method of repeating CNOT-gates.

The figure illustrates the idea behind the zero-noise extrapolation in a good way. From the noise-afflicted bare-circuit expectation value $E[1\lambda]$, the noise amplification moves the found expectation values of the noise-amplified circuits further and further from the noise-free expectation value $E^* = 0.5$. The mitigated expectation value $E[1, 3, 5, 7, 9]$ found by extrapolation then converges to a value within 0.01 of E^* . After $N^* \approx 4 \cdot 10^7$ shots per circuit, the mitigated expectation value for $n = 5$ is estimated to $E[1, 3, 5, 7, 9]_{N=N^*} = 0.491$.

We may also note from Figure 7.3 that the mitigated expectation value converges significantly slower than the expectation values of each noise-amplified circuit. This is as expected and can be seen from Equation (5.5). If we assume that the estimators $E[c_i\lambda]$, for the different noise-amplified circuit expectation values, are uncorrelated, then the variance of the mitigated expectation value is proportional to the variance of each noise-amplified expectation value multiplied by the corresponding γ_i -coefficient squared. The γ_i -coefficients are

therefore very important for the variance scaling in the mitigated expectation value. The coefficients $\gamma_1^{(n)}, \gamma_2^{(n)}, \dots, \gamma_n^{(n)}$ calculated based on noise amplification factors $1, 3, \dots, 2n-1$ for the specific cases of n from $n=2$ to $n=10$ are shown in Table 7.2.

Table 7.2: Values of the coefficients $\gamma_i^{(n)}$, $i=1, 2, \dots, n$, from the Richardson extrapolation with noise amplification factors $c_i = 2i-1$, $i=1, 2, \dots, n$, for different values of n . The coefficients in each case of n are calculated from Equation (5.4). The values are given to 3 significant digits.

	$n=2$	$n=3$	$n=4$	$n=5$	$n=6$	$n=7$	$n=8$	$n=9$	$n=10$
$\gamma_1^{(n)}$	1.50	1.88	2.19	2.46	2.71	2.93	3.14	3.11	2.97
$\gamma_2^{(n)}$	-0.500	-1.25	-2.19	-3.28	-4.51	-5.87	-7.33	-7.05	-6.00
$\gamma_3^{(n)}$	—	0.375	1.31	2.95	5.41	8.80	13.2	12.2	8.80
$\gamma_4^{(n)}$	—	—	-0.313	-1.41	-3.87	-8.38	-15.7	-13.8	-7.52
$\gamma_5^{(n)}$	—	—	—	0.273	1.50	4.89	12.2	9.78	3.07
$\gamma_6^{(n)}$	—	—	—	—	-0.24	-1.60	-6.00	-4.05	0.0247
$\gamma_7^{(n)}$	—	—	—	—	—	0.226	1.69	0.717	-0.240
$\gamma_8^{(n)}$	—	—	—	—	—	—	-0.21	0.0691	-0.274
$\gamma_9^{(n)}$	—	—	—	—	—	—	—	-0.0348	0.239
$\gamma_{10}^{(n)}$	—	—	—	—	—	—	—	—	0.0514

By use of the estimated variances shown in Table 7.1 and the γ_i -coefficients used in the Richardson extrapolation for the various cases of n shown in Table 7.2, one can estimate the number of shots of the different circuits needed to obtain an error in the estimator $E[1, \dots, 2n-1]$ within some chosen tolerance. Let us restrict the case to each amplified circuit being executed for the same number of shots. This may not be optimal, as the different γ_i -coefficients may vary a lot for a given n , but it gives a good idea of the total number of circuit executions needed to mitigate the increased variance in the mitigated expectation values. For a given n and error tolerance ϵ_s , the required number of shots is estimated as

$$N_s^{(n)} = \frac{\sum_{i=1}^n \gamma_i^2 \sigma_i^2}{\epsilon_s^2}. \quad (7.1)$$

The total number of circuit executions over all the noise-amplified circuits is then $N_{\text{tot}}^{(n)} = n \cdot N_s^{(n)}$. The values of $N_s^{(n)}$ found for different values of n from 1 to 10 are shown in Table 7.3, where $n=1$ is the case for the bare circuit with $c_1=1$. This is not found by extrapolation, as the other values are, but is included for completeness.

To investigate whether the error scaling acts as expected, we repeat the error mitigation procedure 1000 times for $n=8$, with $N=8192$ shots for each circuit. The variance in the mitigation estimator $E[1, 3, 5, 7, 9, 11, 13, 19]_{N=8192}$ was then estimated by the mean-square error over the 1000 repeats. This was found to be $\sigma_{n=8}^2 = 0.0812$, such that $\sigma_{n=8} = 0.285$. This variance is in close

Table 7.3: Mitigated expectation values $E[1, \dots, 2n-1]_N$ for the SWAP-test circuit and the estimated number of shots $N_s^{(n)}$ required to achieve an error in the estimator $E[1, \dots, 2n-1]_N$ below the threshold of $\epsilon_{\text{tol}} = 0.01$.

	$n = 1$	$n = 2$	$n = 3$	$n = 4$	$n = 5$
$N_s^{(n)}$	$8.5 \cdot 10^3$	$2.15 \cdot 10^4$	$4.60 \cdot 10^4$	$1.04 \cdot 10^5$	$2.58 \cdot 10^5$
$N_{\text{tot}}^{(n)}$	$8.5 \cdot 10^3$	$4.30 \cdot 10^4$	$1.38 \cdot 10^5$	$4.14 \cdot 10^5$	$1.29 \cdot 10^6$
$E[1, \dots, 2n-1]_{N=N^*}$	0.387	0.461	0.483	0.490	0.492
	$n = 6$	$n = 7$	$n = 8$	$n = 9$	$n = 10$
$N_s^{(n)}$	$7.11 \cdot 10^5$	$2.11 \cdot 10^6$	$6.61 \cdot 10^6$	$5.01 \cdot 10^6$	$1.83 \cdot 10^6$
$N_{\text{tot}}^{(n)}$	$4.27 \cdot 10^6$	$1.48 \cdot 10^7$	$5.29 \cdot 10^7$	$4.51 \cdot 10^7$	$1.83 \cdot 10^7$
$E[1, \dots, 2n-1]_{N=N^*}$	0.493	0.493	0.493	0.493	0.493

agreement with the expected variance found by Equation (5.5). From the per-circuit variances σ_i^2 , the $\gamma_i^{(n=8)}$ -coefficients and $N = 8192$, this is calculated to be $\tilde{\sigma}_{n=8}^2 = 0.0807$, such that $\tilde{\sigma}_{n=8} = 0.284$.

The mitigated expectation values $E[1, \dots, 2n-1]_N$ for $N = N^* \approx 4 \cdot 10^7$ shots are shown in Table 7.3. Here, $E[1, \dots, 2n-1]_N$ is the estimated expectation value found by Richardson extrapolation based on the expectation values $E[c_i \lambda]_N$, $i = 1, \dots, n$, for the noise-amplified circuits with amplification factors $c_i = 2i - 1$, estimated by averaging over N circuit executions. These are shown in the case of $N = N^* \approx 4 \cdot 10^6$ shots. Similarly, $N_s^{(n)}$ is the number of shots for each circuit required to obtain an error lower than an error tolerance $\epsilon_{\text{tol}} = 0.01$, whereas $N_s^{(n)}$ was for each n calculated from Equation (5.5) based on the variances shown in Table 7.1 and the γ -coefficients shown in Table 7.2, under the assumption that of the each noise-amplified circuits are executed for the exactly same amount of shots. $N_{\text{tot}}^{(n)} = n \cdot N_s^{(n)}$ is then the total number of circuit executions. The values for the bare-circuit case $n = 1$ are included for completeness, and are not found by extrapolation.

The expectation values $E[1, \dots, 2n-1]_{N=N^*}$ as a function of n are shown in Figure 7.4. The estimated error in the expectation values for $N = N^*$ is $\epsilon_{N^*} = 0.004$ for the case of $n = 8$, and below for the other displayed cases on n . From Figure 7.4, we observe that the zero-noise extrapolation gives a mitigated expectation value $E[1, \dots, 2n-1]$ with a significant improvement over the bare circuit expectation value of $E[1] = 0.387$. By $n = 4$, the mitigated expectation value is within 0.01 of the noise-free expectation value, representing a relative deviation of $\Delta_{n=4}^* = \frac{|0.5-0.490|}{0.5} = 0.02$. The mitigated expectation values then seem to plateau at a value of 0.493, such that $\Delta_{n \geq 6}^* = \frac{|0.5-0.493|}{0.5} = 0.014$. This is an improvement of one order of magnitude in the deviation from the noise-free expectation value, as compared with the bare-circuit case of $\Delta^* = 0.226$.

The significant improvement achieved by the CNOT error mitigation suggests that the assumption of CNOT-noise being the dominant noise is a valid

one. That the mitigated expectation value converges to 0.493 as a function of n , suggests that the remaining deviation is probably caused by a combination of single-qubit and measurement errors.

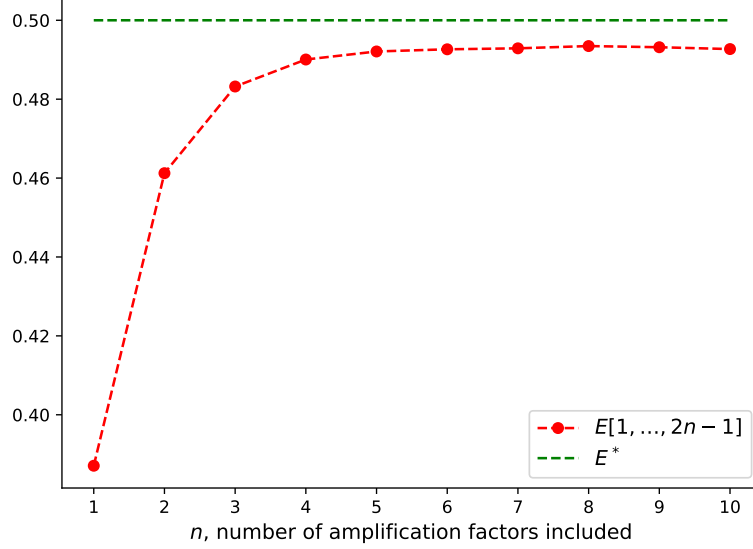


Figure 7.4: Mitigated expectation values $E[1, \dots, 2n-1]$ for the SWAP-test circuit, with n noise-amplification factors $1, 3, \dots, 2n-1$ included in the Richardson extrapolation. The expectation value $E[(2i-1)\lambda]$ for each of the noise-amplified circuits were estimated based on $N^* \approx 4 \cdot 10^7$ shots. The green dotted line is the noise-free expectation value $E^* = 0.5$.

From Tables 7.1 and 7.3, we observe that whereas the per-circuit variances σ_i^2 do not change dramatically for different i , the impact of the specific $\gamma_i^{(n)}$ -coefficients is significant. In fact, even though intuitively one might expect the required total number of circuit executions to increase with n , as the number of circuits increases, we instead find a peak in the variance at $n = 8$. We here find that $N_{\text{tot}}^{(n)}$ is larger for $n = 8$ than for both $n = 9$ and $n = 10$. Still, for each of the cases, the required total numbers of circuit shots are several orders of magnitude above what is needed for the bare-circuit expectation value to converge.

The mitigated expectation value was nonetheless found to have significant improvements over the bare-circuit expectation value even for $n = 2$ and $n = 3$, which do not have the same, prohibitively large requirements in number of circuit executions needed to obtain the same precision. For $n = 2$, we find a relative deviation from the noise-free case of $\Delta_{n=2}^* = 0.078$. At the same time, the required total number of circuit executions to obtain an estimated error within 0.01 is $N_{\text{tot}}^{(2)} = 4.30 \cdot 10^4$, is approximately a factor 5 larger than for the bare circuit.

Chapter 8

Conclusion

In this specialization project, we have studied the possible near-term application of quantum computation. We have studied hybrid algorithms, thought to become the first applicable use-cases for quantum computers on real-world problems, through the quantum approximate optimization algorithm for the max-cut problem. As means to combat the noise present on near-term devices, we have studied some quantum error-mitigation techniques. Near-term devices are yet to reach the scale where quantum error correction might be viable. Therefore, techniques for quantum error mitigation are important for NISQ-era quantum computation, as they have the potential to extend the range of viability for quantum computations, while being applicable of near-term devices.

The potential application of quantum computers in optimization problems was demonstrated by the QAOA for the max-cut problem, run with $p = 1$, on a specific instance of the max-cut problem for which the optimal solution has cost $C = 6$. When run on a quantum simulator, the optimization procedure terminated at a state with expectation value $F(\vec{\gamma}^*, \vec{\beta}^*) = 4.95$. To extract a solution represented as a bit string, this state can be prepared and measured, and the end state was found to have a probability of 0.9 to be measured in a state with cost $C \geq 5$. When run on a noisy, physical backend, the expectation value of the end state was found to be $F(\vec{\gamma}', \vec{\beta}') = 4.30$. In this case, the probability of measuring a state with a “good” cost of $C \geq 5$ was found to be 0.58, only a slight improvement over the random-pick strategy. Future work on the QAOA, as implemented here, includes exploring possible error-mitigation schemes capable of improving the results of the QAOA, when running the circuits on noisy quantum hardware.

Our implementation of the zero-noise extrapolation, with our proposed noise amplification method of repeating CNOT-gates, was tested on the SWAP-test circuit with a mock simulator backend. This gave promising results, with significant improvements over the bare-circuit calculations for the chosen SWAP-test circuit. The relative deviation from the ideal noise-less expectation value of $E^* = 0.5$ was improved by an order of magnitude. From the bare-circuit expectation value of $E[1] = 0.387$, the mitigated expectation value $E[1, 3, \dots, 2n - 1]$

with amplification factors $1, 3, \dots, 2n - 1$ was found to converge to a value of 0.493 as a function of n .

For possible applications, the most realistic compromise between improvements in the expectation value and the increased variance is found for the cases of $n = 2$ and $n = 3$ number of amplification factors. In these cases, the technique still showed significant improvements over the bare-circuit case, but with variances staying within an order of magnitude of the bare-circuit variance. This means that the required number of circuit executions needed to obtain the same precision was only about an order of magnitude larger than for the bare-circuit.

Future improvements on the algorithm, as implemented here, includes adjusting the number of shots of each noise amplified circuit taking the γ_i values of the Richardson extrapolation into account. We have showed a significant impact of these coefficients on the variance in the mitigated expectation values. By taking more shots of the circuits where the product $(\gamma_i \sigma_i)^2$ is largest, the total number of circuit executions, needed to obtain the same precision, can be reduced.

The results, furthermore, remains to be verified on real quantum hardware. Although the mock backends provided by `qiskit` aim to closely mimic their respective physical backend, the noise is defined as quantum channels on each gate, and on each measurement. This may not necessarily be realistic for more complicated noise present on physical quantum hardware.

Finally, it would be interesting to compare the effectiveness of this zero-noise extrapolation technique with other state-of-the-art schemes for error mitigation, as well as to investigate the potential for combining this method with other mitigation methods and examine if this could further improve the noise mitigation.

Bibliography

- [1] F. Arute, K. Arya, R. Babbush and et al., ‘Quantum supremacy using a programmable superconducting processor’, *Nature*, vol. 574, pp. 505–510, 2019. DOI: [10.1038/s41586-019-1666-5](https://doi.org/10.1038/s41586-019-1666-5).
- [2] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information, 10th Anniversary Edition*. Cambridge University Press, 2016, ISBN: 978-1-107-00217-3.
- [3] P. K. Barkoutsos, J. F. Gonthier, I. Sokolov, N. Moll, G. Salis, A. Fuhrer, M. Ganzhorn, D. J. Egger, M. Troyer, A. Mezzacapo, S. Filipp and I. Tavernelli, ‘Quantum algorithms for electronic structure calculations: Particle-hole hamiltonian and optimized wave-function expansions’, *Phys. Rev. A*, vol. 98.2, p. 022322, 2018. DOI: [10.1103/PhysRevA.98.022322](https://doi.org/10.1103/PhysRevA.98.022322).
- [4] E. Farhi and J. Goldstone, ‘A quantum approximate optimization algorithm’, *arXiv preprint arXiv:1411.4028*, 2014.
- [5] P. W. Shor, ‘Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer’, *SIAM J. Comput.*, vol. 26.5, pp. 1484–1509, 2006. DOI: [10.1137/S0097539795293172](https://doi.org/10.1137/S0097539795293172).
- [6] P. W. Shor, ‘Scheme for reducing decoherence in quantum computer memory’, *Phys. Rev. A*, vol. 52.4, R2493–R2496, 1995. DOI: [10.1103/PhysRevA.52.R2493](https://doi.org/10.1103/PhysRevA.52.R2493).
- [7] C. Gidney and M. Ekerå, ‘How to factor 2048 bit rsa integers in 8 hours using 20 million noisy qubits’, *arXiv preprint arXiv:1905.09749*, 2019.
- [8] K. Temme, S. Bravyi and J. M. Gambetta, ‘Error mitigation for short-depth quantum circuits’, *Phys. Rev. Lett.*, vol. 119.18, p. 180509, 2017. DOI: [10.1103/PhysRevLett.119.180509](https://doi.org/10.1103/PhysRevLett.119.180509).
- [9] S. Endo, S. C. Benjamin and Y. Li, ‘Practical quantum error mitigation for near-future applications’, *Phys. Rev. X*, vol. 8.3, p. 031027, 2018. DOI: [10.1103/PhysRevX.8.031027](https://doi.org/10.1103/PhysRevX.8.031027).
- [10] A. Kandala, K. Temme, A. D. Córcoles, A. Mezzacapo, J. M. Chow and J. M. Gambetta, ‘Extending the computational reach of a noisy superconducting quantum processor’, *arXiv preprint arXiv:1801.00862*, 2018.

- [11] J. Sun, X. Yuan, T. Tsunoda, V. Vedral, S. C. Benjamin and S. Endo, ‘Mitigating realistic noise in practical noisy intermediate-scale quantum devices’, *arXiv preprint arXiv:2001.04891*, 2020.
- [12] V. N. Premakumar and R. Joynt, ‘Error mitigation in quantum computers subject to spatially correlated noise’, *arXiv preprint arXiv:1812.07076*, 2018.
- [13] Y. Li and S. C. Benjamin, ‘Efficient variational quantum simulator incorporating active error minimisation’, *Phys. Rev. X*, vol. 7.2, p. 021 050, 2017. DOI: [10.1103/PhysRevX.7.021050](https://doi.org/10.1103/PhysRevX.7.021050).
- [14] P. Suchsland, F. Tacchino, M. Fischer, T. Neupert, P. Baskoutsos and I. Tavernelli, ‘Algorithmic error mitigation scheme for current quantum processors’, *arXiv preprint arXiv:1612.02058*, 2020.
- [15] F. Maciejewski, Z. Zimboras and M. Oszmaniec, ‘Mitigating measurement errors in quantum computers by exploiting state-dependent bias’, *Quantum*, vol. 4, p. 257, 2020. DOI: [10.22331/q-2020-04-24-257](https://doi.org/10.22331/q-2020-04-24-257).
- [16] H. Abraham, A. Offei, R. Agarwal *et al.*, *Qiskit: An open-source framework for quantum computing*, 2019. DOI: [10.5281/zenodo.2562110](https://doi.org/10.5281/zenodo.2562110).
- [17] IBM. (2020). Qiskit documentation. Accessed: 11.11.2020, [Online]. Available: <https://qiskit.org/documentation/>.
- [18] J. B. Fraleigh, *A First Course in Abstract Algebra, 7th Edition*. Pearson Education Limited, 2013, ISBN: 9781292024967.
- [19] P. C. Hemmer, *Kvantemekanikk*. Tapir akademisk forlag, 2005, ISBN: 9788251920285.
- [20] T. E. Cormen, C. E. Leiserson, R. L. Rivest and C. Stein, *Introduction to Algorithms, Third edition*. The MIT Press, 2009, ISBN: 978-0-262-53305-8.
- [21] IBM. (2020). Ibm quantum experience. Accessed: 11.11.2020, [Online]. Available: <https://quantum-computing.ibm.com/>.
- [22] F. Hadlock, ‘Finding a maximum cut of a planar graph in polynomial time’, *SIAM J. Comput*, vol. 4, pp. 221–225, 1973. DOI: [10.1137/0204019](https://doi.org/10.1137/0204019).
- [23] L. F. Richardson, ‘The approximate arithmetical solution by finite differences of physical problems involving differential equations, with an application to the stresses in a masonry dam’, *Phil. Trans. R. Soc. Lond. A*, vol. 210, pp. 307–357, 1910. DOI: [10.1098/rsta.1911.0009](https://doi.org/10.1098/rsta.1911.0009).
- [24] C. J. Ballance, T. P. Harty, N. M. Linke, M. A. Sepiol and D. M. Lucas, ‘High-fidelity quantum logic gates using trapped-ion hyperfine qubits’, *Phys. Rev. Lett.*, vol. 117.6, p. 060 504, 2016. DOI: [10.1103/PhysRevLett.117.060504](https://doi.org/10.1103/PhysRevLett.117.060504).
- [25] E. Nielsen, J. K. Gamble, K. Rudinger, T. Scholten and R. Blume-Kohout, ‘Gate set tomography’, *arXiv preprint arXiv:2009.07301*, 2020.

- [26] Z. Cai and S. Benjamin, ‘Constructing smaller pauli twirling sets for arbitrary error channels’, *arXiv preprint arXiv:1807.04973*, 2018.
- [27] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. J. Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. J. Carey, Í. Polat, Y. Feng, E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt and SciPy 1.0 Contributors, ‘Scipy 1.0: Fundamental algorithms for scientific computing in python’, *Nature Methods*, vol. 17, pp. 261–272, 2020. DOI: [10.1038/s41592-019-0686-2](https://doi.org/10.1038/s41592-019-0686-2).
- [28] F. Gao and L. Han, ‘Implementing the nelder-mead simplex algorithm with adaptive parameters’, *Computational Optimization and Applications*, vol. 51.1, pp. 259–277, 2012. DOI: [10.1007/s10589-010-9329-3](https://doi.org/10.1007/s10589-010-9329-3).
- [29] F. Fuchs, N. H. Aase, H. Ø. Kolden, V. Falch and C. Johnsen. (2020). Open quantum computation. Accessed: 17.12.2020, [Online]. Available: <https://github.com/OpenQuantumComputing>.