A Benchmark For Quantum Computers: The Effective Qubit Number

Et Benchmark For Kvante Computere: Det Effektive Qubit Tal

Frederik Rosendal Rytter
202005589

Master's Thesis in Physics
June 2025

Written in Collaboration with Kvantify ApS
Supervised by Assoc. Prof. Dmitri Fedorov
and Dr. Niels Jakob Søe Loft

Department of Physics and Astronomy
Aarhus University

Abstract

Good benchmarks should enable users of quantum devices to understand their computing power, thereby driving progress toward the long-term goal of performing useful quantum computations, i.e., quantum utility, while poor benchmarks will misinform users and hinder progress.
In the NISQ era, application-oriented single-number metrics are necessary to understand the problem-solving capabilities of quantum devices and to answer questions such as: "What can this quantum computer do?". The primary objective of this thesis is to develop an application-oriented, holistic single-number metric based on quantum phase estimation, which is easy to read, share, and compare.
Reviewing and analyzing the work of a former physics master's student from Aarhus University will lead to the construction of a precise roadmap for achieving the primary objective. Simulations of quantum hardware will be used to guide development and test results, while experiments on quantum hardware will put the results into real-world contexts.
The main result of this thesis is the construction of an application-oriented, holistic single-number metric referred to as the effective qubit number, which is defined as the largest number of qubits on which a device can run a prefixed test implementation of quantum phase estimation before noise "takes over". The effective qubit number is shown to be system-robust and stable under slight variations in the noise strength of a quantum computing system.
The effective qubit number is a novel contribution to the sparse collection of application-oriented single-number metrics, based on quantum phase estimation, which is easy to read, share, and compare.

Contents

CHAPTER 1

Introduction

In the field of quantum computing, benchmarks are tools used to measure the performance of quantum computing devices. The rapid development of quantum computing technologies, from devices with a small number of qubits [1] to devices consisting of tens to hundreds of qubits [2, 3, 4, 5], has led to the creation of a diverse range of benchmarks [6, 7, 8, 9]. Good benchmarks should enable scientists and industrial users of quantum devices to understand their computing power, thus driving progress towards the long-term goal of performing useful quantum computations, i.e., quantum utility. In contrast, poor benchmarks will misinform users and hinder progress toward quantum utility. Constructing good benchmarks for quantum computers thus poses a significant problem.

Benchmarks for quantum computers, roughly speaking, fall into two categories: low-level benchmarks that quantify the effects of qubit and gate errors [6], and high-level benchmarks that quantify the overall performance of an entire quantum computing stack. The latter mainly consists of two types: randomized benchmarks, like the quantum volume [7], which quantify the ability of a quantum computing device to implement random unitaries with high fidelity, and application-oriented benchmarks, like IonQ's number of algorithmic qubits [9], which quantify the quantum computing device's ability to reliably implement unitary gates relevant for solving computationally significant problems. For NISQ-era quantum computers, an important practical question is: "What can this quantum computer do?". Application-oriented benchmarks directly measure computing power, making it easy to answer such questions.

It is well understood that qubit count alone is insufficient to understand the performance of quantum computing devices. Even so, qubit count is still one of the first things mentioned when comparing devices. It is easy to understand why; it is easy to read, share, and compare. Unfortunately, it neither describes progress towards quantum utility nor the computing power of a given device, making it a very misleading single-number metric. A significant problem is thus to construct single-number metrics that, unlike qubit count, capture the problem-solving power of a quantum device, as well as progress towards quantum utility, while remaining easy to read, share, and compare.

The primary objective of this thesis is to develop an application-oriented, holistic single-number metric based on quantum phase estimation, which serves as a proxy for quantum advantage. The discussion above underlines the importance of such a construction. Good benchmarks should provide reliable results, be well-motivated, well-defined, efficient concerning both classical and quantum resources, as well as

implementation-robust, i.e., it should not be possible to "game" the benchmark to obtain better results by manipulating its configurable parameters [10]. Following this definition, a crucial subsequent task is to assess whether the constructed single-number metric is a good benchmark. A final task is to compare it with the values of well-established single-number metrics, such as quantum volume, on actual hardware.

The main result of this thesis is the construction of an application-oriented, holistic single-number metric referred to as the effective qubit number, which is simple to read, share, and compare. It can be interpreted as the largest number of qubits on which a device can run a specific test implementation of quantum phase estimation before noise "takes over", where noise "taking over" will be defined in a precise mathematical way. The effective qubit number is well-defined, well-motivated, efficient concerning both classical and quantum resources, and system-robust, i.e., repeated measurements of the effective qubit number tend to agree. It has also been tested on quantum hardware from different providers and compared to other single-number metrics [7, 9].

In Chapter 2, essential theory and formalism from the field of quantum computing and quantum information used in this thesis are outlined. The effective qubit number will be based on the quantum phase estimation algorithm, which is therefore discussed thoroughly in this chapter.
Chapter 3 discusses the key principles used in producing good benchmarks for quantum computers. Alongside this, examples from the large variety of present benchmarks are provided. Inspiration will be drawn from these examples throughout the thesis.

Chapter 4 is split into several sections. Sec. 4.1 reviews a previous master's student's definition of an application-oriented, holistic single-number metric based on quantum phase estimation. The subsequent section thoroughly analyzes this definition, identifying how it deviates from the key principles underlying good benchmarks as discussed in Chapter 3. Inspired by this discussion, a path forward for the development of the effective qubit number is specified in Sec. 4.3. This is done by writing down explicit open problems, whose solutions are needed in the construction of the effective qubit number. In Sec. 4.4-4.6, these open problems are solved, leading to the definition of the effective qubit number in Sec. 4.7.

In Chapter 5, key properties of the effective qubit number are tested using a custom-made noise model, whose construction is outlined in Appendix B. In addition, the sensitivity of the effective qubit number to slight variations in the underlying noise model is examined.
Based on the noise sensitivity discussions, a continuous extension of the effective qubit number is proposed in Chapter 6, in the hope that for such an extension, slight variations in the underlying noise model always correspond to small variations in the effective qubit number.

Lastly, in Chapter 7, the effective qubit number and its continuous extension are tested on IBM's quantum hardware as well as on hardware from an anonymized provider from the UK. The experimental measurements of the effective qubit number are compared to the values of other single-number metrics.

CHAPTER 2

Quantum Computing in a Nutshell

The foundation of quantum computing is that computation is a physical process. As such, computation should follow the laws of quantum mechanics. At first sight, this might seem like complicating a perfectly flourishing field, but the treasures and implications awaiting make it worthwhile the effort. Specifically, the interference of quantum amplitudes, superposition, and entanglement of quantum states provide powerful tools for producing efficient quantum algorithms in solving problems previously thought to be classically intractable.

Quantum computing lies at the intersection of quantum physics, mathematics, and computer science. Historically, quantum mechanics as we know it today was developed in the 1920s to explain perplexing phenomena at the atomic scale. It continued its development throughout the 20th century, and to this day, alongside the development of the first digital computers, which were produced to replace human computers for tedious computations. The production of these digital computers was the beginning of a field we now call computer science.

The initial motivation for developing quantum computers was the potential ability to simulate the evolution of quantum mechanical systems effectively. The Hilbert space of a given physical system, say a 1-dimensional spin-chain, has dimensionality that scales exponentially with the system size, where the system size, in this case, is the number of sites. It is for this reason that the simulation of quantum systems, e.g., spin-chains or atoms and molecules, is a classically hard problem to solve. Richard Feynman realized this in 1982, when he proposed in an article [11] that hardware based on quantum phenomena might be more efficient in simulating quantum systems. This idea started the development of the field we now call quantum computing.

In 1984, two years after Feynman's article, Charles Bennett and Gilles Brassard applied quantum mechanics to the key distribution problem in cryptography [12]. They demonstrated that quantum key distribution can offer enhanced information security compared to classical key distribution methods. In the following decade, several algorithms for the so-called quantum computers were proposed. These include solutions to oracle problems like Deutsch's algorithm [13], Grover's algorithm for searching unstructured databases [14], and the famous Shor's algorithm [15], promising that scalable quantum computers can break RSA cryptation, the cryptation on which the security of all our bank accounts is based. This development sparked plenty of industrial and public interest in quantum computers.

Quantum computing systems that can effectively simulate any other quantum system are called universal. These will be the focus of this thesis - devices like D-Wave's quantum annealers will thus naturally be excluded from the discussion. Gate-based quantum computers are the most dominant type of universal quantum computers. Still, others exist - it turns out that any quantum algorithm, including unitary evolution, can be simulated using non-unitary measurements [16]! Cluster-state quantum computers are built on this concept. Still, they are not as technologically mature as gate-based quantum computers and are thus also excluded from the discussions in this thesis.

2.1 Gate-Based Quantum Computing

This section is intended as a brief introduction to the key concepts of gate-based quantum computing. Inspiration has been drawn both from the classic textbook *Quantum Computation and Quantum Information* by Michael A. Nielsen and Isaac L. Chuang [17] and the lecture notes *Introduction to Quantum Information Science* by Arthur Eckert et. al. from a course taught at Oxford University [18]. It is assumed from here on and throughout the rest of the thesis that the reader has a working knowledge of quantum mechanics.

The formalism of gate-based quantum computing is very similar to the formalism of classical computing, where bits are replaced by two-dimensional quantum systems called qubits, and single- and multi-bit logical gates are replaced by single- and multi-qubit unitary evolutions called quantum gates. The **state of a qubit** can always be written as,

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle \tag{2.1.1}$$

with α and β being complex numbers, and $\{|0\rangle, |1\rangle\}$ an orthonormal basis for the two-dimensional Hilbert space representing all the possible states of the qubit. There is always a reference basis $\{|0\rangle, |1\rangle\}$ specified, which is usually called the computational basis. Depending on the physical realization of the qubits, the basis $\{|0\rangle, |1\rangle\}$ could correspond to the spins $\{|\uparrow\rangle, |\downarrow\rangle\}$ of an electron or the ground- and excited-states $\{|g\rangle, |e\rangle\}$ of a Rydberg atom. The usual convection is to represent the basis $\{|0\rangle, |1\rangle\}$ by complex two-dimensional vectors as

$$|0\rangle \to \begin{bmatrix} 1 \\ 0 \end{bmatrix} \quad \text{and} \quad |1\rangle \to \begin{bmatrix} 0 \\ 1 \end{bmatrix}. \tag{2.1.2}$$

Whenever a quantum state of a single or multiple qubits is written as a vector, or a unitary gate on a single or multiple qubits is written as a matrix, it is always with respect to the computational basis. To emphasize that we are representing a quantum state or a quantum gate with respect to a basis, the equal sign $\doteq$ is used.

To do interesting computations, we need more than one qubit! Any state of $n \in \mathbb{N}$ identical qubits can be written as,

$$|\psi\rangle = \sum_{m_1,\ldots,m_n} c_{m_1,\ldots,m_n}\,|m_1,\ldots,m_n\rangle \in \mathcal{H}^{\otimes n}, \quad m_i \in \{0,1\}. \tag{2.1.3}$$

Such a n-qubit quantum state is said to be **entangled** if it cannot be written as the tensor product of single-qubit quantum states, i.e., if,

$$|\psi\rangle \neq |\psi_1\rangle \otimes |\psi_2\rangle \otimes \cdots \otimes |\psi_n\rangle \,. \tag{2.1.4}$$

Entanglement proves to be a crucial quantum mechanical resource for constructing efficient quantum algorithms.

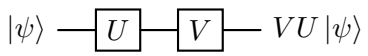
The allowed quantum gates we can perform on qubits are restricted by the constraint of quantum mechanics that the time evolution of quantum systems is unitary. The quantum gates we can apply to qubits are thus unitary operators. An interesting consequence of this **unitarity** requirement is the no-cloning theorem, which states that a quantum computer cannot copy quantum states [17]! This is in great contrast to our usual experience with classical computers, just consider the words you are currently reading.

A few highly relevant examples of **single-qubit quantum gates** are,

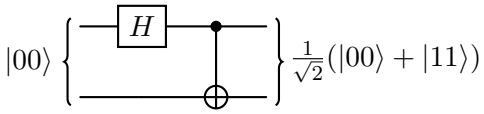
$$H \doteq \frac{1}{\sqrt{2}}\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix}, \quad P(2\pi\phi) \doteq \begin{bmatrix}1 & 0\\ 0 & e^{i2\pi\phi}\end{bmatrix}, \quad T = P(\frac{\pi}{4}) \doteq \begin{bmatrix}1 & 0\\ 0 & e^{i\frac{\pi}{4}}\end{bmatrix}. \tag{2.1.5}$$

These are called the Hadamard-gate, Phase-gate, and T-gate, respectively. For further examples, I refer the reader to the book by Michael Nielsen and Isaac Chuang [17]. In contrast to classical single-bit logic gates, there are infinitely many single-qubit quantum gates. In fact, every element in the group of 2×2 unitary matrices, $U(2)$, constitutes a valid single-qubit quantum gate.

In quantum computing, it is conventional to represent the action of such gates on qubits using what we call **quantum circuits**. As an example:



The left state $|\psi\rangle$ represents the initial quantum state of a single qubit. The wire may represent the evolution of time or movement through space; either way, it goes from left to right. The state $VU\,|\psi\rangle$ represents the outcome of applying U followed by V to $|\psi\rangle$. Note that the order of the operators in $VU\,|\psi\rangle$ is reversed compared to their order in the circuit, which is a consequence of reading the circuit from left to right. If we have more qubits, we just add more wires!



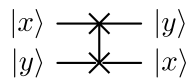
This circuit provides our first example of how entanglement can be constructed using quantum gates. Specifically, the result of this circuit is one of the highly entangled Bell states. In the entire thesis, we follow the convention that the left-most qubit in states like $|00\rangle$ corresponds to the top qubit in the quantum circuit. Besides entanglement, this circuit provides the first, and probably the most important, example of

a **multi-qubit quantum gate** called the CNOT gate,

$$\text{CNOT} = |0\rangle\langle 0| \otimes \mathbb{I} + |1\rangle\langle 1| \otimes X \doteq \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}. \tag{2.1.6}$$

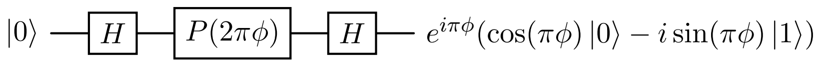
Another important two-qubit quantum gate is the SWAP gate,

$$\text{SWAP} \doteq \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \longrightarrow \tag{2.1.7}$$



which does exactly what you expect, it swaps qubits. The right-hand side above is the conventional circuit notation for the SWAP-gate. This is a valuable tool in quantum computers where not all qubits can interact. The SWAP-gate can move qubits close to each other, such that the desired interaction between them can be applied, after which SWAP-gates can be applied to move them back into their original positions.

Two other important quantum mechanical properties to utilize when constructing efficient quantum algorithms are **quantum superposition and quantum interference**. These two concepts are neatly illustrated in the following short circuit.



Let us step through this circuit one step at a time. Firstly, we apply the Hadamard gate,

$$H|0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \tag{2.1.8}$$

which opens up a superposition of quantum states. Next, we prepare for quantum interference by changing the relative phase of the two basis states,

$$P(2\pi\phi)H|0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + e^{i2\pi\phi}|1\rangle). \tag{2.1.9}$$

Lastly, we do quantum interference by closing the superposition with a second Hadamard-gate,

$$HP(2\pi\phi)H|0\rangle = e^{i\pi\phi}(\cos(\pi\phi)|0\rangle - i\sin(\pi\phi)|1\rangle). \tag{2.1.10}$$

A measurement of this qubit in the computational basis will always yield a 0 or a 1. By varying the phase $\phi \in [0, 1]$, we can vary the probability of obtaining either one. In fact, by choosing $\phi = \frac{1}{2}$, destructive interference ensures that there is 0% probability of obtaining $|0\rangle$ and constructive interference ensures that there is 100% probability of obtaining $|1\rangle$. Those familiar with quantum optics will recognize that this quantum circuit is equivalent to a Mach-Zehnder interferometer [19].

Ideally, we would want to be able to implement any unitary operator on a quantum computer. For this to be experimentally feasible, we need to be able to decompose any unitary operator into a finite set of unitary operators that we can perform using our given experimental setup. Luckily, this is possible! A

universal gate set is a set of gates $\mathcal{U} = \{U_1, U_2, U_3, ...\}$, finite or infinite, that can be used to efficiently implement any unitary gate to arbitrary precision by using a finite number of gates from $\mathcal{U}$. A classical example of a universal gate set is,

$$\{\text{CNOT}, H, S, T\} \tag{2.1.11}$$

where $S = P(\frac{\pi}{2})$ is unimaginatively called the S-gate. In this language, a quantum computer is said to be universal if it can implement a universal gate set.

2.2 Quantum Algorithms

As is stated in the previous section, good quantum algorithms should utilize quantum superposition, quantum interference, and entanglement. Using such quantum phenomena, which are pretty unusual compared to the classical phenomena we are accustomed to observing, to construct quantum algorithms is a challenging task. Specifically, setting up constructive interference for the solution to a given problem and destructive interference for all undesired results, without knowing the solution in advance, is what makes quantum algorithm design hard. Despite this, there exist several cleverly designed quantum algorithms, some of which I mentioned earlier: Deutsch's algorithm, Grover's algorithm, Shor's algorithm, and most importantly for this thesis, the quantum phase estimation algorithm (QPE) [17].

2.2.1 Exact Quantum Phase Estimation

The quantum phase estimation algorithm is at the center of this thesis, and it thus deserves a thorough walkthrough. The natural starting point is to discuss the problem it sets out to solve.

> **Problem:** Given an oracle that implements a controlled U-gate alongside an eigenstate $|u\rangle$ of U with eigenvalue $e^{i2\pi\phi}$, where we are promised that $\phi = \frac{m}{2^n},\ m, n \in \mathbb{N}$. Determine ϕ using the fewest possible queries to the oracle.

This is what we will call exact quantum phase estimation, since we will be able to deduce ϕ exactly. If $\phi \neq \frac{m}{2^n}$, then we cannot obtain it exactly, but only approximately. This is what we will refer to as approximate quantum phase estimation, which will be discussed in Subsec. 2.2.3. For now, the first thing to note is that if $m = m' + 2^n$ then,

$$e^{i2\pi\frac{m}{2^n}} = e^{i2\pi\frac{m'+2^n}{2^n}} = e^{i2\pi\frac{m'}{2^n}} e^{i2\pi} = e^{i2\pi\frac{m'}{2^n}}. \tag{2.2.1}$$

So we might as well assume that $m = 0, 1, \ldots, 2^n - 1$, which enables us to write it in a binary expansion:

$$m = \sum_{i=1}^{n} m_i 2^{n-i}, \quad m_i \in \{0, 1\}. \tag{2.2.2}$$

This reduces the problem to determining m_1, m_2, ..., m_n. Before doing so, let us consider a special case where $\phi = 0, \frac{1}{2}$. In the previous section, we used the quantum circuit analog of a Mach-Zehnder interferometer to exemplify quantum superposition and quantum interference:

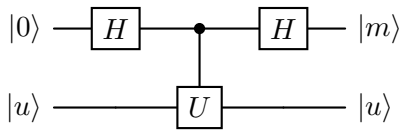
$$|0\rangle \text{ —} \boxed{H} \text{—} \boxed{P(2\pi\phi)} \text{—} \boxed{H} \text{—} \; e^{i\pi\phi}(\cos(\pi\phi)\,|0\rangle - i\sin(\pi\phi)\,|1\rangle)$$

Here we are guaranteed to measure 0 if $\phi = 0$ and guaranteed to measure 1 if $\phi = \frac{1}{2}$. This all sounds very

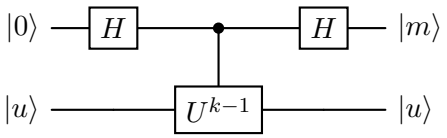
promising, but we are using a Phase-gate $P(2\pi\phi)$ and not our oracle U. To fix this, note that applying the Phase-gate $P(2\pi\phi)$ in the circuit above is the same as applying a controlled U-gate on $(H|0\rangle)\otimes|u\rangle$,

$$\begin{aligned} CU(H|0\rangle)\otimes|u\rangle &= (|0\rangle\langle 0|\otimes\mathbb{I}+|1\rangle\langle 1|\otimes U)\frac{1}{\sqrt{2}}(|0\rangle+|1\rangle)\otimes|u\rangle \\ &= \frac{1}{\sqrt{2}}(|0\rangle+e^{i2\pi\phi}|1\rangle)\otimes|u\rangle \end{aligned} \tag{2.2.3}$$

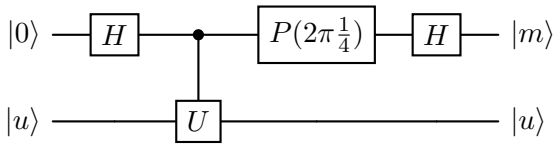
The first register is in the same state as if we had applied a Phase-gate! This effect is called phase kickback, since the phase we apply to the target state $|u\rangle$ is pulled back to the control state $|1\rangle$. In circuit notation:



This solves the special case, since if $\phi=0$ then $m=0$ and if $\phi=\frac{1}{2}$ then $m=1$. A quick generalization of this special case is the one where $\phi=0,\frac{1}{2^k}$ with k an integer. This is quickly reduced to the previous special case by noting that $|u\rangle$ is an eigenstate of U^{k-1} with eigenvalue $e^{i2\pi(2^{k-1}\phi)}$ where $2^{k-1}\phi=0,\frac{1}{2}$. So the circuit that solves this generalization to our special case is:



A last generalization to our special case is to perform a translation of $\phi\to\phi-\frac{1}{4}$ such that $\phi=-\frac{1}{4},\frac{1}{4}$. This reduces to the first special case by preceding the controlled U-gate with a Phase-gate $P(2\pi\frac{1}{4})$, i.e.:



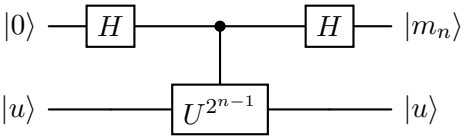
This reduces everything to the first special case, since $|1\rangle$ accumulates an overall phase of the form:

$$e^{i2\pi\phi}e^{i2\pi\frac{1}{4}}=e^{i2\pi(\phi+\frac{1}{4})} \tag{2.2.4}$$

where $\phi+\frac{1}{4}=0,\frac{1}{2}$. The entire structure of the quantum phase estimation algorithm follows from these simple considerations! Let us see how this unfolds by going back to the original problem of determining $m_1, m_2, \ldots, m_n$. The first thing we note is that,

$$e^{2^{n-1}(i2\pi\phi)}=e^{2^{n-1}(i\frac{\pi}{2^{n-1}}\sum_{i=1}^{n}m_i2^{n-i})}=e^{i2\pi\frac{m_n}{2}+i\pi\sum_{i=1}^{n-1}2^{n-i}}=e^{i2\pi\frac{m_n}{2}}. \tag{2.2.5}$$

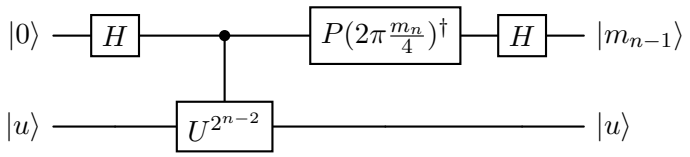
Following our discussion above, we see immediately that the following circuit gives us m_n.



We could try to obtain m_{n-1} naively by using a controlled $U^{2^{n-2}}$-gate, which would produce the following phase on $|1\rangle$,

$$e^{2^{n-2}(i2\pi\phi)} = e^{2^{n-2}(i\frac{\pi}{2^{n-1}}\sum_{i=1}^{n} m_i 2^{n-i})} = e^{i2\pi\frac{m_n}{4}+i2\pi\frac{m_{n-1}}{2}+i\frac{\pi}{2}\sum_{i=1}^{n-2}2^{n-i}} = e^{i2\pi\frac{m_n}{4}+i2\pi\frac{m_{n-1}}{2}}. \quad (2.2.6)$$

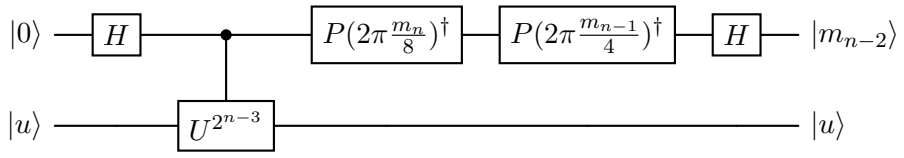
This is almost produces the phase we want $\frac{m_{n-1}}{2}$ except for the annoying term $\frac{m_n}{4}$. We already determined m_n. So by doing a phase shift $P(-2\pi\frac{m_n}{4}) = P(2\pi\frac{m_n}{4})^\dagger$ we can find m_{n-1}. This is exactly the same idea as in the special case concerning translations. The resulting circuit is visualized below:



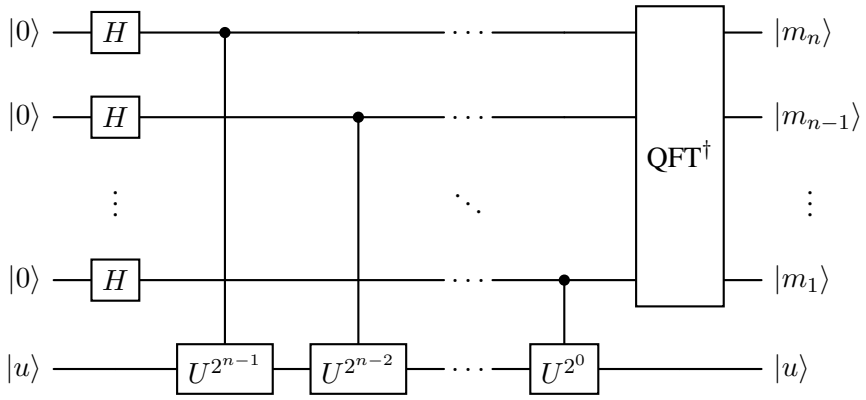
A pattern starts to emerge, but just for the sake of clarity, let us find m_{n-2}. Following the same idea we use a controlled $U^{2^{n-3}}$-gate, which produces a phase of the form:

$$e^{2^{n-3}(i2\pi\phi)} = e^{i2\pi\frac{m_n}{8}+i2\pi\frac{m_{n-1}}{4}+i2\pi\frac{m_{n-2}}{2}}. \quad (2.2.7)$$

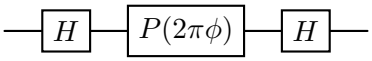
Here we already know m_n and m_{n-1}, which can be used to perform the appropriate phase shifts, giving us m_{n-2} as follows:



By iteratively following this procedure, we eventually find all the m_i's. This iterative solution to our problem can be formulated in a single algorithm, which is what we call the quantum phase estimation algorithm:



Here QFT† is called the inverse quantum Fourier transform (QFT), which is responsible for performing the appropriate phase shifts and closing the superpositions with Hadamard-gates. Note here that we follow the same convention as the Qiskit SDK, by letting the top-most qubit be the least significant one. This algorithm may appear complicated, but at its core, it is not significantly different from the circuit we initially started with.



Firstly, it prepares a superposition of quantum states using Hadarmard-gates, secondly, it prepares for quantum interference by changing relative phases, and lastly, it performs quantum interference by closing the superposition using QFT†.

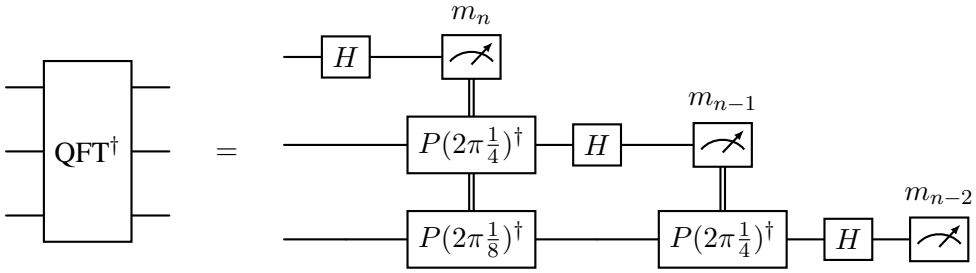
The great question is then, does the quantum phase estimation algorithm provide a quantum advantage over classical algorithms for determining eigenvalues of unitaries U given an eigenstate $|u\rangle$? Let us see, U is a $2^n \times 2^n$ matrix, so classically we naively need to do $(2^n)^3$ operations, in order to compute $U|u\rangle$ and find $e^{i2\pi\phi}$. In the quantum phase estimation algorithm, we need n Hadamard-gates, some amount of gates to implement the CU^{2^k}-gates, and $O(n^2)$ gates for QFT† [18]. The Big-O notation $O(n^2)$ means that the number of gates, as a function of n, includes terms scaling at most quadratically in n; which implies that we obtain an exponential speedup in n! At least if we have an efficient way of implementing the CU^{2^k}-gates. If we only have access to the oracle U, then we would need to call it,

$$2^{n-1} + 2^{n-2} + \cdots + 2 = 2^n - 1 \tag{2.2.8}$$

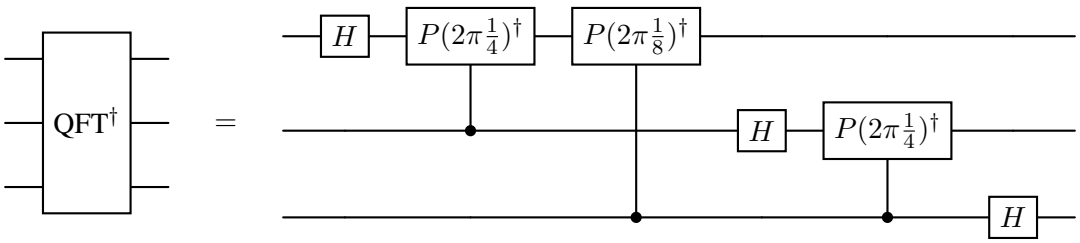
many times, which scales exponentially in n, and the speedup is lost.

2.2.2 The Quantum Fourier Transform

It is now time to investigate the inverse quantum Fourier transform, which was introduced in the quantum phase estimation algorithm in the previous subsection. The quantum circuit for QFT† on $n = 3$ qubits is given below:



A word on notation. The meter symbol represents a measurement in the computational basis, and the double lines connecting it to Phase-gates represent that the Phase-gates are conditioned on the classical measurement outcome m_i, i.e., do it if $m_i = 1$, and do nothing if $m_i = 0$. By utilizing the phase kickback effect, we can void these measurements altogether [18]. Let us visualize this on $n = 3$ qubits:



The n-qubit generalization of QFT† is not too hard to construct, and is contained as a standard gate

in most quantum software development kits like Qiskit. By explicitly computing matrix elements like $\langle x|\,\mathrm{QFT}^\dagger\,|y\rangle$ it can be shown that [18]:

$$\mathrm{QFT}^\dagger = \frac{1}{\sqrt{2^n}}\sum_{x,y=0}^{2^n-1} e^{-i2\pi\frac{xy}{N}}\,|x\rangle\langle y| \quad \Longrightarrow \quad \mathrm{QFT} = \frac{1}{\sqrt{2^n}}\sum_{x,y=0}^{2^n-1} e^{i2\pi\frac{xy}{N}}\,|y\rangle\langle x| \tag{2.2.9}$$

Note here that $x = 0, 1, \ldots, 2^n - 1$ is the integer representation of a bitstring $x_1, x_2, \ldots, x_n$ representing a n-qubit computational basis state $|x_n x_{n-1} \ldots x_1\rangle$. It is now very clear why it is called the quantum Fourier transform; it takes an input state $|\psi\rangle = \sum_{x=0}^{2^n-1} c_x\,|x\rangle$ and maps it to:

$$\mathrm{QFT}\,|\psi\rangle = \sum_{x=0}^{2^n-1}\Big(\frac{1}{\sqrt{2^n}}\sum_{y=0}^{2^n-1} c_x e^{i2\pi\frac{xy}{N}}\,|y\rangle\Big) = \sum_{y=0}^{2^n-1}\Big(\frac{1}{\sqrt{2^n}}\sum_{x=0}^{2^n-1} c_x e^{i2\pi\frac{xy}{N}}\Big)\,|y\rangle\,. \tag{2.2.10}$$

It thus encodes the Fourier components of the classical discrete Fourier transform of $(c_1, c_2, \ldots, c_{2^n-1})$, into the quantum amplitudes of a superposition of computational basis states. It does so using $O(n^2)$ gates, in contrast to the optimized version of the discrete Fourier transform, called the fast discrete Fourier transform, which would use $O(n2^n)$ operations [18]. So we get an exponential speedup! This is all fine, but the information we desire is encoded in the quantum amplitudes, which are inaccessible to experiments - all we can get from measurements are an observation like $01001\ldots001$, so we need to do exponentially many measurements to obtain the amplitudes, and the speedup is lost. Even so, the quantum Fourier transform is an essential subroutine in many quantum algorithms, just as we saw for quantum phase estimation.

2.2.3 Approximate Quantum Phase Estimation

We are now ready to discuss approximate quantum phase estimation, where $\phi \neq \frac{m}{2^n}$. An obvious approach would be to use the algorithm we derived for exact quantum phase estimation, but where $\phi \neq \frac{m}{2^n}$. Doing so yields the following output state [18]:

$$\mathrm{QPE}(|0\rangle^{\otimes n} \otimes |u\rangle) = \Big(\frac{1}{2^n}\sum_{x,y=0}^{2^n-1} e^{i2\pi x(\phi-\frac{y}{2^n})}\,|y\rangle\Big)\otimes|u\rangle\,. \tag{2.2.11}$$

The bottom register containing the eigenstate $|u\rangle$ is of no further relevance to us, and will thus be traced out. The probability of measuring $y = m$, is given as:

$$p(m) = \Big|\frac{1}{2^n}\sum_{x=0}^{2^n-1} e^{i2\pi x(\phi-\frac{m}{2^n})}\Big|^2 \overset{\text{finite geo. series}}{=} \begin{cases} \frac{1}{2^{2n}}\big|\frac{1-e^{i2\pi 2^n(\phi-\frac{m}{2^n})}}{1-e^{i2\pi(\phi-\frac{m}{2^n})}}\big|^2, & \text{if } \phi \neq \frac{m}{2^n} \\ 1, & \text{if } \phi = \frac{m}{2^n} \end{cases}. \tag{2.2.12}$$

Our results are thus compatible with what we got for exact quantum phase estimation; if $\phi = \frac{m}{2^n}$ then we are guaranteed to get m upon measurement. Now define $\delta_n = \phi - \frac{m}{2^n}$ and remember $\sin(x) = \frac{e^{ix}-e^{-ix}}{2i}$.

$$p(m) = \frac{1}{2^{2n}}\Big|\frac{1-e^{i2\pi 2^n(\phi-\frac{m}{2^n})}}{1-e^{i2\pi(\phi-\frac{m}{2^n})}}\Big|^2 = \frac{1}{2^{2n}}\frac{\sin^2(\pi 2^n\delta_n)}{\sin^2(\pi\delta_n)}. \tag{2.2.13}$$

We are no longer guaranteed to get ϕ exactly, but instead we get an approximation $\frac{m}{2^n}$ to ϕ upon measurement. There will always be an optimal approximation $\phi_{\text{opt.}} = \frac{m_{\text{opt.}}}{2^n}$ that minimizes the distance to our input ϕ,

$$\phi_{\text{opt.}} = \frac{1}{2^n}\min_{m\in\{0,1,\ldots,2^n-1\}}|\phi - \frac{m}{2^n}|. \tag{2.2.14}$$

In Appendix A, we find that this optimal approximation is indeed the most probable one! In more precise terms:

$$p(m_{\text{opt.}}) > p(m), \quad m \neq m_{\text{opt.}}. \tag{2.2.15}$$

So, naively using our derived algorithm from exact quantum phase estimation provides us with the optimal result $\phi_{\text{opt.}}$, with the largest probability among all possible outcomes.

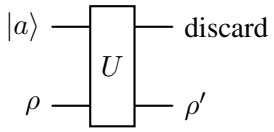
2.3 Quantum Channels

Consider an atomistic physical realization of a qubit into an excited state $|e\rangle = |1\rangle$ and a ground state $|g\rangle = |0\rangle$. Given a finite lifetime, any initial state of our atom,

$$|\psi\rangle = \alpha\,|g\rangle + \beta\,|e\rangle \tag{2.3.1}$$

will slowly decay into the ground state $|g\rangle$, loosing information about the amplitudes α and β in the process. This seems to contradict our previous assumption that time evolution in quantum mechanics is unitary! Well, not really; the information just escaped into the environment. So, if we consider our atom and everything it interacts with, which we refer to as the environment, then time evolution is unitary. We only observe non-unitary evolution because we are not seeing the full picture.

To be more precise, the state of any quantum system can always be specified by a density matrix ρ. Assume now that our system ρ is initially not entangled with the environment. By possibly enlarging the environment using a method called purification, we can assume that it is instead described by a pure state $|a\rangle$ [17]. Even though ρ might not undergo unitary evolution, the system plus environment $|a\rangle\langle a| \otimes \rho$ will undergo unitary evolution U, after which, we can trace out the environment, leaving us with the non-unitary evolution of our system $\rho \to \rho'$ [18]:



It is in this sense that all non-unitary evolution can be derived from the unitary evolution of a larger system. Such a map from density operators ρ to density operators ρ', that arises as the unitary evolution of some larger system, is called a **quantum channel**. They represent the admissible non-unitary, as well as unitary, evolutions of quantum systems, and are mathematically given as completely positive and trace-preserving maps [18].

Quantum channels are an invaluable tool when describing the possible effects of noise on quantum computers, such as the presence of unwanted interactions between qubits and the loss of information to the environment. The **depolarization channel** on an n-qubit state ρ provides an important example,

$$\rho \to (1-\lambda)\rho + \frac{\lambda}{2^n}\mathbb{I}, \tag{2.3.2}$$

where the complete positivity requirement restricts the allowed values of λ. Two important examples of one-qubit quantum channels are the **amplitude damping channel**, which describes energy dissipation

such as atomic decays, and the **phase damping channel**, which describes loss of information without energy dissipation, such as the loss of phase information in a photonic state as it scatters randomly through a waveguide. These two processes are described respectively by the times T_1 and T_2, which are the characteristic timescales on which amplitude and phase information is lost [17]. The combined effect of such processes is also referred to as the **thermal relaxation** quantum channel.

Before concluding this section, we will discuss why noisy quantum hardware tends to produce uniformly distributed outcomes independent of the specific algorithm or input. Intuitively, such a statement seems reasonable, since noise destroys structure, which leads to decoherence and an increasing uncertainty of outcomes. The precise reason it happens is rooted in the concept of entropy from thermodynamics, specifically the Von Neumann entropy $S(\rho)$ of a quantum state ρ, which is defined as:

$$S(\rho) = -\text{Tr}(\rho \log \rho). \tag{2.3.3}$$

Pure states $\rho = |\psi\rangle \langle\psi|$ minimize the Von Neumann entropy, since, in contrast to mixed states, we know the quantum state $|\psi\rangle$ of the system with certainty. The d-dimensional maximally mixed state $\rho = \frac{1}{d}\mathbb{I}$ maximizes the Von Neumann entropy with a value of $S(\rho) = \log(d)$, and it does so uniquely [17], i.e ,

$$S(\rho) = \log(d) \quad \iff \quad \rho = \frac{1}{d}\mathbb{I}. \tag{2.3.4}$$

Many physically relevant noise processes — such as depolarizing and dephasing, described above — correspond to unital quantum channels, meaning they preserve the identity operator. Unital quantum channels $\mathcal{E}(\rho)$ tend to increase the Von Neumann entropy [17],

$$S(\mathcal{E}(\rho)) \geq S(\rho). \tag{2.3.5}$$

An exception is amplitude damping, which tends to lower the Von Neumann entropy, as is readily seen in the atomic case, where it moves the qubit towards its pure ground state $|g\rangle$. The statement in the equation above captures the intuition that noise tends to destroy coherence and increase the randomness of a quantum system. As the level of unital noise increases, the system evolves towards the maximally mixed state, i.e., a state of maximal entropy. This state assigns equal probability to all computational basis states. Consequently, when a quantum computer is subject to strong unital noise, the output state will approach the maximally mixed state, which results in a uniform distribution of outcomes, independent of the specific algorithm or input.

2.4 Quantum Hardware

This section provides a brief overview of the entire quantum computing stack and discusses the guiding principles used in constructing physical platforms for qubits, the fundamental building blocks of universal gate-based quantum computers.

2.4.1 Guiding Principles

A natural question to ask is: "What are the experimental requirements needed to build a universal quantum computing device?". The fundamental building blocks are qubits, i.e., two-level quantum systems. To

realize a quantum computer, it is not enough to give some robust physical representation of qubits; one must choose a representation in which the qubits can be evolved unitarily in any desired way. This immediately poses a problem: To be robust, qubits must be shielded from the environment, but to be able to evolve them unitarily, the qubits must be accessible so that they can be manipulated into performing the desired unitary evolutions. A good physical implementation of qubits must strike a delicate balance between these two opposing constraints.

To implement arbitrary quantum algorithms, it is necessary to be able to implement a universal set of unitary transformations. Note that isolated qubits are closed quantum systems that evolve unitarily according to their Hamiltonian. To control them, one must be able to control their Hamiltonian to a degree that allows for the implementation of a universal set of unitary transformations. This leads to a complication, as the control system is also a quantum system; therefore, the full Hamiltonian of the qubits and the control system should include the backactions of the qubits on the control system. Think of how photons in cavity QED [19] can, after interacting with an atom, carry away information about the state of the atom. This leads to decoherence of the qubits.

To predict the outcome of algorithms, preparing fiducial initial states is necessary, e.g., aligning magnetic spins or cooling atoms into their ground state. There are two critical factors determining the quality with which this can be done: The fidelity with which the initial state can be prepared, and the entropy of the initial state. Entropy is essential, since it is easy to prepare the maximally mixed state $\rho = \frac{1}{2^n}\mathbb{I}$ on n-qubits - nature does it for us, decoherence and thermal noise increases the entropy of a system, pushing it towards the maximally mixed state [17]. Still, this state is invariant under unitary transformations and is thus of very little interest to us. Ideally, we would want a pure state as the initial state, i.e., a state with zero entropy.

Lastly, the outcome of a quantum computation is a superposition of quantum states, which, if designed carefully, returns the desired result with high probability upon measurement of the qubits. Therefore, to retrieve information, measurements are necessary, and the quality with which this is done sets a natural boundary on the usefulness of the quantum computation.

To summarize, the experimental requirements needed to build universal gate-based quantum computers are the following:

1. Robustly represent quantum information
2. Implement a universal set of unitary transformations
3. Prepare a fiducial initial state
4. Measure the output state

Examples of physical realizations of qubits that are constructed according to these guiding principles include superconducting qubits [2], trapped ions [4, 5], neutral atoms [20, 21], photonics [22], and many more. Each architecture has its pros and cons regarding gate speed, gate fidelity, coherence time, and scalability [23], e.g., superconducting qubits generally have faster gate times than trapped ions. In comparison, the latter have far greater coherence times.

2.4.2 The Quantum Computing Stack

As quantum computers mature, they evolve from being physics experiments to fully developed computing stacks, ready for the public to use. The structure of a quantum computing stack is usually conceptualized as layers, where each layer serves as an abstraction over the one below it [24].

- **Application Layer:** This is the top of the abstraction layers, and consists of domain-specific and hardware-agnostic algorithms and applications in areas such as quantum chemistry, cryptography, and machine learning [25].
- **Software and Runtime Layer:** Below the application layer sits the programming environment and runtime system. The framework Qiskit [26], widely used in this thesis, is an example that allows users to write high-level quantum algorithms and manage execution on hardware over cloud-based systems. Such tools are also responsible for hybrid classical-quantum workflows and error mitigation [24].
- **Quantum Compilation and Optimization Layer:** Abstract quantum algorithms must be translated into hardware native gates to be executable. This is achieved by quantum compilers, which perform circuit transpilation, optimization, and mapping of logical qubits in the abstract algorithm to physical qubits on the hardware [26].
- **Control and Readout Layer:** This layer bridges the quantum system, comprised of qubits, and the classical part of the quantum stack. It consists of control systems used to perform unitary transformations and measure qubits. In superconducting architectures, for example, it involves circuit quantum electrodynamics for state measurement [27].
- **Physical Hardware Layer:** This is the bottom layer of the stack, consisting of the chosen physical realizations of qubits. Different platforms include superconducting circuits, trapped ions, neutral atoms, and photonics, each with its pros and cons [23].

The benchmark constructed in this thesis will be a holistic, single-number metric measuring the computing power of an entire quantum computing stack, rather than just testing the properties of the hardware layer.

CHAPTER 3

Benchmarking

In quantum computing, benchmarking means measuring the quality of quantum computers. Good benchmarks should enable researchers and industrial users to understand the power of a quantum computer or an entire quantum computing stack. Bad benchmarks, on the other hand, will misdirect users and thus inhibit progress within the field of quantum computing.

Over the years, a wide variety of benchmarks have been developed. Some try to measure progress towards quantum utility, while other application-oriented benchmarks measure the problem-solving power of a given quantum computing stack. The former holds long-term significance, while the latter is more significant for near-term research and industrial users of NISQ devices.

A significant bottleneck for achieving quantum utility is the comparatively high error rates of logical gates on quantum computers compared to those on classical computers. Errors will thus, in the foreseeable future, dominate the quality and design of quantum computers. Most quantum computer benchmarks, therefore, measure the effects of those errors.

In this chapter, we will follow the review article *Benchmarking Quantum Computers* by Proctor et al. [10] closely. Similar to Proctor et al., we will only consider benchmarks of the primary quantum computing paradigm, i.e., gate-based quantum hardware.

3.1 Examples of Benchmarks

To truly appreciate the diversity of quantum computing benchmarks, a few examples must be digested. The examples considered, roughly speaking, fall into two categories: low-level and high-level benchmarks.

Low-level benchmarks quantify the effects of qubit and gate errors. A great example of this is a randomized benchmark that measures the **average gate fidelity**. Let us discuss this benchmark in depth.

Suppose noisy hardware implements a single-qubit quantum channel Λ that should implement something close to $V \in U(2)$, i.e. ideally Λ transforms the initial state $|0\rangle\langle 0|$ into $V|0\rangle\langle 0|V^\dagger$. The fidelity,

$$F(\Lambda, V) = \langle 0| V^\dagger \Lambda(|0\rangle\langle 0|) V |0\rangle \tag{3.1.1}$$

is a single-number metric that quantifies the performance of the quantum channel Λ. If $\Lambda(|0\rangle\langle 0|) = V|0\rangle\langle 0|V^\dagger$ then $F(\Lambda, V) = 1$, otherwise $0 \leq F(\Lambda, V) < 1$. What if the initial state was different from $|0\rangle$, say $|\psi\rangle$? Any $|\psi\rangle$ can be written as $|\psi\rangle = U|0\rangle$ for some $U \in U(2)$. By averaging over all $U \in U(2)$ we average over all initial states $|\psi\rangle$ and thus obtain the average fidelity:

$$\bar{F}(\Lambda, V) = \int_{U(2)} d\mu(U)(\langle 0|U^\dagger)V^\dagger\Lambda(U|0\rangle\langle 0|U^\dagger)V(U|0\rangle) \tag{3.1.2}$$

where $d\mu(U)$ is the unitarily invariant Haar measure on $U(2)$ [28]. In this form, $\bar{F}(\Lambda, V)$ is not obtainable through experiments since we would have to compute $\Lambda(U|0\rangle\langle 0|U^\dagger)$ for infinitely many unitaries U! Luckily, we are saved by a beautiful idea called unitary t-designs [29]. A unitary t-design is a finite subset $K \subset U(2)$, such that for any homogeneous polynomial $f(U)$ of degree t in the entries of U and U^* respectively:

$$\frac{1}{|K|}\sum_{U\in K} f(U) = \int_{U(2)} d\mu(U) f(U). \tag{3.1.3}$$

The integrand of $\bar{F}(\Lambda, V)$ is a homogeneous polynomial of degree $t = 2$ in the entries of U and U^*, respectively. So if we have a unitary 2-design K, then,

$$\bar{F}(\Lambda, V) = \frac{1}{|K|}\sum_{U\in K} (\langle 0|U^\dagger)V^\dagger\Lambda(U|0\rangle\langle 0|U^\dagger)V(U|0\rangle) \tag{3.1.4}$$

which resolves our problem! Now we only have to evaluate $\Lambda(U|0\rangle\langle 0|U^\dagger)$ for the finitely many $U \in K$. It turns out that the Clifford group C is a unitary 2-design [29], so we could choose $K = C$. This idea of averaging over a continuous parameter space using finitely many samples will reappear in Chapter 4.

A **high-level benchmark** quantifies the overall performance of an entire quantum computing stack. A great example is the widely used **quantum volume** [7]. Let us discuss this benchmark in depth.

The quantum volume is a randomized benchmark, i.e., it tests performance using randomized quantum circuits rather than circuits representing known useful algorithms, such as the quantum phase estimation algorithm. What does it measure? Quantum volume measures the largest square circuits that a given device can run reliably.

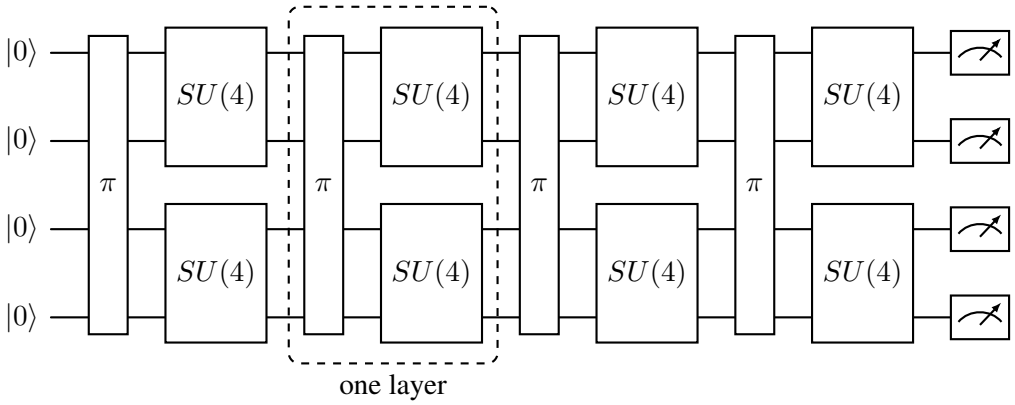


Figure 3.1: Quantum volume circuit. It consists of d-layers of permutation plus Haar-random $SO(4)$ gates on d qubits, in the special case $d = 4$. The $SO(4)$ gates are applied to the first $2\lfloor \frac{d}{2} \rfloor$ qubits in pairs, leaving the dth qubit idle. Note that the dth qubit for d odd still participates due to permutations π.

The idea is to define success criteria and run circuits as in Fig. 3.1 for increasing d until it fails. The largest d, which results in a success, will then be used to define the quantum volume as:

$$V_Q = 2^d. \tag{3.1.5}$$

This idea of formulating a benchmark in terms of success criteria will be revisited in Chapter 4. Now, the precise nature of the success criteria of quantum volume is unimportant to us. What is important to note is the following two things: Firstly, the success criteria are based on solving a problem called the *heavy output generation* problem [30], which has roots in proposals to prove quantum advantage [30]. Secondly, computing this success criterion requires classical simulation of the circuits in Fig. 3.1, which scales exponentially with d and is thus only viable for devices with a small to moderate number of qubits.

Another **high-level benchmark** is IonQ's **number of algorithmic qubits** (#AQ) [9]. The #AQ is an application-oriented benchmark in contrast to V_Q, which is a randomized benchmark. Therefore, instead of testing a device's ability to implement arbitrary unitary gates, application-oriented benchmarks assess a device's capacity to reliably implement unitary gates relevant to solving computationally significant problems. Let us discuss this benchmark in depth.

In essence, the idea of #AQ is to quantify the overall performance of hardware on a collection of valuable algorithms. These algorithms are collected into a suite C that may be updated over time to meet the needs of the quantum computing community. For each algorithm $c \in C$ and each number of qubits N, a specific choice of implementation is made, one that every user of the benchmark will use when measuring #AQ. The algorithm for measuring #AQ goes as follows [9]:

1. Each implementation for specific N and $c \in C$ will be decomposed using a compiler into R_x, R_y, R_z, and $CNOT$ gates. Let $d_c = \#CNOT$ be the number of $CNOT$ gates and $w_c = N$ be the number of qubits.

2. The success of each implementation is measured with the classical fidelity F_c against the ideal output distribution P_{ideal}:

$$F_c(P_{ideal}, P_{output}) = \left(\sum_x \sqrt{P_{output}(x) P_{ideal}(x)} \right)^2 \tag{3.1.6}$$

   where the output distribution P_{output} is measured on hardware with x each possible output.

3. The success of an implementation of $c \in C$ for specific N is defined to be:

$$F_c - \epsilon_c > t \tag{3.1.7}$$

   where $\epsilon_c = \sqrt{\frac{F_c(1-F_c)}{s_c}}$ is the statistical error based on s_c number of shots, and $t = \frac{1}{e} \simeq 0.37$.

4. Now, each $c \in C$ will be plotted on a volumetric background at (d_c, w_c). The #AQ is now the largest rectangle with width N and depth N^2 such that all the points inside correspond to circuits that ran successfully. See Fig. 3.2 for a concrete example provided by IonQ in their blog post [9].

For further details and information, see IonQ's description on their website [9]. Note that the number of two-qubit gates d_c shown in Fig. 3.2 and the number of two-qubit gates implemented on hardware can vary drastically, potentially making it seem like a device can implement a larger amount of two-qubit gates with high fidelity than it is capable of. This is a great flaw pointed out by Quantinuum [31].

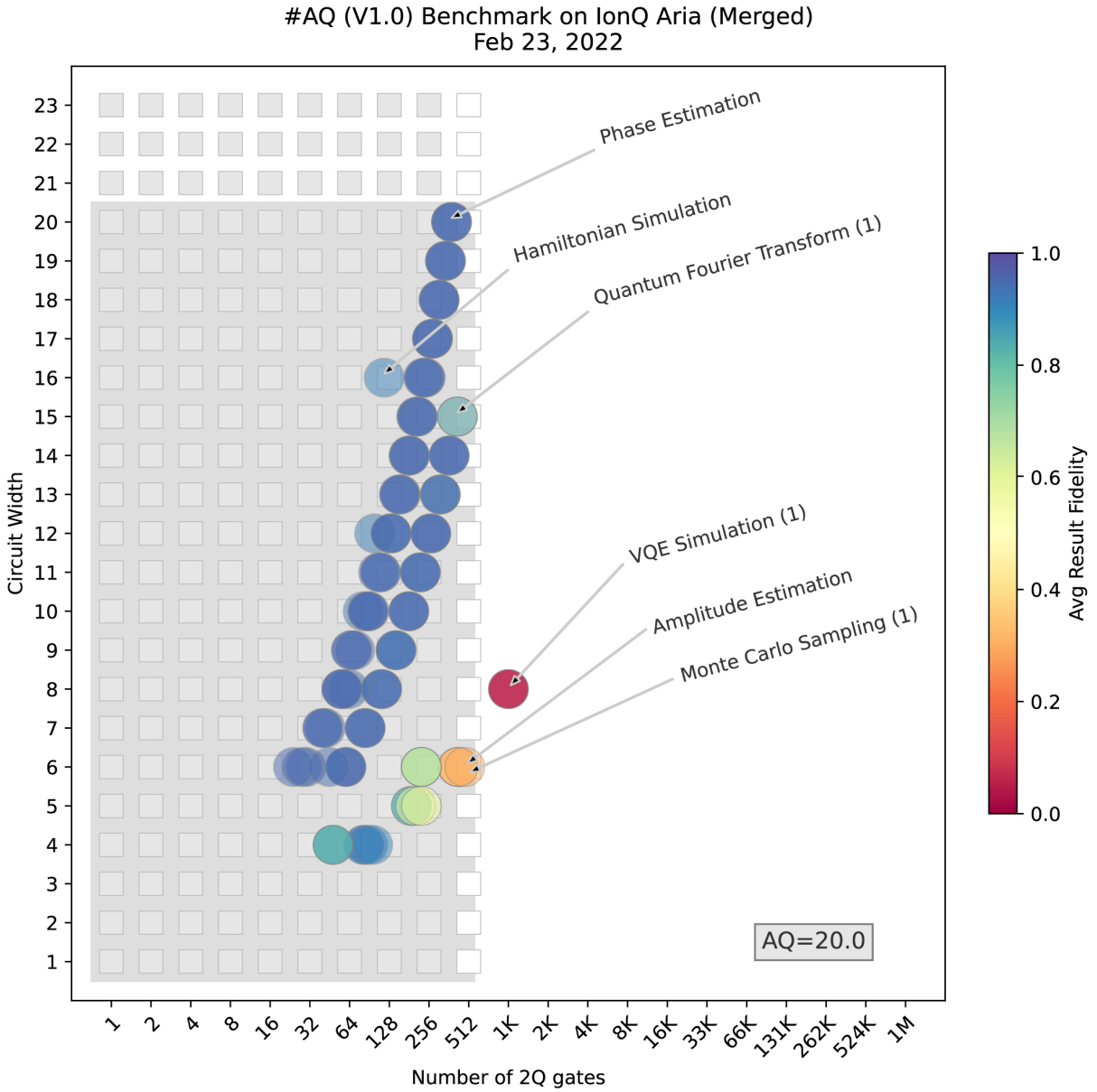


Figure 3.2: Measurement of #AQ on IonQ's Aria device. This picture is imported from IonQ's blog post [9]

3.2 The Ideal Benchmark

In light of these examples, a reasonable definition of a quantum computer benchmark is the following [10]: (1) A set of computational tasks along with a procedure to perform those tasks on a quantum computer, and (2) a method for computing one or more performance metrics from the classical data obtained by running those tasks. For most benchmarks, the tasks consist of one or more quantum circuits, and the output is a single-number metric, e.g., average gate fidelity or quantum volume. However, a benchmark can measure several single-number metrics or non-scalar quantities, e.g., volumetric benchmarks with plots as #AQ in Fig. 3.2.

Whether a benchmark is application-oriented, randomized, low-level, or high-level, Proctor et al. wrote down a set of properties that an ideal benchmark should have.

These are repeated here for future reference:

1. *Well-motivated*. A benchmark should measure well-motivated metrics of performance.
2. *Well-defined*. A benchmark should have an unambiguous procedure, i.e., any unspecified steps in that procedure should be intentionally configurable parameters
3. *Implementation-robust*. It should not be possible to exploit ("game") a benchmark's configurable parameters to obtain misleading results.
4. *System-robust*. A benchmark's results should not be corrupted when the tested quantum computer experiences a priori unknown (small) errors.
5. *Efficient*. A benchmark should use a reasonable amount of all resources (e.g., quantum and classical computer time).
6. *Technology independent*. A benchmark should specialize to particular technologies or architectures only inasmuch as its metrics are only relevant in those contexts.

Now, these properties are not binary, and most present benchmarks do not score highly on all of them. For example, quantum volume requires inefficient classical computation of heavy output frequencies and thus scores lowly on efficiency in property (5). In Chapter 4, these properties will be used as guidelines when defining and working on the effective qubit number.

3.3 The Future of Benchmarking

It is well understood that qubit count alone is not sufficient to understand a quantum computer's performance. Even so, qubit count is still among the first things mentioned when comparing quantum computers. It is easy to understand why; it is easy to read, share, and compare. Even so, it is very misleading, as it does not properly explain either problem-solving power or progress towards quantum utility. A significant problem is thus to construct single-number metrics that, unlike qubit count, accurately capture the problem-solving power of a quantum computer, as well as progress toward quantum utility.

There has been an increasing focus on application-oriented benchmarks such as IonQ's #AQ [9], as opposed to randomized benchmarks such as IBM's quantum volume [7]. The reason is that, out of all the infinitely many unitary evolutions a quantum computer can implement, only a small subset of them correspond to solutions to practically relevant problems. For near-term quantum computers, an important, practically relevant question is: "What can this quantum computer do?". Application-oriented benchmarks produce metrics that directly measure a quantum computer's problem-solving power, making it easy to answer such questions. The effective qubit number is a single-number application-oriented metric and thus follows both of the above trends.

CHAPTER 4

The Effective Qubit Number

In this chapter, we develop an algorithmic benchmark for universal gate-based quantum computers based on the quantum phase estimation algorithm (QPE). The quantum phase estimation algorithm provides, under certain assumptions, an exponential speedup over classical algorithms for computing eigenvalues. It is thus a proxy for quantum advantage, making it an ideal candidate for algorithmic benchmarks. Our benchmark will be a single-number metric, which takes a holistic approach to quantum computers, ideally making it as easy to understand, share, and compare as the qubit count.

The benchmark will be referred to as the effective qubit number. The main idea is to choose a specific test implementation of quantum phase estimation, for which the effective qubit number is the largest number of qubits that a device can run on, before noise "takes over" in the system. Formulating in a precise mathematical way, when noise "takes over", will be the main difficulty. The reason is that quantum mechanics is probabilistic, so even if we have a perfect noise-free quantum computer, we still have statistical noise affecting our results. This statistical noise is intrinsic to quantum mechanics and thus not a feature of the imperfections of the quantum device under consideration. We therefore want to ensure that when we measure that noise has "taken over", it is the imperfections of the quantum device we are measuring, i.e., qubit decoherence, gate errors, crosstalk, etc., and not just statistical noise.

Throughout this section, simulations will be used to test concepts and ideas. These simulations are all performed using the Qiskit SDK. A custom noise model will be used to imitate the behavior of real quantum computers. The same noise model will be used throughout the entire thesis. This noise model naively imitates the behavior of IBM's Sherbrooke superconducting QPU. It is made up of 10 fully-connected qubits with thermal relaxation times T_1 and T_2 equal to the corresponding average values over all qubits of the Sherbrooke device. The basis gates are the same as the Sherbrooke device, and they have the same implementation times. Gate errors are modeled naively using depolarization channels, one for each basis gate, each with error probability p equal to the average error rate of the corresponding gate on the Sherbrooke device. Consider Appendix B for a more thorough discussion.

4.1 Reviewing Previous Ideas

A previous master's student from Aarhus University, Ebbe Maru Storm, also worked on developing a single-number metric based on quantum phase estimation [32] under the supervision of Niels Jakob

Søe Loft from Kvantify and Dmitri Fedorov from Aarhus University. In this section, we briefly discuss Ebbe's ideas. To begin with, he chose, just as we will, the simplest test implementation of quantum phase estimation, where $U = P(2\pi\phi)$ and $|u\rangle = |1\rangle$. The choice $U = P(2\pi\phi)$ has the property that,

$$(P(2\pi\phi))^{2^k} = P(2\pi(2^k\phi)). \tag{4.1.1}$$

This ensures that we can effectively implement the middle part of the quantum phase estimation algorithm. The chosen implementation is visualized in Fig. 4.1.

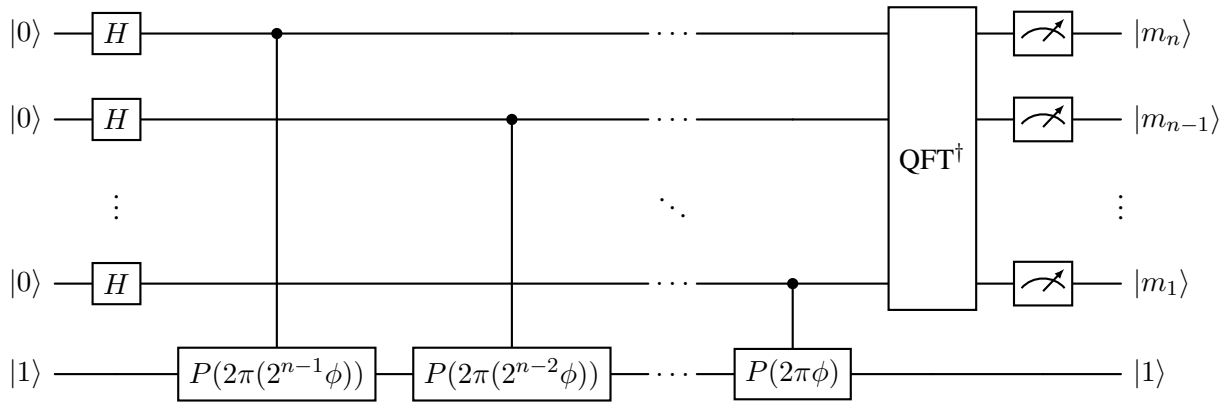


Figure 4.1: The test implementation of quantum phase estimation that Ebbe Maru Storm used in his thesis, with $U = P(2\pi\phi)$ and $|u\rangle = |1\rangle$. It will be the same one we use throughout this thesis.

Every time we run the circuit in Fig. 4.1 with a choice of phase ϕ in $P(2\pi\phi)$, we get as output a phase approximation $\frac{m}{2^n}$ of our input phase ϕ, where,

$$m = \sum_{i=1}^{n} m_i 2^{n-i}. \tag{4.1.2}$$

Following the notation of Chapter 2, we let $p(m)$ denote the theoretical probability of obtaining output m given input ϕ to the quantum phase estimation algorithm. The probability $p(m_{\text{opt}})$ of obtaining the best approximation possible on n qubits, $\phi_{\text{opt.}} = \frac{m_{\text{opt.}}}{2^n}$, is in general not equal to 1. So motivated by the standard result from the literature [17],

$$p(m_{\text{opt.}}) \geq \frac{4}{\pi^2} \approx 0.41, \tag{4.1.3}$$

Ebbe chose to make a phase estimate based on $n_s = 100$ QPE-calls, as the most frequently occurring output. Since $p(m_{\text{opt.}}) \geq \frac{4}{\pi^2}$ this estimate is very likely to be $\phi_{\text{opt.}} = \frac{m_{\text{opt.}}}{2^n}$. Let $n(m) =$ *# outputs equal to m*, then after n_s shots we get an empirical probability distribution,

$$\hat{p}(m, n_s) = \frac{n(m)}{n_s}. \tag{4.1.4}$$

The phase estimate can then be defined more precisely as:

$$\phi_{est.}(n) = \frac{1}{2^n} \underset{m}{\text{argmax}}\{\hat{p}(m, n_s = 100)\}. \tag{4.1.5}$$

This is all under the assumption that we have a well-defined maxima. As we will discuss in the Sec. 4.3, this assumption makes a lot of sense theoretically. Even so, given a finite number of shots $n_s < \infty$, we

are statistically never guaranteed to get a well-defined maximum. If you obtain several maxima, choose one of them.

The idea behind Ebbe's definition of the effective qubit number n_{eff} goes as follows: The best obtainable prediction error $|\phi - \phi_{\text{est.}}(n)|$ from quantum phase estimation on n qubits decreases with n, but the more qubits we use, the more gates we need. Gates are noisy, so as we increase the number of qubits n, we accumulate more noise, which results in more uniformly distributed outcomes, which in turn results in larger errors $|\phi - \phi_{\text{est.}}(n)|$. Based on this, he argued that it would seem reasonable that there should exist some n for which this error is minimal. This number $n = n_{\text{eff}}^{\text{Ebbe}}$ is what he called the effective qubit number, which is the number of qubits, where you on the choice of test implementation in Fig. 4.1 get the best numerical accuracy, i.e., the smallest prediction error. The idea is illustrated in Fig. 4.2.

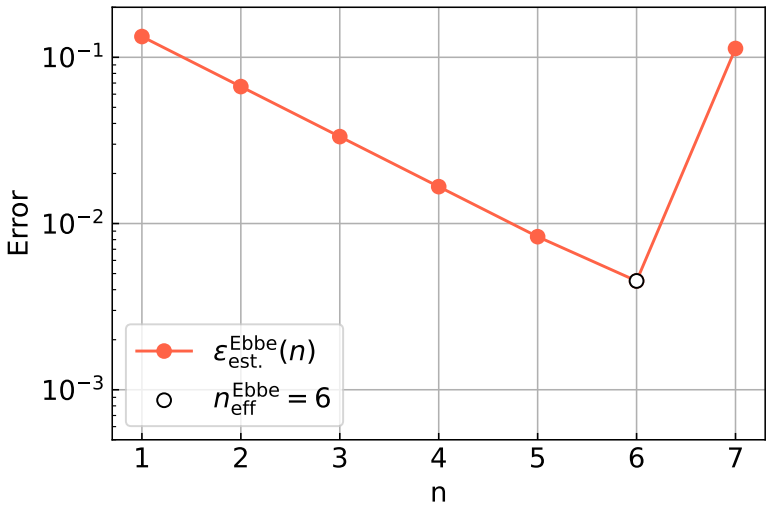


Figure 4.2: Illustration of Ebbe's idea. Experimentally obtained prediction error is plotted versus qubit count n. The effective qubit number $n_{\text{eff}}^{\text{Ebbe}}$ is the number of qubits giving the best numerical accuracy, i.e., the smallest prediction error. The prediction error is computed according to Eq. 4.1.7 defined below using data from simulations on our custom noise model. The data gathered from the simulations are phase estimates as given in Eq. 4.1.5 for all $\phi \in \Phi^{\text{Ebbe}}$ and all appropriate qubit counts n.

The implementation in Fig. 4.1 introduces a free variable ϕ. Some representatives $\Phi^{\text{Ebbe}} \subset [0, 1]$ for this variable must be chosen, which Ebbe did uniformly:

$$\Phi^{\text{Ebbe}} = \{0.1, 0.2, 0.3, \ldots, 0.9\}. \tag{4.1.6}$$

Instead of computing the prediction error $|\phi - \phi_{\text{est.}}(n)|$ for one $\phi \in \Phi^{\text{Ebbe}}$, he computed an average, in the hope of obtaining more stable results:

$$\epsilon_{\text{est.}}^{\text{Ebbe}}(n) = \frac{1}{|\Phi|} \sum_{\phi \in \Phi} |\phi - \phi_{\text{est.}}(n)|. \tag{4.1.7}$$

In terms of this, he defined the effective qubit number as:

$$n_{\text{eff}}^{\text{Ebbe}} = \underset{n}{\text{argmin}}\{\epsilon_{\text{est.}}^{\text{Ebbe}}(n)\}, \tag{4.1.8}$$

which assumes the existence of a well-defined minimum. To avoid future confusion, all definitions made by Ebbe that will differ from ours later on have been marked with a superscript *Ebbe*.

4.2 Analyzing Previous Ideas

Let us take Ebbe's definition of the effective qubit number and compare it to the six criteria (1)-(6) in the definition of an ideal benchmark given in Sec. 3.2. To make certain points, simulations of his effective qubit number will be made using our custom noise model. The data gathered from simulations are phase estimates as given in Eq. 4.1.5 for all $\phi \in \Phi$ and appropriate n, from which the effective qubit number can be computed according to Eq. 4.1.8.

Let us begin with point (1). The effective qubit number in Eq. 4.1.8 is the number of qubits on which the device obtains the best numerical accuracy on a test implementation of quantum phase estimation. This is undoubtedly a well-motivated metric of performance. Even so, the choice of phase representatives in Eq. 4.1.6 used in defining the effective qubit number is less well-motivated.

Let us discuss points (2) and (3). By choosing the phase representatives in Eq. 4.1.6, well-motivated or not, he eliminated the last degree of freedom in the benchmark. This ensures that it is implementation-robust. It is, unfortunately, not entirely well-defined. To see why, consider Fig. 4.3. This figure contains the same data as in Fig. 4.2, except for one extra data point at $n = 8$ qubits. Obtaining this new data point changes our prior belief that $n_{\text{eff}}^{\text{Ebbe}} = 6$ to $n_{\text{eff}}^{\text{Ebbe}} = 8$. How do we know that this will not happen again for some larger qubit count n?

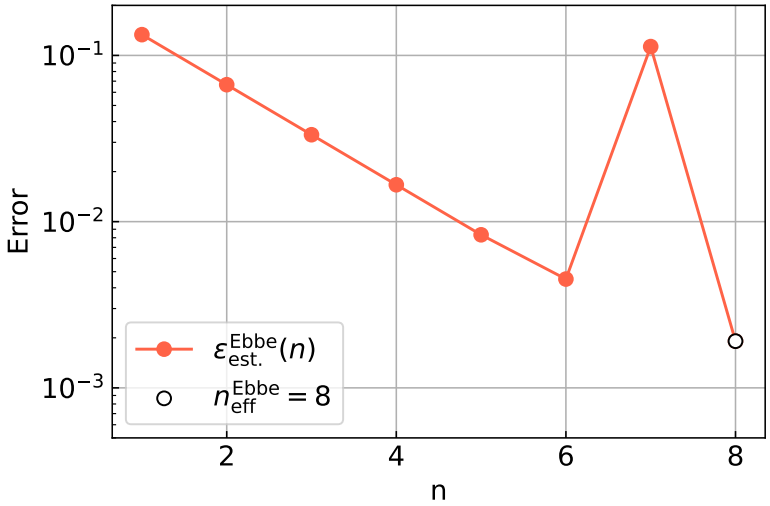


Figure 4.3: Average prediction error over Φ as given in Eq. 4.1.7 versus qubit count n. The data is gathered from simulations using our custom noise model, just as in Fig. 4.2. The data is the same as in Fig. 4.2, except for the extra data point at $n = 8$ qubits.

A naive way to try to circumvent this problem goes as follows: The effective qubit number should be 6 since the average prediction error $\epsilon_{\text{est.}}^{\text{Ebbe}}(n)$ jumps rapidly for $n = 7, 8$ qubits, which seems to be a clear sign of noise taking over, so the effective qubit number should not be 8. The issue with this is that it either requires visual inspection, which introduces an unwanted bias in the definition, or a precise mathematical definition of "jumping rapidly". Unless the problem is solved, it seems that the effective qubit number as defined by Ebbe is not entirely well-defined.

Let us discuss point (4). A natural way to test system-robustness is to gather simulation data and compute $n_{\text{eff}}^{\text{Ebbe}}$ as we did in Fig. 4.3, say five more times. The resulting data is visualized in Fig. 4.4.

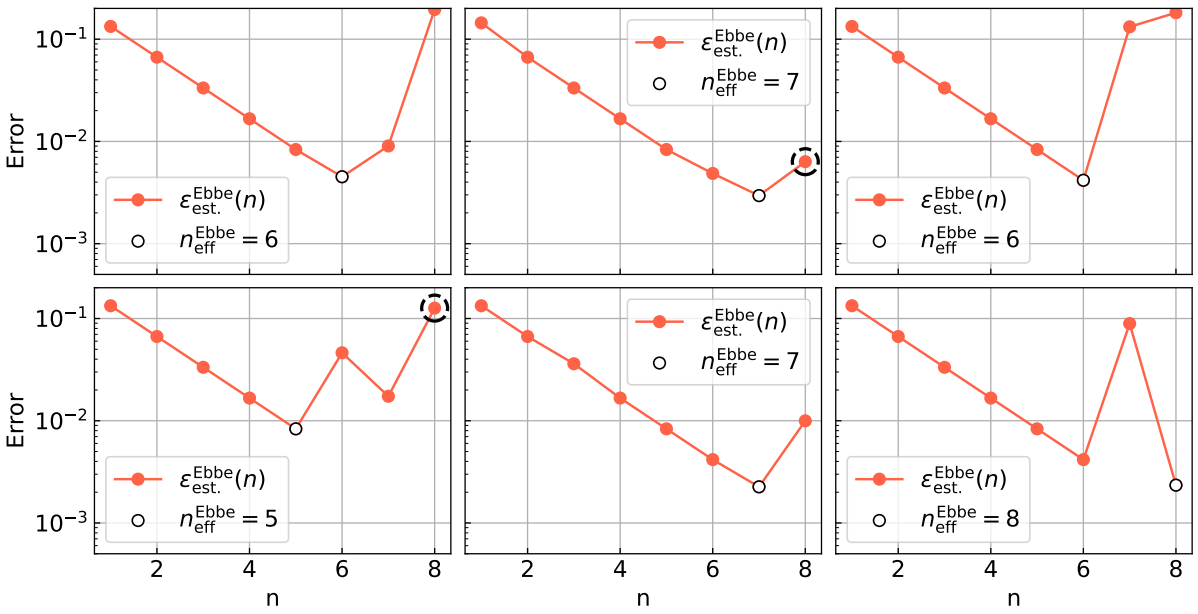


Figure 4.4: Average prediction error over Φ as given in Eq. 4.1.7 versus qubit count n for six independent simulations using the custom noise model. The circled data points are used to illustrate a point regarding system-robustness versus implementation-robustness in the paragraphs below.

By inspecting Fig. 4.4, we find effective qubit numbers ranging from 5 to 8 on the same noise model! This clearly illustrates that the effective qubit number as defined by Ebbe is not system-robust. Let $\epsilon_{\text{est.}}^{\text{Ebbe}}(n, i)$ for $i = 1, 2, \ldots, d$ be the average errors corresponding to each of the $d = 6$ simulations in Fig. 4.4 labeled from left to right and top to bottom and let $n_{\text{eff}}^{\text{Ebbe}}(i)$ be the corresponding effective qubit numbers. Then, a naive approach to solving the system-robustness problem would be to redefine our effective qubit number to:

$$n_{\text{eff}}^{\text{Mean}} = \frac{1}{d} \sum_{i=1}^{d} n_{\text{eff}}^{\text{Ebbe}}(i) \in \mathbb{R}. \tag{4.2.1}$$

The value of this redefinition, together with the original effective qubit numbers of the $d = 6$ simulations in Fig. 4.4, is shown in Fig. 4.5.

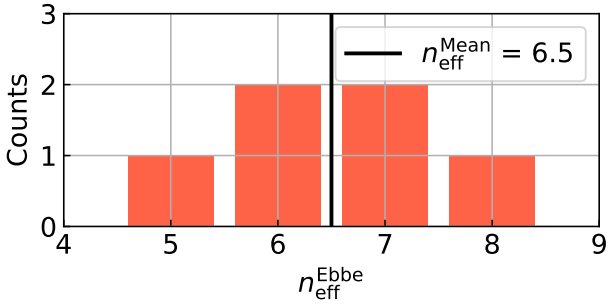


Figure 4.5: Visual representation of the effective qubit numbers of the simulations in Fig. 4.4 together with the value of the naive redefinition of the effective qubit number given in Eq. 4.2.1.

Instead of an integer, the redefinition of the effective qubit is a real number ranging from 1 to one less

than the maximum number of qubits on the device - remember we have one qubit reserved for the eigenstate in the bottom register of the test implementation of quantum phase estimation in Fig. 4.1. It seems reasonable to expect that as we gather more data $n_{\text{eff}}^{\text{Mean}}$ will converge towards some stable value, solving our system-robustness problem. There is just one problem: this comes at the cost of losing implementation-robustness. To see why, note that each effective qubit number $n_{\text{eff}}^{\text{Ebbe}}(i)$ is computed from a sequence of prediction errors,

$$\{\epsilon_{\text{est.}}^{\text{Ebbe}}(n=1,i),\ \epsilon_{\text{est.}}^{\text{Ebbe}}(n=2,i),\ \ldots,\ \epsilon_{\text{est.}}^{\text{Ebbe}}(n=8,i)\}, \tag{4.2.2}$$

according to Eq. 4.1.7. As mentioned previously, we label the graphs in Fig. 4.4 from left to right and top to bottom, starting from $i=1$. Now consider the sequence of prediction errors at $i=2$ and $i=4$, i.e. the graphs with circled data points at $n=8$,

$$\begin{gathered}\{\epsilon_{\text{est.}}^{\text{Ebbe}}(n=1,i=2),\ \epsilon_{\text{est.}}^{\text{Ebbe}}(n=2,i=2),\ \ldots,\ \epsilon_{\text{est.}}^{\text{Ebbe}}(n=8,i=2)\} \overset{\text{Eq. 4.1.8}}{\longrightarrow} n_{\text{eff}}^{\text{Ebbe}}(i=2)=7 \\ \text{and} \\ \{\epsilon_{\text{est.}}^{\text{Ebbe}}(n=1,i=4),\ \epsilon_{\text{est.}}^{\text{Ebbe}}(n=2,i=4),\ \ldots,\ \epsilon_{\text{est.}}^{\text{Ebbe}}(n=8,i=4)\} \overset{\text{Eq. 4.1.8}}{\longrightarrow} n_{\text{eff}}^{\text{Ebbe}}(i=4)=5.\end{gathered} \tag{4.2.3}$$

Each of the prediction errors $\epsilon_{\text{est.}}^{\text{Ebbe}}(n,i)$ for different n and i are computed from phase estimates $\phi_{\text{est.}}(n,i)$ that are sampled independently. So we might as well by chance have gotten the following sequences of data instead, which leads to better effective qubit numbers,

$$\begin{gathered}\{\epsilon_{\text{est.}}^{\text{Ebbe}}(n=1,i=2),\ \epsilon_{\text{est.}}^{\text{Ebbe}}(n=2,i=2),\ \ldots,\ \epsilon_{\text{est.}}^{\text{Ebbe}}(n=8,i=4)\} \overset{\text{Eq. 4.1.8}}{\longrightarrow} n_{\text{eff}}^{\text{Ebbe}}(i=2)=7 \\ \text{and} \\ \{\epsilon_{\text{est.}}^{\text{Ebbe}}(n=1,i=4),\ \epsilon_{\text{est.}}^{\text{Ebbe}}(n=2,i=4),\ \ldots,\ \epsilon_{\text{est.}}^{\text{Ebbe}}(n=8,i=2)\} \overset{\text{Eq. 4.1.8}}{\longrightarrow} n_{\text{eff}}^{\text{Ebbe}}(i=4)=8.\end{gathered} \tag{4.2.4}$$

In other words, there is no way for people to know whether we have swapped the data points.

$$\epsilon_{\text{est.}}^{\text{Ebbe}}(n=8,i=2) \quad \longleftrightarrow \quad \epsilon_{\text{est.}}^{\text{Ebbe}}(n=8,i=4). \tag{4.2.5}$$

Since doing the swap leads to better effective qubit numbers individually, it also improves our naive redefinition in Eq. 4.2.1, which is visualized in Fig. 4.6.

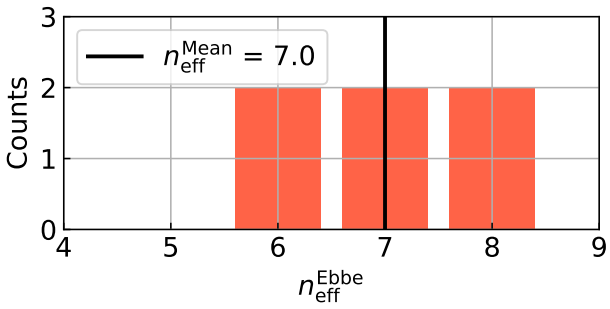


Figure 4.6: Same representation of the effective qubit numbers of the simulations as in Fig. 4.5, but after the swap of data points described in Eq. 4.2.4. The swapped data points are marked with circles in Fig. 4.4.

This swap illustrates that it is possible to game the naive extension of Ebbe's effective qubit number given in Eq. 4.2.1. From this redefinition, we have gained system-robustness at the cost of implementation-robustness, which suggests that a better alternative approach may be needed.

Lastly, let us discuss points (5) and (6). The test implementation of quantum phase estimation in Fig. 4.1 that Ebbe's effective qubit number is based on, is efficient with respect to gate count due to Eq. 4.1.1. Since the benchmark includes no inefficient classical computation, the benchmark as a whole is efficient. It is also technology-independent in the sense that it is defined for all universal gate-based quantum computers. Since quantum phase estimation can be implemented on any universal quantum computer, such as cluster-state quantum computers mentioned briefly in Chapter 2, the benchmark should also be well-defined for such devices. Even so, it is not quite obvious what the effective qubit number would correspond to on such cluster-state quantum computers.

4.3 Setting a Path Forward

Based on the discussion in Sec. 4.2, we will set a path forward for the development of a new improved algorithmic benchmark based on quantum phase estimation. This definition naturally draws inspiration from the work described in Sec. 4.1 in ways that will be explained in this section. As in Sec. 4.2, we will often refer back to the individual criteria (1)-(6) in the definition of an ideal benchmark given in Sec. 3.2 to ensure that it is always clear which properties we are trying to improve upon.

Just as Ebbe's benchmark, ours will also be called the effective qubit number, n_{eff}. It will also be defined on the test implementation of the quantum phase estimation algorithm given in Fig. 4.1. This guarantees that our benchmark will satisfy point (6), i.e., efficiency, due to Eq. 4.1.1, as long as we do not include any inefficient classical computations in the definition. We will also use the phase estimate given in Eq. 4.1.5, based on $n_s = 100$ QPE-calls on n qubits with input phase ϕ. Note that n here and throughout the rest of the thesis refers to the number of qubits in the upper register of the test implementation of quantum phase estimation in Fig. 4.1, excluding the qubit used to store the eigenstate $|1\rangle$. The phase estimate is included here for convenience,

$$\phi_{est.}(n) = \frac{1}{2^n}\underset{m}{\text{argmax}}\{\hat{p}(m, n_s = 100)\}, \tag{4.3.1}$$

where $\hat{p}(m, n_s)$ is the empirical probability distribution constructed according to Eq. 4.1.4 from the classical data obtained from the $n_s = 100$ QPE-calls. In Appendix A we proved that

$$p(m_{\text{opt.}}) > p(m), \quad m \neq m_{\text{opt.}} \tag{4.3.2}$$

which immediately implies that,

$$m_{\text{opt.}} = \underset{m}{\text{argmax}}\{p(m)\}. \tag{4.3.3}$$

In absence of noise the empirical probability distribution $\hat{p}(m, n_s)$ should approach the theoretical probability distribution $p(m)$ in the limit $n_s \to \infty$, i.e.,

$$p(m) = \lim_{n_s \to \infty} \hat{p}(m, n_s). \tag{4.3.4}$$

This ensures that our phase estimate $\phi_{\text{est.}}(n)$ is equal to the optimal phase approximation $\phi_{\text{opt.}}(n) = \frac{m_{\text{opt.}}}{2^n}$ in the absence of noise and the limit $n_s \to \infty$, which is the main motivation for choosing exactly this phase estimate. It is worth noting that if $\phi = \frac{1}{2}(\frac{m}{2^n} + \frac{m+1}{2^n})$, then the two phase approximations $\frac{m}{2^n}$ and

$\frac{m+1}{2^n}$, that ϕ lies in the middle of, are both optimal phase approximations in the sense that they minimize the distance $|\phi - \frac{m}{2^n}|$, and according to Appendix A they are equally probable,

$$p(m) = p(m+1), \quad \text{if } \phi = \frac{1}{2}(\frac{m}{2^n} + \frac{m+1}{2^n}). \tag{4.3.5}$$

We thus have that the optimal phase approximation is always the most probable; however, in some instances, there are not only one optimal phase approximation, but two.

The effective qubit number n_{eff} will be defined in terms of numerical accuracy $|\phi - \phi_{\text{est.}}(n)|$, just as Ebbe's effective qubit number, instead of the classical choice of fidelity. The reason is that, in practical applications, numerical accuracy is often the primary performance metric of interest, making it a natural quantity to optimize. To use this performance metric, a choice $\Phi \subset [0,1]$ for the possible values of the free variable ϕ must be made. Ebbe did so uniformly according to Eq. 4.1.6. It is natural to ask whether this choice is representative of all $\phi \in [0,1]$, or if a more well-motivated choice exists? Answering this question would make the benchmark more well-motivated as required in point (1). Ideally, we would want to be able to compute the prediction error $|\phi - \phi_{\text{est.}}(n)|$ for all $\phi \in [0,1]$ and average the results,

$$\int_0^1 |\phi - \phi_{\text{est.}}(n)| d\phi. \tag{4.3.6}$$

This way, all choices of the free variable $\phi \in [0,1]$ are represented equally, but we cannot do an infinite number of experiments, so this is clearly not possible. This same issue appears when discussing average gate fidelity in Sec. 3.1, where it is expressed as an integral over $U(2)$ with respect to the Haar measure $d\mu(U)$,

$$\bar{F}(\Lambda, V) = \int_{U(2)} d\mu(U) (\langle 0| U^\dagger) V^\dagger \Lambda(U |0\rangle \langle 0| U^\dagger) V (U |0\rangle). \tag{4.3.7}$$

It turns out, as is discussed in that section, that using the concept of unitary t-designs, this integral is expressible as a finite sum over the Clifford group C,

$$\bar{F}(\Lambda, V) = \frac{1}{|C|} \sum_{U \in C} (\langle 0| U^\dagger) V^\dagger \Lambda(U |0\rangle \langle 0| U^\dagger) V (U |0\rangle). \tag{4.3.8}$$

To ensure that our benchmark is technology-independent, we aim to avoid making assumptions about the underlying noise model of the hardware. We will thus most likely not be able to prove something similar for the integral $\int_0^1 |\phi - \phi_{\text{est.}}(n)| d\phi$, but we might be able to prove something in absence of noise in the limit $n_s \to \infty$ where we are guaranteed that $\phi_{\text{est.}}(n) = \phi_{\text{opt.}}(n)$. In this special case, the average of the prediction error over all $\phi \in [0,1]$ is given as,

$$\epsilon(n) := \int_0^1 \epsilon(n, \phi) d\phi, \quad \epsilon(n, \phi) = |\phi - \phi_{\text{opt.}}(n)|. \tag{4.3.9}$$

Motivated by the result for the average gate fidelity, we seek in this special case a finite set of phase representatives $\Phi \subset [0,1]$ such that,

$$\epsilon(n) = \frac{1}{|\Phi|} \sum_{\phi \in \Phi} w(\phi) \epsilon(n, \phi). \tag{4.3.10}$$

Note that we have allowed for different phases ϕ to have different positive weights $w(\phi) > 0$. Such a set Φ of phase representatives is surely a more well-motivated choice for the values of the free variable

$\phi \in [0, 1]$ than the uniform choice made by Ebbe. Finding such a set will be the first goal in creating a better algorithmic benchmark based on quantum phase estimation, which is successfully done in Sec. 4.4.

The effective qubit number will be defined as the largest number of qubits on which we can run our test implementation of quantum phase estimation before noise "takes over". Specifically, we want to make sure that when we measure that noise has "taken over", we are measuring the quantum device's imperfections, not just the statistical noise due to the probabilistic nature of quantum mechanics. We will approach this problem by choosing n_s large enough that the phase estimate,

$$\phi_{est.}(n) = \frac{1}{2^n} \underset{m}{\text{argmax}}\{\hat{p}(m, n_s)\}, \tag{4.3.11}$$

is equal to $\phi_{\text{opt.}}(n)$ in the absence of noise with very large probability, since this would ensure that,

$$\epsilon_{\text{est.}}(n) := \frac{1}{|\Phi|} \sum_{\phi \in \Phi} w(\phi) |\phi - \phi_{\text{est.}}(n)| = \frac{1}{|\Phi|} \sum_{\phi \in \Phi} w(\phi) |\phi - \phi_{\text{opt.}}(n)| = \epsilon(n), \tag{4.3.12}$$

with still a very large, but slightly smaller, probability. If we measure that $\epsilon_{\text{est.}}(n) \neq \epsilon(n)$ then we can be very confident that it is due to imperfections of the quantum device, and not statistical noise, since such deviation is very unlikely to occur in the absence of noise. Does this go the other way around, i.e., does the presence of noise imply increased probability of observing such deviations $\epsilon_{\text{est.}}(n) \neq \epsilon(n)$? In general, this seems to be the case, the reason being that as noise increases in quantum computers, they tend to produce more uniformly distributed outcomes as discussed in Sec. 2.3, independent of the specific algorithm or input. This leads to phase estimate variations $\phi_{\text{est.}}(n) \neq \phi_{\text{opt.}}(n)$ being increasingly likely and thus to larger probabilities of observing deviations $\epsilon_{\text{est.}}(n) \neq \epsilon(n)$. We can thus use $\epsilon(n)$ as a reference for detecting the presence of noise in the system. In Sec. 4.5, it is shown that for the phase representatives $\phi \in \Phi$ found in Sec. 4.4, $n_s = 100$ shots is enough to guarantee that $\phi_{\text{est.}}(n) = \phi_{\text{opt.}}(n)$ with very high probability in the absence of noise, as desired.

To construct a system-robust benchmark as required by point (4), the definition of the effective qubit number as defined by Ebbe in Eq. 4.1.8 needs some work, since his definition as the minimum of the prediction error turned out not to be system-robust. Once again, we will draw inspiration from another well-known benchmark, the quantum volume, which is discussed in Sec. 3.1 in more depth. What matters right now is that quantum volume is defined in terms of a success criterion for each square circuit, as two to the power of the size of the largest square circuit for which the success criterion is satisfied. Inspired by this, we seek to define the effective qubit number in terms of a success criterion,

$$S_n(D) \in \{0, 1\} \tag{4.3.13}$$

for each size n of the test implementation in Fig. 4.1, where D is the collection of phase estimates for $\phi \in \Phi$, each computed according to Eq. 4.1.5 a certain number of times. The convention will be that $S_n(D) = 0$ corresponds to failure, and $S_n(D) = 1$ corresponds to success, respectively. The effective qubit number n_{eff} will then be defined as the largest n for which $S_n(D) = 1$. The main problem will then be to define the success criterion $S_n(D)$ such that the first n at which $S_n(D) = 0$ corresponds exactly to the n at which noise "takes over". Furthermore, this necessitates a precise mathematical definition of noise "taking over". A possible solution to this problem is provided in Sec. 4.6.

To summarize the open problems discussed above, each problem and the section in which it is solved will be written down below:

(**Sec. 4.4**) Find a finite set of phase representatives $\Phi \subset [0, 1]$ that satisfies Eq. 4.3.10.

(**Sec. 4.5**) Find n_s large enough that $\phi_{\text{est.}}(n) = \phi_{\text{opt.}}(n)$ in the absence of noise with large probability.

(**Sec. 4.6**) Define a success criterion $S_n(D) \in \{0, 1\}$ that properly captures when noise "takes over".

(**Sec. 4.7**) Define n_{eff} using $S_n(D)$ in a precise algorithmic way.

4.4 Finding Phase Representatives

To find a finite set of phase representatives $\Phi \subset [0, 1]$ satisfying Eq. 4.3.10, we will first need to find an analytic expression for the integral,

$$\epsilon(n) = \int_0^1 \epsilon(n, \phi) d\phi, \quad \epsilon(n, \phi) = |\phi - \phi_{\text{opt.}}(n)|. \tag{4.4.1}$$

To do so, remember that the optimal phase approximation $\phi_{\text{opt.}}(n)$ is in most cases a unique fraction of the form $\frac{m_{\text{opt.}}}{2^n}$, but for specific $\phi = \frac{1}{2}(\frac{m}{2^n} + \frac{m+1}{2^n})$ it might take on two distinct values $\phi_{\text{opt}}(n) = \frac{m}{2^n}, \frac{m+1}{2^n}$, each of which results in the same distance $\epsilon(n, \phi)$. Since all we currently care about is the distance $\epsilon(n, \phi)$, we can always choose $\phi_{\text{opt.}}(n)$ to be the smallest of the two possibilities. Under these assumptions, we can write $\phi_{\text{opt.}}(n)$ for $\phi \in [\frac{m}{2^n}, \frac{m+1}{2^n}]$ as follows,

$$\phi_{\text{opt.}}(n) = \begin{cases} \frac{m}{2^n}, & \text{if } \phi \in [\frac{m}{2^n}, \frac{1}{2}(\frac{m}{2^n} + \frac{m+1}{2^n})] \\ \frac{m+1}{2^n}, & \text{if } \phi \in (\frac{1}{2}(\frac{m}{2^n} + \frac{m+1}{2^n}), \frac{m+1}{2^n}] \end{cases}. \tag{4.4.2}$$

From this, it is evident that the distance $\epsilon(n, \phi)$ is 0 at $\phi = \frac{m}{2^n}$, it increases linearly until its maximum at $\phi = \frac{1}{2}(\frac{m}{2^n} + \frac{m+1}{2^n})$, after which it decreases linearly until it reaches 0 at $\phi = \frac{m+1}{2^n}$. This behavior repeats for all the 2^n possible values of m, resulting in a sawtooth shape, as is visualized in Fig. 4.7 for $n = 2$.

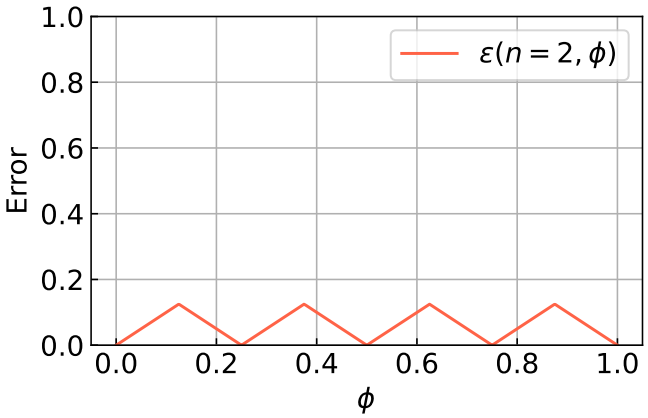


Figure 4.7: Distance $\epsilon(n = 2, \phi) = |\phi - \phi_{\text{opt.}}(n = 2)|$ between input phase ϕ and the optimal phase approximation $\phi_{\text{opt.}}(n = 2)$ on $n = 2$ qubits as a function of ϕ.

The sawtooth shape is made up of 2^n triangles. The width of each triangle is $\frac{m+1}{2^n} - \frac{m}{2^n} = \frac{1}{2^n}$ and its height is $|\frac{1}{2}(\frac{m}{2^n} + \frac{m+1}{2^n}) - \frac{m}{2^n}| = \frac{1}{2}\frac{1}{2^n}$, resulting in a total area of each triangle of,

$$\text{area of triangle} = \frac{1}{2} \cdot (\frac{1}{2}\frac{1}{2^n}) \cdot (\frac{1}{2^n}) = \frac{1}{4}\frac{1}{2^{2n}}. \tag{4.4.3}$$

The entire area under the distance function $\epsilon(n, \phi)$ for $\phi \in [0, 1]$ is then equal to,

$$\epsilon(n) = \int_0^1 \epsilon(n, \phi) d\phi = (\#\text{ triangles}) \cdot (\text{area of triangle}) = 2^n \cdot \frac{1}{4} \frac{1}{2^{2n}} = \frac{1}{2^{n+2}}. \tag{4.4.4}$$

This is the analytical expression we are looking for! The key thing to note is that $\epsilon(n+1) = \frac{1}{2}\epsilon(n)$, i.e., the average prediction error $\epsilon(n, \phi)$ over all phases $\phi \in [0, 1]$ halves every time we increase the qubit count n by one.

Finding a finite set $\Phi \subset [0, 1]$ that satisfies Eq. 4.3.10 is by Eq. 4.4.4 equivalent to finding one that satisfies the following equation,

$$\frac{1}{|\Phi|} \sum_{\phi \in \Phi} w(\phi)\epsilon(n, \phi) = \frac{1}{2^{n+2}}. \tag{4.4.5}$$

To find such Φ, the behavior of the prediction errors $\epsilon(n, \phi)$ as a function of n for different choices of ϕ needs to be examined. This is done in Fig. 4.8 for three such choices $\phi \in \{\frac{1}{6}, \frac{3}{5}, \frac{8}{9}\}$. Alongside these, the analytical expression $\epsilon(n) = \frac{1}{2^{n+2}}$ is plotted. In Fig. 4.8 we observe that the behavior of $\epsilon(n, \phi)$,

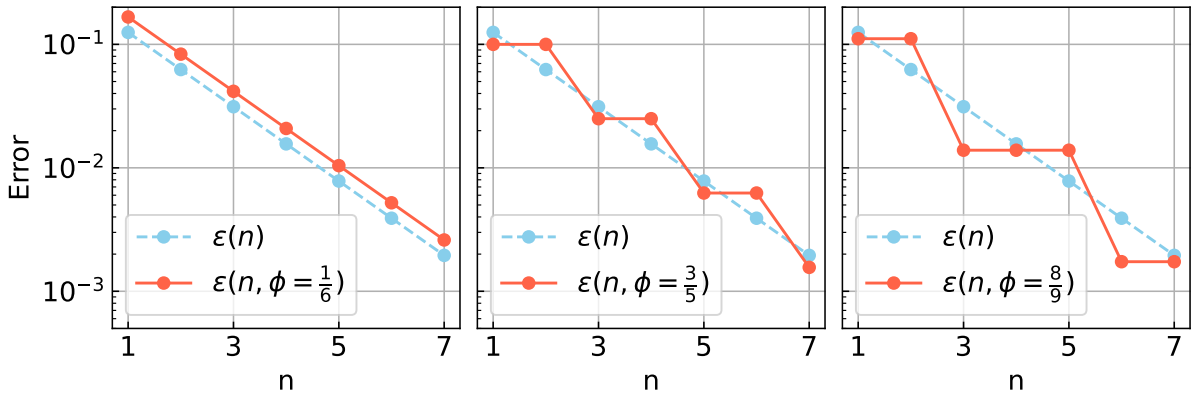


Figure 4.8: Numerical examination of the behavior of the distance function $\epsilon(n, \phi)$ as a function of n for different choices of $\phi \in \{\frac{1}{6}, \frac{3}{5}, \frac{8}{9}\}$.

as a function of n on a logarithmic scale, ranges from straight lines to stair-like functions. The former is of most interest to us, as it follows the behavior of $\epsilon(n)$ up to a translation, which on a regular scale corresponds to a scaling, i.e.,

$$\epsilon(n) = \alpha\epsilon(n, \phi = \frac{1}{6}), \quad n \in \{1, 2, \ldots, 7\} \tag{4.4.6}$$

The scaling factor can numerically be determined to be $\alpha = \frac{3}{4}$. Now it seems that $\Phi = \{\frac{1}{6}\}$ solves our problem immediately with $w(\phi = \frac{1}{6}) = \alpha$, at least for $n \in \{1, 2, \ldots, 7\}$, but in the hope of getting more stable results, we would like to find more phases ϕ, that just like $\phi = \frac{1}{6}$, satisfies that,

$$\epsilon(n) = w(\phi)\epsilon(n, \phi). \tag{4.4.7}$$

The reason being that if each $\phi \in \Phi$ satisfies Eq. 4.4.7, then Φ satisfies Eq. 4.4.5:

$$\frac{1}{|\Phi|} \sum_{\phi \in \Phi} w(\phi)\epsilon(n, \phi) = \frac{1}{|\Phi|} \sum_{\phi \in \Phi} \epsilon(n) = \epsilon(n) = \frac{1}{2^{n+2}}. \tag{4.4.8}$$

The next step needed to find more phases ϕ satisfying Eq. 4.4.7 is to consider the following equivalence,

$$\epsilon(n) = w(\phi)\epsilon(n,\phi) \quad \Longleftrightarrow \quad \epsilon(n+1,\phi) = \frac{1}{2}\epsilon(n,\phi) \neq 0. \tag{4.4.9}$$

The implication $\Longrightarrow$ is obvious, while the other implication $\Longleftarrow$ needs some explaining. By using $\epsilon(n+1,\phi) = \frac{1}{2}\epsilon(n,\phi)$ inductively it follows that:

$$\epsilon(n,\phi) = \frac{1}{2}\epsilon(n-1,\phi) = \frac{1}{4}\epsilon(n-2,\phi) = \cdots = \frac{1}{2^{n-1}}\epsilon(1,\phi). \tag{4.4.10}$$

From this, it immediately follows that,

$$\epsilon(n,\phi) = \frac{8}{2^{n+2}}\epsilon(1,\phi) \quad \Longrightarrow \quad \epsilon(n) = \frac{1}{8\epsilon(1,\phi)}\epsilon(n,\phi), \tag{4.4.11}$$

proving the desired equivalence. So finding a finite set $\Phi \subset [0,1]$ satisfying Eq. 4.3.10 is equivalent to finding a finite set $\Phi \subset [0,1]$ such that each phase $\phi \in \Phi$ satisfies that,

$$\epsilon(n+1,\phi) = \frac{1}{2}\epsilon(n,\phi). \tag{4.4.12}$$

Furthermore, if this equation is satisfied, then it immediately follows that the scaling factors $w(\phi)$ in Eq. 4.3.10 take the following form,

$$w(\phi) = \frac{1}{8\epsilon(1,\phi)}. \tag{4.4.13}$$

Solutions to the two equations above are found in the paragraph below.

For ϕ to satisfy Eq. 4.4.12 it is necessary to satisfy the following,

$$\epsilon(2,\phi) = \frac{1}{2}\epsilon(1,\phi), \tag{4.4.14}$$

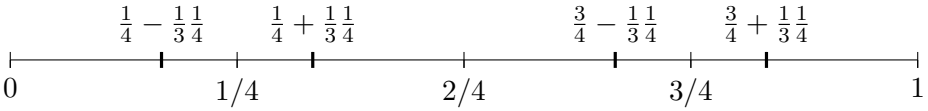
which also turns out to be sufficient, as we will see later. For now, we will find four $\phi \in [0,1]$ that satisfy the equation above. If we choose say $\phi = \frac{3}{5} = 0.6$ then,

$$\phi_{\text{opt.}}(n=1) = \frac{1}{2} = \frac{2}{4} = \phi_{\text{opt.}}(n=2) \quad \Longrightarrow \quad \epsilon(n=1,\phi=\frac{3}{5}) = \epsilon(n=2,\phi=\frac{3}{5}), \tag{4.4.15}$$

which gives rise to the stair-like behavior observed in Fig. 4.8, which is not what we want! So why did $\phi = \frac{3}{5}$ not work? To figure this out, consider the possible phase approximations on $n=1$ and $n=2$ qubits,

$$P_{\text{approx.}}(n=1) = \{0, \frac{1}{2}\} \quad \text{and} \quad P_{\text{approx.}}(n=2) = \{0, \frac{1}{4}, \frac{2}{4}, \frac{3}{4}\}. \tag{4.4.16}$$

Since $P_{\text{approx.}}(n=2)$ contains $P_{\text{approx.}}(n=1)$ there will be phases like $\phi = \frac{3}{5}$ that is so close to a common phase approximation $\frac{1}{2} \in P_{\text{approx.}}(n=1) \subset P_{\text{approx.}}(n=2)$ that $\frac{1}{2}$ is optimal on both $n=1$ and $n=2$ qubits, giving rise to the stair-like behavior. We thus need to choose ϕ close to the the phase approximations in $P_{\text{approx.}}(n=2)$ not contained in $P_{\text{approx.}}(n=1)$, i.e., $\frac{1}{4}$ and $\frac{3}{4}$. The exact distance is determined by Eq. 4.4.14 to be one-third of the distance between adjacent phase approximations, i.e., $\frac{1}{3}\frac{1}{4}$:



To see this, note that this distance ensures that Eq. 4.4.14 is satisfied,

$$\epsilon(1,\phi) = \frac{2}{3}\frac{1}{4} = 2\epsilon(2,\phi). \tag{4.4.17}$$

The resulting four phases $\phi \in [0,1]$ ends up being,

$$\Phi' = \{\frac{1}{6} = \frac{1}{4} - \frac{1}{3}\frac{1}{4},\ \ \frac{1}{3} = \frac{1}{4} + \frac{1}{3}\frac{1}{4},\ \ \frac{2}{3} = \frac{3}{4} - \frac{1}{3}\frac{1}{4},\ \ \frac{5}{6} = \frac{3}{4} + \frac{1}{3}\frac{1}{4}\}, \tag{4.4.18}$$

with equal scaling factors, since $\epsilon(1,\phi) = \frac{1}{6}$ for all $\phi \in \Phi'$,

$$w(\phi) = \frac{1}{8\epsilon(1,\phi)} = \frac{6}{8} = \frac{3}{4}. \tag{4.4.19}$$

This is great! We have rediscovered the phase $\phi = \frac{1}{6}$ with scaling factor $w(\phi) = \frac{3}{4}$ that we found accidentally in numerical simulations, but we still need to check that it is sufficient for $\phi \in \Phi'$ to satisfy Eq. 4.4.14 for it to satisfy Eq. 4.4.12. To do so, note that,

$$\epsilon(2,\phi) = \frac{1}{3}\frac{1}{4} = \frac{2}{3}\frac{1}{8}, \tag{4.4.20}$$

which immediately implies that $\phi_{\text{opt.}}(n=3)$ must be a distance of $\frac{1}{3}\frac{1}{8}$ away from ϕ, i.e.,

$$\epsilon(3,\phi) = \frac{1}{3}\frac{1}{8} = \frac{1}{2}\epsilon(2,\phi). \tag{4.4.21}$$

By proceeding with this argument iteratively or formally using induction, it follows that,

$$\epsilon(n+1,\phi) = \frac{1}{2}\epsilon(n,\phi), \quad \text{for all } \ \phi \in \Phi', \tag{4.4.22}$$

as desired. The phases Φ' thus solve the problem initially stated in Sec. 4.3, and thus provide a more well-motivated choice for the values of the free variable ϕ in the test implementation of quantum phase estimation in Fig. 4.1 on which the benchmark will be based.

Lastly, the effective qubit number n_{eff} will be computed by iteratively collecting data D and checking a success criterion $S_n(D)$ for larger and larger qubit count n. The test implementation in Fig. 4.1 needs at least one qubit in the upper register, so the smallest possible value is $n_{\text{eff}} = 1$. This value will be our initial belief for the value of the effective qubit number before any data D is collected. To check whether we can increase our belief to $n_{\text{eff}} = 2$ we collect data D on $n = 2$ qubits and check the success criterion $S_2(D)$. This procedure continues iteratively until $S_n(D) = 0$. The important thing to note is that we start at $n = 2$ qubits, so in fact we only need the phase representatives Φ to satisfy Eq. 4.4.12 for $n \geq 2$, i.e.,

$$\epsilon(n+1,\phi) = \frac{1}{2}\epsilon(n,\phi) \neq 0, \quad \text{for } \ n \geq 2. \tag{4.4.23}$$

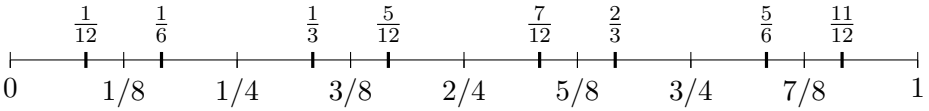
The scaling factor $w(\phi)$ for ϕ satisfying the equation above can be formulated as,

$$w(\phi) = \frac{1}{16\epsilon(2,\phi)} \tag{4.4.24}$$

following the same argument as previously. Following the same argument as in the previous paragraph, it can be shown that ϕ satisfying Eq. 4.4.23 is equivalent to satisfying the following equation,

$$\epsilon(3,\phi) = \frac{1}{2}\epsilon(2,\phi). \tag{4.4.25}$$

Finding $\phi \in [0,1]$ that satisfies this equation is again done by looking at phase approximations in $P_{\text{approx.}}(n=3)$, not in $P_{\text{approx.}}(n=2)$, and choosing ϕ a distance away from these equal to one-third times the distance between adjacent phase approximations, i.e., $\frac{1}{3}\frac{1}{8}$:



The phase representatives we get thus include the previous ones Φ' and four additional ones,

$$\Phi = \{\frac{1}{12}, \frac{1}{6}, \frac{1}{3}, \frac{5}{12}, \frac{7}{12}, \frac{2}{3}, \frac{5}{6}, \frac{11}{12}\}, \tag{4.4.26}$$

This is great; more phases should provide more stable results, but it comes at the cost of higher computational requirements due to the larger number of phase estimates needed. The scaling factors $w(\phi)$ are all equal, since $\epsilon(2,\phi) = \frac{2}{3}\frac{1}{8} = \frac{1}{12}$ for all $\phi \in \Phi$,

$$w(\phi) = \frac{1}{16\epsilon(2,\phi)} = \frac{12}{16} = \frac{3}{4}, \tag{4.4.27}$$

just as before. These phase representatives Φ satisfy Eq. 4.3.10 for $n \geq 2$, and will be the ones we use throughout this thesis.

4.5 Suppressing Statistical Noise

The goal of this section is to figure out how many QPE-calls, n_s, that are needed for the phase estimate,

$$\phi_{est.}(n) = \frac{1}{2^n}\underset{m}{\text{argmax}}\{\hat{p}(m,n_s)\}, \tag{4.5.1}$$

to be equal to $\phi_{\text{opt.}}(n)$ in the absence of noise with at least a probability of $P \in [0,1]$. Following the ideas of the previous section, we will restrict our attention to the phases $\phi \in \Phi$ defined in Eq. 4.4.26. Remember that the empirical probability distribution $\hat{p}(m,n_s)$ is constructed from the classical data obtained from n_s QPE-calls according to Eq. 4.1.4.

In the absence of noise, the empirical probability distribution will approach the theoretical probability distribution $p(m)$ given in Eq. 2.2.13 in the limit of infinitely many QPE-calls, i.e.,

$$p(m) = \lim_{n_s \to \infty} \hat{p}(m,n_s). \tag{4.5.2}$$

In the rest of the section, we will assume that we have perfect hardware, i.e., that we are in the absence of noise. Under this assumption, we are sampling from $p(m)$ when running the quantum phase estimation algorithm, making it a learning problem. In other words, we want to learn properties of $p(m)$ to a given precision given an empirical probability distribution $\hat{p}(m,n_s)$ on n_s samples with a given probability $P \in [0,1]$. Naively, we could try to learn the entire structure of the theoretical probability distribution,

$$p : \Omega = \{0,1,\ldots,2^n-1\} \to [0,1], \tag{4.5.3}$$

to a given precision $\epsilon > 0$ with respect to some metric, say the trace distance $\frac{1}{2}\sum_{m\in\Omega}|p(m)-\hat{p}(m,n_s)|$, with some probability $P \in [0,1]$. To do so, we would as a bare minimum need to resolve where $p(m) \neq 0$,

which would require at least $|\text{supp}(p)|$ samples, where,

$$\text{supp}(p) = \{m \in \Omega \mid p(m) \neq 0\}, \tag{4.5.4}$$

is called the support of p. Unfortunately for us, this support is almost all of Ω of size 2^n, as is easily verified by plotting $p(m)$ for a couple of values of n, so we would need an exponential number of samples in the qubit count n. The reason that this approach did not work is twofold. Firstly, we did not use any of the structure of $p(m)$. Secondly, we are only interested in the mode of $p(m)$,

$$\text{mode}(p) := \underset{m}{\text{argmax}}\{p(m)\} = m_{\text{opt}}. \tag{4.5.5}$$

So there is no need for us to learn the entire structure of $p(m)$. In the following paragraphs, we will discuss how to utilize the structure of $p(m)$ to learn its mode with high probability and constant resources in the qubit count n.

The important structural property of $p(m)$ is that it is sharply peaked around its mode. From Appendix A we know that the value of the peak at the mode $m_{\text{opt.}}$ for arbitrary $\phi \in [0, 1]$ is bounded from below, i.e.,

$$p(m_{\text{opt.}}) \geq \frac{4}{\pi^2} \approx 0.41. \tag{4.5.6}$$

The larger the lower bound, the narrower $p(m)$ will be around its mode. We will only work with $\phi \in \Phi$ and not arbitrary phases, so it is natural to ask whether this provides us with some extra structure? To investigate this, we will plot $p(m)$ for different $\phi \in \Phi$ and different values of n. This is done in Fig. 4.9 for the phases $\phi \in \{\frac{1}{12}, \frac{5}{12}, \frac{5}{6}\} \subset \Phi$ and the qubit counts $n \in \{2, 3, 4\}$.

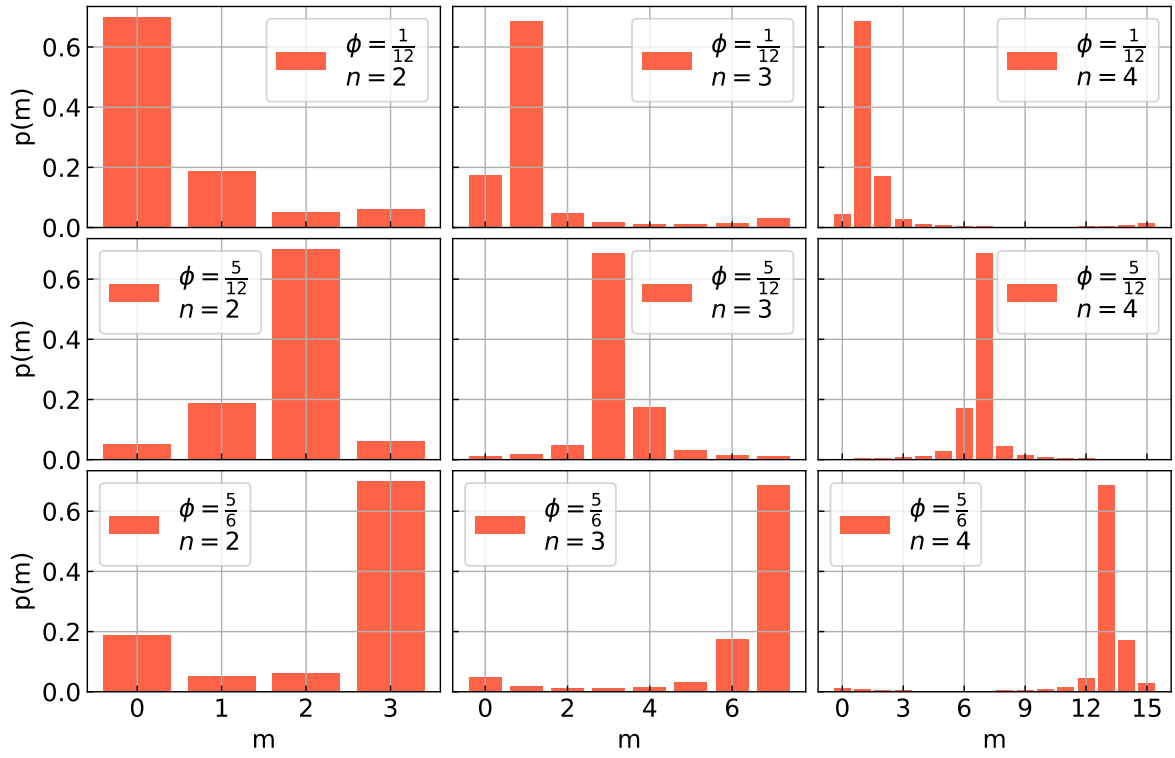


Figure 4.9: The probability distribution $p(m)$ for different phases $\phi \in \{\frac{1}{12}, \frac{5}{12}, \frac{5}{6}\} \subset \Phi$ and different qubit counts $n \in \{2, 3, 4\}$.

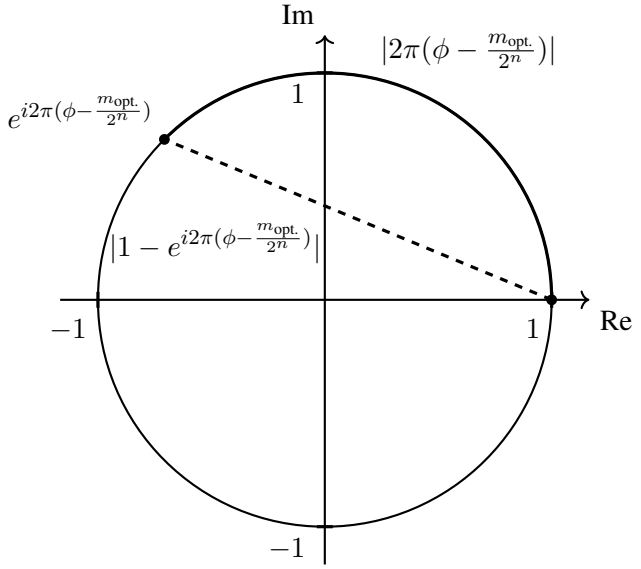
The important observation from this figure is that the value of $p(m)$ at the mode $m_{\text{opt.}}$ seems to have a way higher lower bound of,

$$p(m_{\text{opt.}}) \overset{?}{\geq} 0.65, \quad \text{for } \phi \in \Phi, \tag{4.5.7}$$

than the previously mentioned lower bound of 0.41 for arbitrary $\phi \in [0, 1]$. Guided by this observation, we seek to prove the existence of a higher lower bound of $p(m_{\text{opt.}})$ for $\phi \in \Phi$. To do so, consider the alternative expression of $p(m)$ given in Eq. 2.2.13,

$$p(m) = \frac{1}{2^{2n}} \left| \frac{1 - e^{i2\pi 2^n (\phi - \frac{m}{2^n})}}{1 - e^{i2\pi(\phi - \frac{m}{2^n})}} \right|^2 = \frac{1}{2^{2n}} \frac{|1 - e^{i2\pi 2^n (\phi - \frac{m}{2^n})}|^2}{|1 - e^{i2\pi(\phi - \frac{m}{2^n})}|^2}. \tag{4.5.8}$$

Note that both 1, $e^{i2\pi(\phi - \frac{m}{2^n})}$, and $e^{i2\pi 2^n(\phi - \frac{m}{2^n})}$ lie on the complex unit circle $S^1 \subset \mathbb{C}$, implying that the numerator and denominator in the expression of $p(m)$ above, are distances between points on S^1. Restricting our attention to the mode $m = m_{\text{opt.}}$, we get that $|\phi - \frac{m_{\text{opt.}}}{2^n}| = |\phi - \phi_{\text{opt.}}(n)| \leq \frac{1}{2}\frac{1}{2^n}$, which immediately implies that $e^{i2\pi(\phi - \frac{m_{\text{opt.}}}{2^n})}$ lies on the upper hemisphere of S^1, as is sketched in the figure below.



Since the shortest path between two points in the plane is always a straight line, it follows that,

$$|1 - e^{i2\pi(\phi - \frac{m_{\text{opt.}}}{2^n})}| \leq |2\pi(\phi - \frac{m_{\text{opt.}}}{2^n})|. \tag{4.5.9}$$

One last thing to note, before we can prove our desired inequality, is that if $\phi \in \Phi$ then $\phi = \phi_{\text{opt.}}(n) \pm \frac{1}{3}\frac{1}{2^n}$, as is immediately apparent from the following calculation based on the discussion in Sec. 4.4,

$$\epsilon(n, \phi) = \frac{1}{2}\epsilon(n-1, \phi) = \cdots = \frac{1}{2^{n-2}}\epsilon(2, \phi) = \frac{1}{2^{n-2}}\frac{1}{3}\frac{1}{4} = \frac{1}{3}\frac{1}{2^n}. \tag{4.5.10}$$

We are now in a position to prove the desired inequality for $\phi \in \Phi$:

$$\begin{aligned} p(m_{\text{opt.}}) &= \frac{1}{2^{2n}} \frac{|1 - e^{i2\pi 2^n(\phi - \phi_{\text{opt.}}(n))}|^2}{|1 - e^{i2\pi(\phi - \phi_{\text{opt.}}(n))}|^2} \\ &\geq \frac{1}{2^{2n}} \frac{|1 - e^{i2\pi 2^n(\phi - \phi_{\text{opt.}}(n))}|^2}{4\pi^2|\phi - \phi_{\text{opt.}}(n)|^2} \\ &= \frac{1}{2^{2n}} \frac{|1 - e^{i2\pi 2^n(\phi_{\text{opt.}}(n) \pm \frac{1}{3}\frac{1}{2^n} - \phi_{\text{opt.}}(n))}|^2}{4\pi^2|\phi_{\text{opt.}}(n) \pm \frac{1}{3}\frac{1}{2^n} - \phi_{\text{opt.}}(n)|^2} \\ &= \frac{1}{2^{2n}} \frac{|1 - e^{\pm i\frac{2\pi}{3}}|^2}{4\pi^2|\pm\frac{1}{3}\frac{1}{2^n}|^2} \\ &= \frac{9|1 - e^{\pm i\frac{2\pi}{3}}|^2}{4\pi^2} \\ &= \frac{27}{4\pi^2} \\ &\approx 0.68. \end{aligned} \tag{4.5.11}$$

The last equality follows from the identity $|1 - e^{\pm i\frac{2\pi}{3}}| = \sqrt{3}$. This is in great correspondence with what we observed numerically in Fig. 4.9! This extra structure of $p(m)$ for $\phi \in \Phi$ will be exactly what we need.

The goal is now to learn the mode of $p(m)$ for $\phi \in \Phi$ with probability $P \in [0, 1]$ and constant resources in the qubit count n, using the result in Eq. 4.5.11. We will solve this problem by playing a game, where we sample some value m from $p(m)$ and win if $m = m_{\text{opt.}}$ and lose if $m \neq m_{\text{opt.}}$. Since $p(m_{\text{opt.}}) > 0.5$, we are more likely to win than lose every time we play. If we play n_s times, then the probability of winning n_w times, given a probability to win of $p(m_{\text{opt.}})$, is modeled by a Binomial distribution,

$$f(n_w, n_s, p(m_{\text{opt.}})) = \binom{n_s}{n_w} p(m_{\text{opt.}})^{n_w}(1 - p(m_{\text{opt.}}))^{n_s - n_w}, \tag{4.5.12}$$

as is well-known from elementary statistics. If we win more than half of the n_s times we play, i.e., $n_w > \lfloor \frac{n_s}{2} \rfloor$, then we are guaranteed to recover the mode $m_{\text{opt.}}$ of $p(m)$ from the empirical probability distribution $\hat{p}(m, n_s)$ as follows,

$$m_{opt.}(n) = \underset{m}{\text{argmax}}\{\hat{p}(m, n_s)\}. \tag{4.5.13}$$

We want to choose n_s large enough that this happens with at least a probability of P. To do so, consider the probability that this does not happen, which is given by the cumulative probability distribution,

$$F(n'_w, n_s, p(m_{\text{opt.}})) = \sum_{n_w=0}^{n'_w} f(n_w, n_s, p(m_{\text{opt.}})), \tag{4.5.14}$$

evaluated at $n'_w = \lfloor \frac{n_s}{2} \rfloor$. Since we are more likely to win than lose each time we play, $F(\lfloor \frac{n_s}{2} \rfloor, n_s, p(m_{\text{opt.}}))$ will be a decreasing function in $p(m_{\text{opt.}}) > 0.5$, and it will thus attain its largest value at the lower bound $\frac{27}{4\pi^2} \approx 0.68$ of $p(m_{\text{opt.}})$, i.e.,

$$F(\lfloor \frac{n_s}{2} \rfloor, n_s, p(m_{\text{opt.}})) \leq F(\lfloor \frac{n_s}{2} \rfloor, n_s, \frac{27}{4\pi^2}). \tag{4.5.15}$$

Now let us say that we want to learn the mode of $p(m)$ with at least a probability of $P = 99.99\%$, how many QPE-calls, n_s, would we need? To figure this out, remember that we are guaranteed to obtain the mode $m_{\text{opt.}}$ from $\hat{p}(m, n_s)$ if $n_w > \lfloor \frac{n_s}{2} \rfloor$, which happens with at least a probability of $1 - F(\lfloor \frac{n_s}{2} \rfloor, n_s, \frac{27}{4\pi^2})$, so we need to have that,

$$F(n_s) := F(\lfloor \frac{n_s}{2} \rfloor, n_s, \frac{27}{4\pi^2}) \leq 0.0001. \tag{4.5.16}$$

The shorthand notation $F(n_s)$ illustrates the fact that $F(\lfloor \frac{n_s}{2} \rfloor, n_s, \frac{27}{4\pi^2})$ is only a function of n_s. All that is left to do is to numerically compute $F(n_s)$ for increasing values of n_s until we reach a point where $F(n_s) \leq 0.0001$. The point at which this happens is the smallest number of QPE-calls needed for us to obtain the mode of $p(m)$ with a probability of $P \geq 99.99\%$. This is done and visualized in Fig. 4.10.

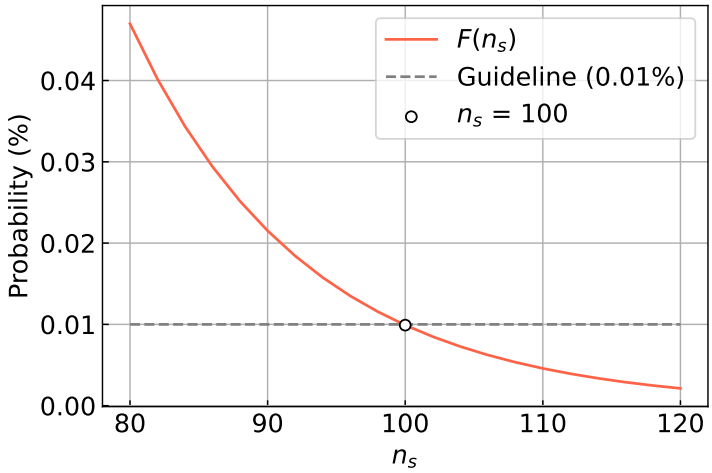


Figure 4.10: The cumulative probability distribution $F(n_s)$ of the binomial distribution given in Eq. 4.5.12 with parameters given in Eq. 4.5.16 versus the number of QPE-calls n_s. It is found that the smallest number of QPE-calls for which $F(n_s) \leq 0.01\%$ is $n_s = 100$.

From the numerical computation in Fig. 4.10, it is observed that the smallest number of QPE-calls for which $F(n_s) \leq 0.01\%$ is $n_s = 100$, which is thus the smallest number of QPE-calls needed for us to obtain the mode of $p(m)$ with a probability of $P \geq 99.99\%$. This entire analysis did only depend on the inequality $p(m_{\text{opt.}}) \geq \frac{27}{4\pi^2} \approx 0.68$, which is independent of both n and $\phi \in \Phi$, so $n_s = 100$ QPE-calls is enough to ensure that

$$m_{opt.}(n) = \underset{m}{\text{argmax}}\{\hat{p}(m, n_s)\} \quad \Longrightarrow \quad \phi_{\text{est.}}(n) = \phi_{\text{opt.}}(n) \tag{4.5.17}$$

with at least a probability of 99.99% for all $\phi \in \Phi$ and for all $n \geq 2$. Remember that all this holds only in the absence of noise; in the presence of noise, the distribution of outcomes for quantum phase estimation $p(m)$ will change in ways that directly depend on the characteristics of the noise model.

4.6 Defining a Success Criterion

This section aims to define a success criterion that properly characterizes when noise "takes over" in the system. Such a success criterion,

$$S_n(D) \in \{0, 1\}, \tag{4.6.1}$$

will be defined for each size n of the test implementation of quantum phase estimation in Fig. 4.1, as a function of D, the collection of phase estimates for $\phi \in \Phi$, each computed according to Eq. 4.1.5 a certain

number of times. We will follow the convention that $S_n(D) = 0$ corresponds to failure, and $S_n(D) = 1$ corresponds to success, respectively. Ideally, $S_n(D) = 1$ for increasing qubit count n as long as noise has not "taken over", and right as it does, it should change to $S_n(D) = 0$. This will naturally lead to the definition of the effective qubit number as the largest qubit count n for which $S_n(D) = 1$, as is done in Sec. 4.7.

The main problem will then be to define mathematically precisely that noise "takes over". Before attempting to solve this problem, a quick recap of the results of Sec. 4.4 and 4.5 will be provided. In Sec. 4.4 representatives $\Phi \subset [0, 1]$ for the free variable ϕ was found,

$$\Phi = \{\frac{1}{12}, \frac{1}{6}, \frac{1}{3}, \frac{5}{12}, \frac{7}{12}, \frac{2}{3}, \frac{5}{6}, \frac{11}{12}\}. \tag{4.6.2}$$

These had two important properties. Firstly, they allowed us to reexpress,

$$\epsilon(n) := \int_0^1 \epsilon(n, \phi) d\phi = \frac{1}{2^{n+2}}, \quad \epsilon(n, \phi) = |\phi - \phi_{\text{opt.}}(n)|, \tag{4.6.3}$$

into a finite weighted sum over Φ,

$$\epsilon(n) = \frac{1}{|\Phi|} \sum_{\phi \in \Phi} w(\phi)\epsilon(n, \phi), \quad w(\phi) = \frac{3}{4} \text{ and } n \geq 2. \tag{4.6.4}$$

This ensures that the weighted average of prediction errors $|\phi - \phi_{\text{est.}}(n)|$ over Φ,

$$\epsilon_{\text{est.}}(n) = \frac{1}{|\Phi|} \sum_{\phi \in \Phi} w(\phi)|\phi - \phi_{\text{est.}}(n)| \tag{4.6.5}$$

is equal to $\epsilon(n)$ in the absence of noise and the limit $n_s \to \infty$, since under these conditions, we have that $\phi_{\text{est.}}(n) = \phi_{\text{opt.}}(n)$. In reality, we can only use a finite $n_s < \infty$ number of QPE-calls, which naturally leads to the second property of the representatives Φ. In Sec. 4.5, it is shown that $n_s = 100$ QPE-calls are enough to ensure that,

$$\phi_{\text{est.}}(n) = \phi_{\text{opt.}}(n), \tag{4.6.6}$$

in the absence of noise, with at least a 99.99% probability. For $\epsilon_{\text{est.}}(n)$ to be equal to $\epsilon(n)$, it is necessary that $\phi_{\text{est.}}(n) = \phi_{\text{opt.}}(n)$ for all eight phases $\phi \in \Phi$. The probability that this happens is at least equal to,

$$(0.9999)^8 \approx 0.9992. \tag{4.6.7}$$

So $\epsilon_{\text{est.}}(n) = \epsilon(n)$ in the absence of noise using $n_s = 100$ QPE-calls with at least a 99.92% probability. It is thus in the absence of noise statistically very unlikely to observe deviations $\epsilon_{\text{est.}}(n) \neq \epsilon(n)$, while they are expected to occur with larger probabilities in the presence of noise. This allows us to use $\epsilon(n)$ as a reference for detecting the presence of noise on hardware.

To test all this work, we gather phase estimates $\phi_{\text{est.}}(n)$ according to Eq. 4.1.5 from our custom noise model for each $\phi \in \Phi$ and $n \in \{2, 3, \ldots, 8\}$, from which $\epsilon_{\text{est.}}(n)$ are computed according to Eq. 4.6.5. The results are visualized in Fig. 4.11. For a small number of qubits, we only need a small number of gates, which results in a low amount of noise, for which we expect $\epsilon_{\text{est.}}(n) = \epsilon(n)$. For a larger number of qubits, we expect deviations $\epsilon_{\text{est.}}(n) \neq \epsilon(n)$ due to the higher amount of noise, as is observed in Fig. 4.11.

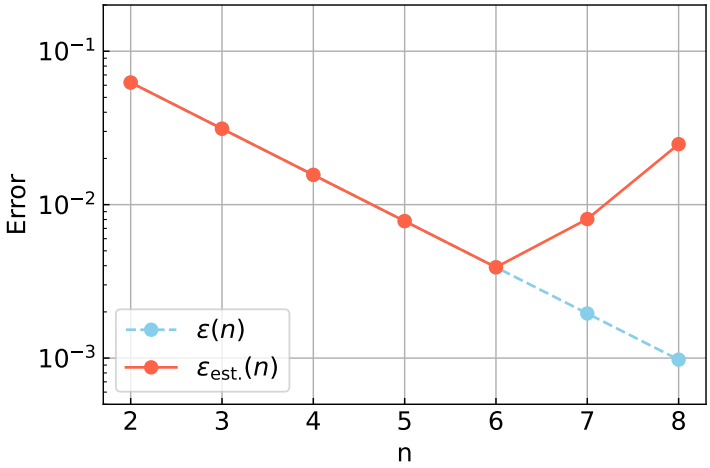


Figure 4.11: Average prediction error $\epsilon_{\text{est.}}(n)$ as a function of qubit count n, where $\epsilon_{\text{est.}}(n)$ is computed according to Eq. 4.6.5, from the phase estimates $\phi_{\text{est.}}(n)$ sampled from simulations on our custom noise model according to Eq. 4.1.5.

The basic idea will be to define that noise has "taken over" as soon as the deviation of $\epsilon_{\text{est.}}(n)$ from $\epsilon(n)$ is above some threshold. Unfortunately $\epsilon_{\text{est.}}(n)$ suffers from the same problem as $\epsilon^{\text{Ebbe}}_{\text{est.}}(n)$, i.e., it varies slightly from experiment to experiment, making such a definition less well-defined. This is illustrated in Fig. 4.12 for six independent simulations on our custom noise model.

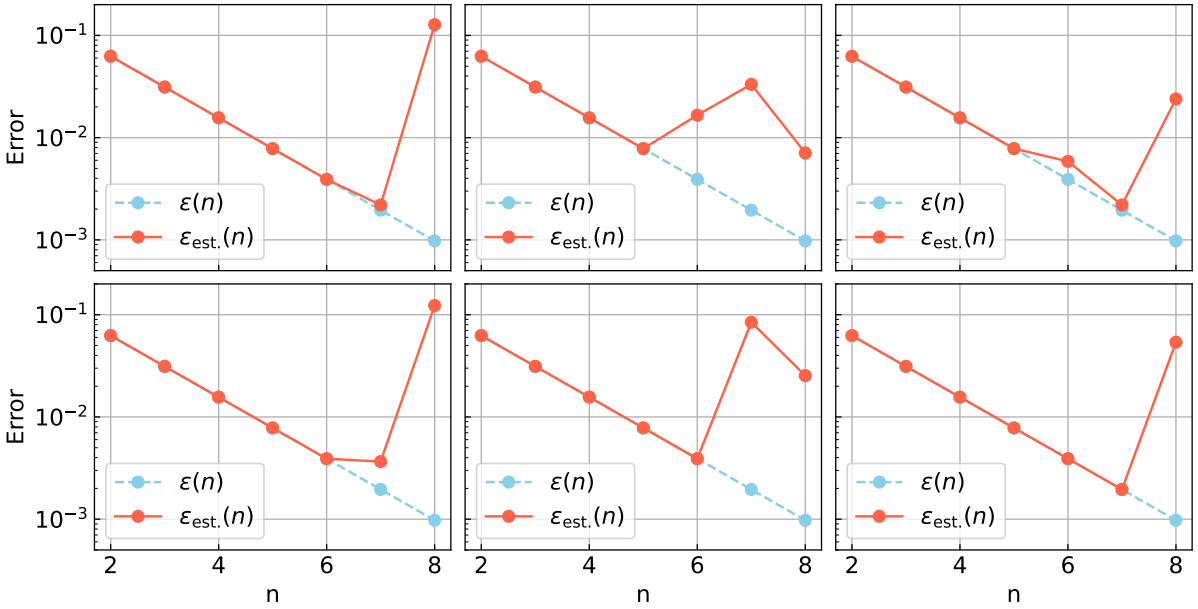


Figure 4.12: Average prediction error $\epsilon_{\text{est.}}(n)$ versus qubit count n for six independent simulations on our custom noise model, each done exactly like in Fig. 4.11.

This problem will be approached by sampling $\epsilon_{\text{est.}}(n)$ several times, say $n_\epsilon \in \mathbb{N}$, resulting in a collection of data points $\epsilon^1_{\text{est.}}(n)$, $\epsilon^2_{\text{est.}}(n)$, ..., $\epsilon^{n_\epsilon}_{\text{est.}}(n)$, for which the mean,

$$\mu_\epsilon(n) := \frac{1}{n_\epsilon} \sum_{i=1}^{n_\epsilon} \epsilon^i_{\text{est.}}(n), \tag{4.6.8}$$

can be used to characterize when noise has "taken over". Each data point $\epsilon^i_{\text{est.}}(n)$ is computed as,

$$\epsilon^i_{\text{est.}}(n) = \frac{1}{|\Phi|}\sum_{\phi\in\Phi} w(\phi)|\phi - \phi^i_{\text{est.}}(n)| = \frac{3}{4|\Phi|}\sum_{\phi\in\Phi} w(\phi)|\phi - \phi^i_{\text{est.}}(n)|, \tag{4.6.9}$$

using $|\Phi|$ independent phase estimates $\phi^i_{\text{est.}}(n)$, each computed according to Eq. 4.1.5. The success criterion will be a function of the collection of phase estimates,

$$D(n) := \{\phi^i_{\text{est.}}(n) \;\mid\; \phi \in \Phi \text{ and } i \in \{1, 2, \ldots, n_\epsilon\}\}, \tag{4.6.10}$$

from which the mean $\mu_\epsilon(n)$ can be computed. Before proceeding to define the success criterion, we make an important remark. Unlike the naive extension of Ebbe's idea in Eq. 4.2.1, the mean value $\mu_\epsilon(n)$ is invariant under swaps of the phase estimates,

$$\phi^i_{\text{est.}}(n) \;\longleftrightarrow\; \phi^j_{\text{est.}}(n), \tag{4.6.11}$$

in different weighted averages $\epsilon^i_{\text{est.}}(n)$ and $\epsilon^j_{\text{est.}}(n)$, as a direct consequence of the commutativity of addition. This property ensures that the effective qubit number, as it will be defined in Sec. 4.7, will be more implementation-robust than the naive extension of Ebbe's ideas presented in Sec. 4.2. There is one caveat to this: Assume a user has sampled the data $D(n)$, consisting of phase estimates $\phi^i_{\text{est.}}(n)$, from which they can compute $\mu_\epsilon(n)$. They could then cheat by swapping counts between empirical probability distributions, $\hat{p}^i(n, n_s = 100)$ and $\hat{p}^j(n, n_s = 100)$, corresponding to different phase estimates, $\phi^i_{\text{est.}}(n)$ and $\phi^j_{\text{est.}}(n)$, thus changing $\mu_\epsilon(n)$. This involves data manipulation, which is not permitted. This, alongside other rules, will be written down at the end of Sec. 4.7, to ensure that the effective qubit number is measured on different devices under the same premises, making comparisons possible.

Let us discuss a possible definition for when noise "takes over", followed by a definition of a success criterion $S_n(D(n))$ on which the effective qubit number will be based. In the absence of noise, $\epsilon_{\text{est.}}(n) = \epsilon(n)$ with at least a 99.92% probability. So as we increase the qubit count $n-1 \to n$, we expect an average gain in numerical accuracy of,

$$\delta_{\text{gain}}(n) = \epsilon(n-1) - \epsilon(n) = \epsilon(n-1) - \frac{1}{2}\epsilon(n-1) = \frac{1}{2}\epsilon(n-1) = \epsilon(n). \tag{4.6.12}$$

In the presence of noise, deviations $\epsilon_{\text{est.}}(n) \neq \epsilon(n)$ are expected, leading to a loss in numerical accuracy. To make this more precise, note that,

$$|\phi - \phi_{\text{est.}}(n)| \geq |\phi - \phi_{\text{opt.}}(n)|, \tag{4.6.13}$$

since $\phi_{\text{opt.}}(n)$ minimizes the distance to ϕ. This immediately implies that,

$$\epsilon_{\text{est.}}(n) = \frac{1}{|\Phi|}\sum_{\phi\in\Phi} w(\phi)|\phi - \phi_{\text{est.}}(n)| = (\frac{1}{|\Phi|}\sum_{\phi\in\Phi} w(\phi)|\phi - \phi_{\text{opt.}}(n)|) + \delta_{\text{loss}}(n) = \epsilon(n) + \delta_{\text{loss}}(n), \tag{4.6.14}$$

where $\delta_{\text{loss}}(n) \geq 0$ is the difference between $\epsilon_{\text{est.}}(n)$ and $\epsilon(n)$, i.e., the loss in numerical accuracy due to noise. The mean $\mu_\epsilon(n)$ can be rewritten in terms of this definition as,

$$\mu_\epsilon(n) := \frac{1}{n_\epsilon}\sum_{i=1}^{n_\epsilon} \epsilon^i_{\text{est.}}(n) = \epsilon(n) + \Delta_{\text{loss}}(n), \quad \Delta_{\text{loss}}(n) = \frac{1}{n_\epsilon}\sum_{i=1}^{n_\epsilon} \delta^i_{\text{loss}}(n), \tag{4.6.15}$$

where $\Delta_{\text{loss}}(n) \geq 0$ is the deviation of the mean $\mu_\epsilon(n)$ from $\epsilon(n)$, and $\delta^i_{\text{loss}}(n) \geq 0$ is the deviation of the i'th sample $\epsilon^i_{\text{est.}}(n)$ from $\epsilon(n)$. Motivated by the central limit theorem [33], we use the standard deviation of the mean, also called the standard error,

$$\alpha_\epsilon(n) = \frac{\sigma_\epsilon(n)}{\sqrt{n_\epsilon}}, \quad \sigma_\epsilon(n) := \sqrt{\frac{1}{n_\epsilon - 1}\sum_{i=1}^{n_\epsilon}(\epsilon^i_{\text{est.}}(n) - \mu_\epsilon(n))^2}. \tag{4.6.16}$$

as an error estimate for the mean $\mu_\epsilon(n)$. Note that the factor $\frac{1}{n_\epsilon - 1}$ is used instead of $\frac{1}{n_\epsilon}$ since one degree of freedom has already been used to compute the mean $\mu_\epsilon(n)$. Since $\mu_\epsilon(n) = \epsilon(n) + \Delta_{\text{loss}}(n)$, the standard error $\alpha_\delta(n)$ of $\Delta_{\text{loss}}(n)$ is equal to that of $\mu_\epsilon(n)$, i.e.,

$$\alpha_\delta(n) = \alpha_\epsilon(n). \tag{4.6.17}$$

A possible precise definition of when noise has "taken over", goes as follows: If the average loss of numerical accuracy $\Delta_{\text{loss}}(n)$ due to noise, as we increase the qubit count $n - 1 \to n$, is smaller than the theoretical gain of numerical accuracy $\delta_{\text{gain}}(n)$, within its uncertainties,

$$\Delta_{\text{loss}}(n) + \alpha_\delta(n) < \delta_{\text{gain}}(n), \tag{4.6.18}$$

then we say that noise has not yet "taken over". If this is not the case, then we will, on average, have lost more than we have gained, and it would thus be redundant to increase the qubit count, leading to the conclusion that noise has "taken over". This directly leads to the definition of the success criterion $S_n(D(n))$, as a function of the data $D(n)$ defined in Eq. 4.6.10 as,

$$S_n(D(n)) := \begin{cases} 1, & \text{if } \Delta_{\text{loss}}(n) + \alpha_\delta(n) < \delta_{\text{gain}}(n) \\ 0, & \text{otherwise} \end{cases}, \quad n \geq 2. \tag{4.6.19}$$

In the following section, this success criterion will be used to define the effective qubit number n_{eff}.

4.7 Definition of the Effective Qubit Number

This section aims to define the effective qubit number n_{eff} as the largest number of qubits, on which a given device can run the test implementation of quantum phase estimation in Fig. 4.1 before noise "takes over". This will be done using the success criterion $S_n(D(n))$ defined in Eq. 4.6.19, which captures when noise has "taken over", as defined in the paragraphs surrounding Eq. 4.6.18.

The test implementation of quantum phase estimation in Fig. 4.1 requires at least one qubit in the upper register, so the smallest possible value of the effective qubit number is $n_{\text{eff}} = 1$. This will be our initial belief for the value of n_{eff} before any data has been collected. To check whether we can increase our belief to $n_{\text{eff}} = 2$, we collect data $D(n)$ as defined in Eq. 4.6.10 on $n = 2$ qubits, and check whether $S_2(D(2)) = 1$. This procedure continues iteratively until $S_{n'}(D(n')) = 0$, resulting in $n_{\text{eff}} = n' - 1$, or until we reach the maximum number of qubits on the device, in which case $n_{\text{eff}} = n'$. An alternative, but equivalent, formulation is given below,

$$n_{\text{eff}} = 1 + \sum_{n=2}^{n'} S_n(D(n)). \tag{4.7.1}$$

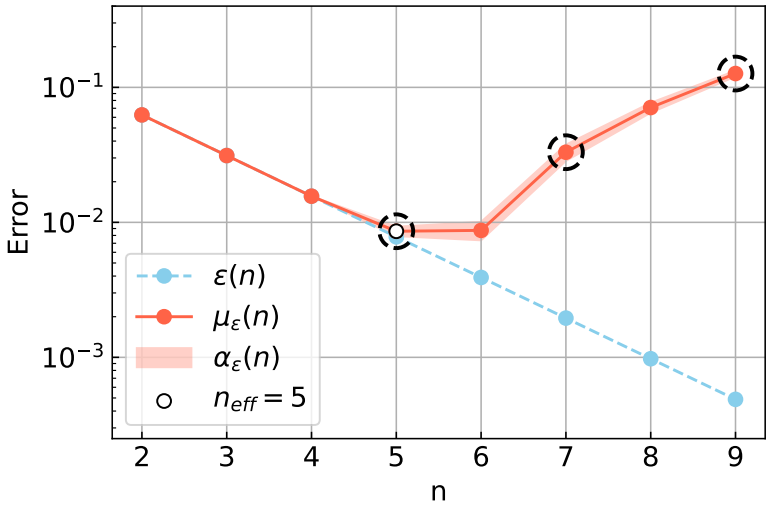


Figure 4.13: Example of computation of n_{eff} according to Eq. 4.7.1 using data obtained from our custom noise model with $n_\epsilon = 100$. The circled data points are used later to discuss the chosen value of n_ϵ

In Fig. 4.13, the results of running this procedure with $n_\epsilon = 100$ on our custom noise model are visualized. The resulting effective qubit number is $n_{\text{eff}} = 5$. To see this, note that the condition in Eq. 4.6.18 is equivalent to,

$$\Delta_{\text{loss}}(n) + \alpha_\delta(n) + \epsilon(n) < \epsilon(n) + \epsilon(n) \quad \Longleftrightarrow \quad \mu_\epsilon(n) + \alpha_\epsilon(n) < \epsilon(n-1). \tag{4.7.2}$$

Using this, it is simple to verify by visual inspection that $n_{\text{eff}} = 5$ for the data given in Fig. 4.13.

All the steps above are summarized in a concise algorithmic format below, as to ensure the ease of measuring the effective qubit number n_{eff}.

Algorithm 1 The effective qubit number (n_{eff})

1: Let $n_{\text{max}} + 1$ be the maximum number of qubits available
2: Fix $n_\epsilon \in \mathbb{N}$
3: Start with $n = 2$ and $n_{\text{eff}} = 1$
4: For each $\phi \in \Phi$ sample $\phi^i_{\text{est.}}(n)$ for $i = 1, 2, \ldots, n_\epsilon$ ▷ Eq. 4.1.5
5: Compute $\epsilon^i_{\text{est.}}(n)$for $i = 1, 2, \ldots, n_\epsilon$ ▷ Eq. 4.6.5
6: Compute $\Delta_{\text{loss}}(n)$ and $\alpha_\delta(n)$ ▷ Eq. 4.6.15, 4.6.16, and 4.6.17
7: Compute $S_n(D(n))$ ▷ Eq. 4.6.19
8: **while** $S_n(D(n)) = 1$ **do**
9: **if** $n = n_{\text{max}}$ **then**
10: $n_{\text{eff}} = n_{\text{max}}$
11: **break**
12: **else**
13: $n_{\text{eff}} = n$
14: $n \to n + 1$
15: Repeat steps $4 - 7$
16: **end if**
17: **end while**
18: **return** n_{eff}

Classical processing tools, such as transpilers and error mitigation, are not mentioned in the algorithm.

The reason is that we take a holistic approach to quantum computers, considering them part of the entire quantum computing stack. All kinds of classical processing tools are therefore allowed as long as the following list of strict rules is upheld.

Transparency All used classical processing tools, no matter the nature of them, should be reported alongside with n_{eff}. This will ensure transparency between the providers of quantum devices, who report values of different benchmarks, and the users of these devices.

Cheating No knowledge of the theoretically expected outcomes $p(m)$ of the test implementation, on which n_{eff} is based, is allowed in the use of classical processing tools. Otherwise, we could always adjust the experimental results to bring them closer to their expected values, resulting in better data.

Data manipulation Manipulating the empirical probability distributions, $\hat{p}^i(n, n_s)$ and $\hat{p}^j(n, n_s)$, underlying different phase estimates, $\phi^i_{\text{est.}}(n)$ and $\phi^j_{\text{est.}}(n)$, by swapping counts, can lead to changes in $\mu_\epsilon(n)$ and thus to changes in n_{eff}. This is data manipulation and is not permitted under any circumstances.

This concludes the construction of the effective qubit number n_{eff}, a holistic and application oriented benchmark for universal gate-based quantum computers. Its construction drew inspiration from $n^{\text{Ebbe}}_{\text{eff}}$ and the definition of an ideal benchmark. It has been improved on several criteria of an ideal benchmark compared to $n^{\text{Ebbe}}_{\text{eff}}$. Firstly, it is more well-motivated due to the choice of phase representatives Φ chosen in Sec. 4.4. Secondly, the definition in terms of a minimum, whose existence is unknown, has been replaced with an iterative definition in terms of a success criterion, making it more well-defined. Thirdly, it seems to be very system-robust, as is shown in Chapter 5. Besides all this, a fixed number of QPE-calls needed to get reasonable phase estimates in the absence of noise has been found in Sec. 4.5.

Before we finish this chapter, let us quickly discuss the resources used to measure n_{eff}. An important low-level resource estimate is the number of one- and two-qubit gates used in each test-implementation of quantum phase estimation in Fig. 4.1. Since this depends on the types of basis-gates available, we will make a generic choice, $\mathcal{B} = \{R_x, R_y, R_z, \text{CNOT}\}$. Now for each $n \in \{2, 3, \ldots, 15\}$, the circuit in Fig. 4.1 with $\phi = \frac{1}{12}$ is decomposed into the basis-gates of $\mathcal{B}$, using Qiskit's transpiler with optimization level set to 3. The result is visualized in Fig. 4.14.

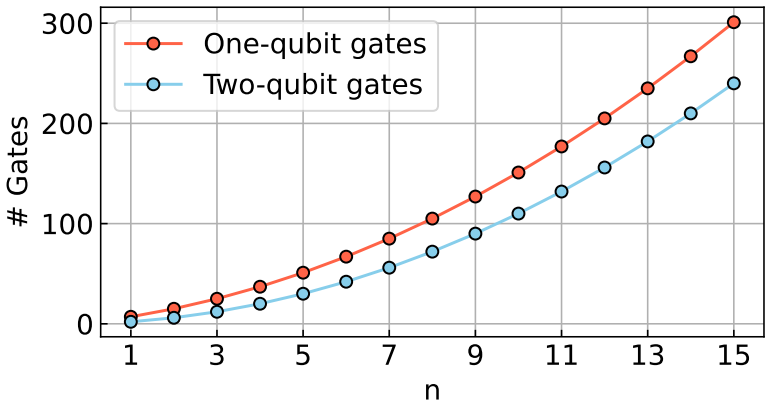


Figure 4.14: Number of one- and two-qubit gates in the decomposition of the circuit in Fig. 4.1 into the basis gates in $\mathcal{B}$ using Qiskit's transpiler with optimization level set to 3.

A natural high-level resource estimate is the total number of QPE-calls. To find this, let n' be the first qubit count for which $S'_n(D(n')) = 0$. Then for each qubit count between 2 and n', there will be sampled n_ϵ phase estimates for each $\phi \in \Phi$, each based on $n_s = 100$ QPE-calls. This results in the following total number of QPE-calls,

$$n_{\text{QPE}} = (\#\, n\text{'s}) \cdot n_\epsilon \cdot |\Phi| \cdot n_s = 800 \cdot (n' - 1) \cdot n_\epsilon. \tag{4.7.3}$$

For the simulation in Fig. 4.13 the total number of QPE-calls equals $n_{\text{QPE}} = 800 \cdot 5 \cdot 100 = 400\,000$. It is clear that n_ϵ controls the number of QPE-calls needed. The size of n_ϵ determines the accuracy to which we can learn the mean,

$$\mu_\epsilon(n) \pm \alpha_\epsilon(n). \tag{4.7.4}$$

To do proper statistics, we should have that $n_\epsilon \geq 30$, but the actual value depends on the distribution that we want to learn the mean of. The distribution underlying $\mu_\epsilon(n)$ at the circled data points in Fig. 4.13 is plotted in Fig. 4.15. To indicate that noise has not yet "taken over", i.e.,

$$\Delta_{\text{loss}}(n) + \alpha_\delta(n) < \delta_{\text{gain}}(n), \tag{4.7.5}$$

the line $\Delta_{\text{loss}}(n)$ with shaded background $\pm\alpha_\delta(n)$ is colored green, while it is colored red if noise has "taken over". Another aspect affecting the choice of n_ϵ is discussed in Sec. 5.2. The distributions in Fig.

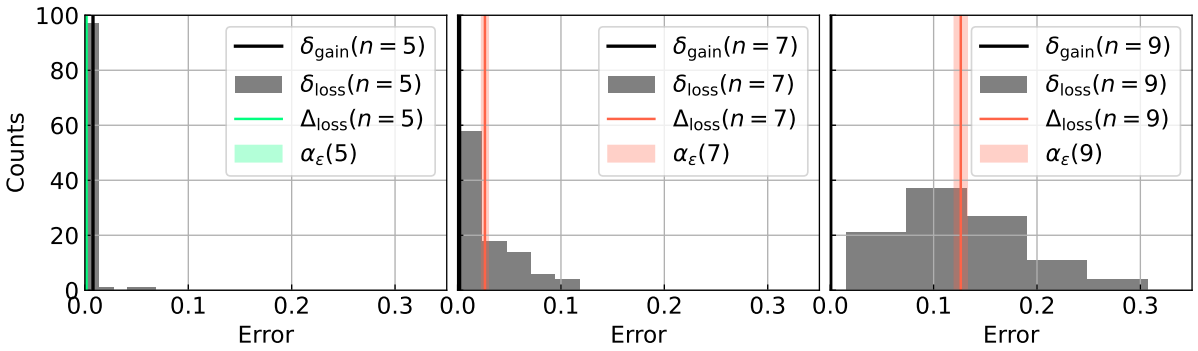


Figure 4.15: Distributions underlying marked data points in Fig. 4.13. The line $\Delta_{\text{loss}}(n)$ with shaded background $\pm\alpha_\delta(n)$ is colored green if noise has not yet "taken over", and red otherwise.

4.15 are for a specific choice of noise model, and might vary for other devices. Since we will not make any assumptions regarding the noisy character of quantum devices, to remain technology-independent, the choice of n_ϵ may vary from device to device. The choice $n_\epsilon = 100$ is used in all simulations throughout this thesis.

CHAPTER 5

Simulations of the Effective Qubit Number

This chapter aims at testing the system-robustness of the effective qubit number as defined in Sec. 4.7 using the custom noise model described in Appendix B, as well as testing its sensitivity towards changes in the underlying noise model. These changes will be modeled by parameterizing the custom noise model with a parameter $g \in \mathbb{R}$, such that the gate errors are directly proportional to g and the qubit thermal relaxation times are inversely proportional to g. The custom noise model used so far corresponds to $g = 1$.

5.1 System-Robustness

In Sec. 4.7, the hope was that by defining n_{eff} using the success criterion $S_n(D(n))$ in Eq. 4.6.19 it would become more system-robust than the definition made by Ebbe. To test this, n_{eff} will be computed with $n_\epsilon = 100$ according to Eq. 4.7.1 six independent times. The results are visualized in Fig. 5.1.

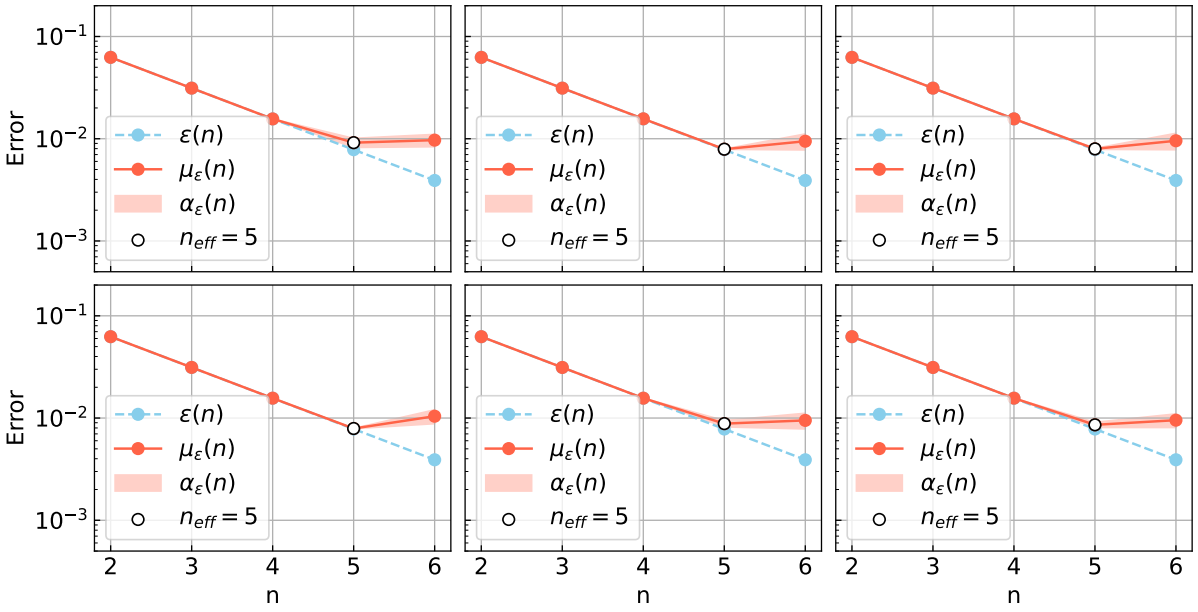


Figure 5.1: The effective qubit number is computed according to Eq. 1 with $n_\epsilon = 100$ six independent times using data collected from our custom noise model.

This resulted in the same value of the effective qubit number for all six independent simulations, i.e., $n_{\text{eff}} = 5$. Doing this four more times, resulting in a total of ten independent simulations, the result remains the same, all effective qubit numbers are $n_{\text{eff}} = 5$. This is a strong indication of an increase in system-robustness, compared to that of $n_{\text{eff}}^{\text{Ebbe}}$, where effective qubit numbers in the range $5 - 8$ from six independent simulations was observed in Fig. 4.4.

5.2 Noise Sensitivity

By varying the continuous parameter g characterizing our custom noise model, and computing n_{eff} each time, we can get an intuition for the sensitivity of n_{eff} to small changes in the underlying noise model. This is done in Fig. 5.2 for $g \in \{0.8, 0.85, 0.9, \ldots, 1.4\}$ and $n_\epsilon = 100$.

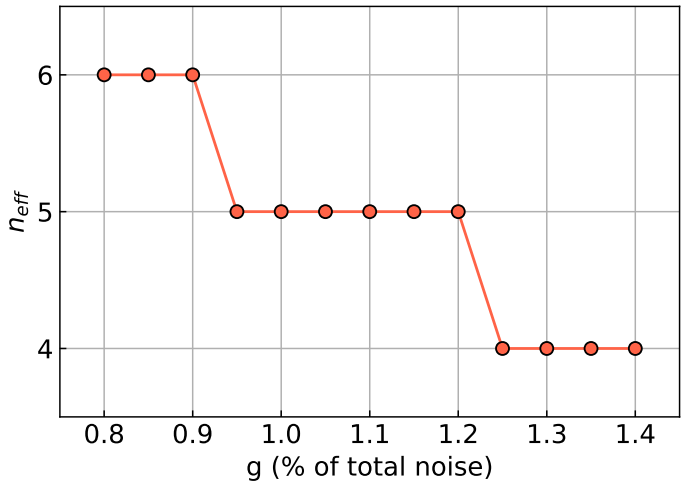


Figure 5.2: The effective qubit number is computed for $g \in \{0.8, 0.85, 0.9, \ldots, 1.4\}$ and $n_\epsilon = 100$ using our custom noise model.

Each of the plateaus in Fig. 5.2 corresponds to a class of simulators with different noise parameters g, each having the same effective qubit number. Within each plateau n_{eff} is stable, but as we approach the edge of one, it abruptly changes to $n_{\text{eff}} \pm 1$ during a small change in g. To see why, consider Fig. 4.13, which corresponds to the data point $g = 1$ in Fig. 5.2. In Fig. 4.13, $S_n(D(n = 6)) = 0$, but a small decrease in noise g, would lead to a small decrease in $\Delta_{\text{loss}}(n = 6)$, which would lead to a success $S_n(D(n = 6)) = 1$, increasing the effective qubit number to $n_{\text{eff}} = 6$. This is precisely the behavior observed in Fig. 5.2 at $g = 1$, where a small decrease in g, leads to an increase in n_{eff} by one. It is worth noting that close to the edges of a plateau, it will take a lot of data points n_ϵ to resolve whether $\Delta_{\text{loss}}(n)$ is above or below $\delta_{\text{gain}}(n)$, within its uncertainties. Nonetheless, such resolution should always be possible by increasing n_ϵ to arbitrarily large numbers.

All this motivates the search for a continuous extension $n_{\text{eff}} \in \mathbb{R}$, where slight variations in the underlying noise model correspond to small changes in n_{eff}, thus avoiding the abrupt changes at the edges of the plateaus in Fig. 5.2. Defining such a continuous extension is attempted in Chapter 6. It is found that the gain of continuity comes at the cost of system-robustness, ease of comparison between devices, and the simplistic interpretation as the largest number of qubits on which the test implementation in Fig. 4.1 can be run before noise takes over.

CHAPTER 6

The Effective Qubit Number Continued

As discussed at the end of Chapter 5, this section aims a extending n_{eff} continuously to a real number $\mathbb{R}$. To avoid confusion, the continuous extension will be denoted by n_{eff}^c, to distinguish it from the original definition n_{eff} in Sec. 4.7. The hope is that a continuous extension would enforce small changes in the underlying noise model of a quantum device, to result in small changes in n_{eff}^c. This will be tested alongside system-robustness, and compared to n_{eff} in the end.

6.1 Extending the Effective Qubit Number Continuously

A continuous extension of n_{eff} is only a small step and a simple idea away from the definition provided in Sec. 4.7. Here, the effective qubit number is defined as,

$$n_{\text{eff}} = 1 + \sum_{n=2}^{n'} S_n(D(n)), \tag{6.1.1}$$

where n' is either the first qubit count for which $S_{n'}(D(n')) = 0$, or one less than the maximum number of qubits on the device, if the success criterion $S_n(D(n)) = 1$ for all qubit counts n. The basic idea is to promote the binary success criterion $S_n(D(n))$ to a continuous success score $S_n^c(D(n)) \in [0, 1]$. It will similarly to $S_n(D(n))$ be defined for each qubit count n as a function of the classical data $D(n)$,

$$S_n^c(D(n)) := \begin{cases} \frac{\delta_{\text{gain}}(n) - \Delta_{\text{loss}}(n)}{\delta_{\text{gain}}(n)}, & \text{if } \Delta_{\text{loss}}(n) + \alpha_\delta(n) < \delta_{\text{gain}}(n) \\ 0, & \text{otherwise} \end{cases}. \tag{6.1.2}$$

This can be interpreted in the following way. If we are confident that noise has not yet "taken over", i.e., $\Delta_{\text{loss}}(n) + \alpha_\delta(n) < \delta_{\text{gain}}(n)$, then we assign the device a score between 0 and 1, that describes linearly how far we are from the ideal case $\Delta_{\text{loss}}(n) = 0$. This leads naturally to a continuous extension of n_{eff} by replacing the binary success criterion $S_n(D(n))$ with the continuous success score $S_n^c(D(n))$,

$$n_{\text{eff}}^c = 1 + \sum_{n=2}^{n'} S_n^c(D(n)), \tag{6.1.3}$$

where n' is defined in the same way. Since $n_{\text{eff}}^c \in \mathbb{R}$ small variations will most likely occur between different measurements of it. So, to compare measurements of it for different devices, an uncertainty is necessary. To compute one, note that each $S_n^c(D(n))$ is computed from $\Delta_{\text{loss}}(n)$ for which we have an

uncertainty $\alpha_\delta(n)$. Using standard error propagation calculus [33] this results in an uncertainty $\alpha_s(n)$ for $S_n^c(D(n))$ of the form,

$$\alpha_s(n) = \begin{cases} \frac{\alpha_\delta(n)}{\delta_{\text{gain}}(n)}, & \text{if } \Delta_{\text{loss}}(n) + \alpha_\delta(n) < \delta_{\text{gain}}(n) \\ 0, & \text{otherwise} \end{cases}. \tag{6.1.4}$$

By a second application of standard error propagation calculus, this results in an uncertainty α for n_{eff}^c of the form,

$$\alpha = \sum_{n=2}^{n'} \alpha_s(n), \tag{6.1.5}$$

since success scores $S_n^c(D(n))$ for different n are independent variables. This continuous extension $n_{\text{eff}}^c \pm \alpha$ is computed for the data in Fig. 4.13, for which $n_{\text{eff}} = 5$. The result is $n_{\text{eff}}^c = 4.9 \pm 0.1$ and is plotted in Fig. 6.1.

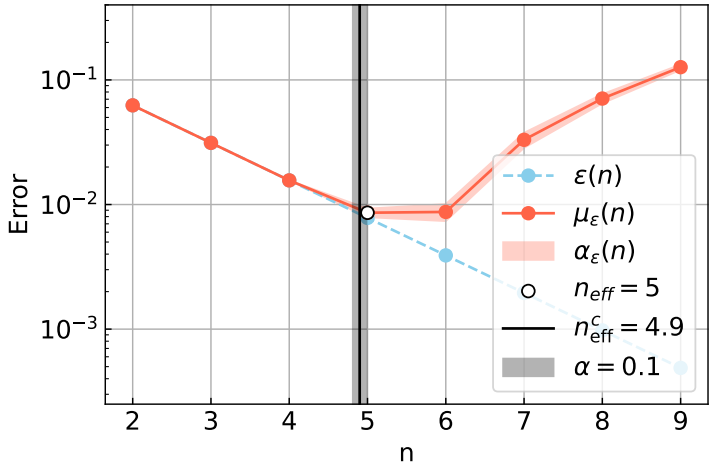


Figure 6.1: Computation of n_{eff}^c according to Eq. 6.1.3 using data obtained from our custom noise model with $n_\epsilon = 100$. The data is the same as in Fig. 4.13.

All the steps above are summarized in a concise algorithmic format in Alg. 2, as to ensure the ease of measuring the continuous extension n_{eff}^c of the effective qubit number.

Algorithm 2 The continuous effective qubit number (n_{eff}^c)

1: Let $n_{\text{max}} + 1$ be the maximum number of qubits available
2: Fix $n_\epsilon \in \mathbb{N}$
3: Start with $n = 2$, $n_{\text{eff}}^c = 1$, and $\alpha = 0$
4: For each $\phi \in \Phi$ sample $\phi_{\text{est.}}^i(n)$ for $i = 1, 2, \ldots, n_\epsilon$ ▷ Eq. 4.1.5
5: Compute $\epsilon_{\text{est.}}^i(n)$for $i = 1, 2, \ldots, n_\epsilon$ ▷ Eq. 4.6.5
6: Compute $\Delta_{\text{loss}}(n)$ and $\alpha_\delta(n)$ ▷ Eq. 4.6.15, 4.6.16, and 4.6.17
7: Compute $S_n^c(D(n))$ ▷ Eq. 6.1.2
8: **while** $\Delta_{\text{loss}}(n) + \alpha_\delta(n) < \delta_{\text{gain}}(n)$ **do**
9: $\quad n_{\text{eff}}^c \to n_{\text{eff}}^c + S_n^c(D(n))$ and $\alpha \to \alpha + \frac{\alpha_\delta(n)}{\delta_{\text{gain}}(n)}$
10: $\quad n \to n + 1$
11: $\quad$ Repeat steps $4 - 7$
12: **end while**
13: **return** $n_{\text{eff}}^c \pm \alpha$

Measuring this extension according to this algorithm should be done while following the rules written down in Sec. 4.7, regarding transparency, data manipulation, and cheating.

6.2 System-Robustness of Continuous Effective Qubit Number

In Sec. 5.1 ten independent simulations of n_{eff} were run, all resulting in the same outcome, $n_{\text{eff}} = 5$. The effective qubit number n_{eff} as defined in Sec. 4.7 thus seems to be very system robust. As argued previously, we might expect the continuous extension n_{eff}^c to be less system robust, since it is a continuous number. This is indeed the case as can be seen in Fig. 6.2, where n_{eff}^c is computed from the same data used in Sec. 5.1.

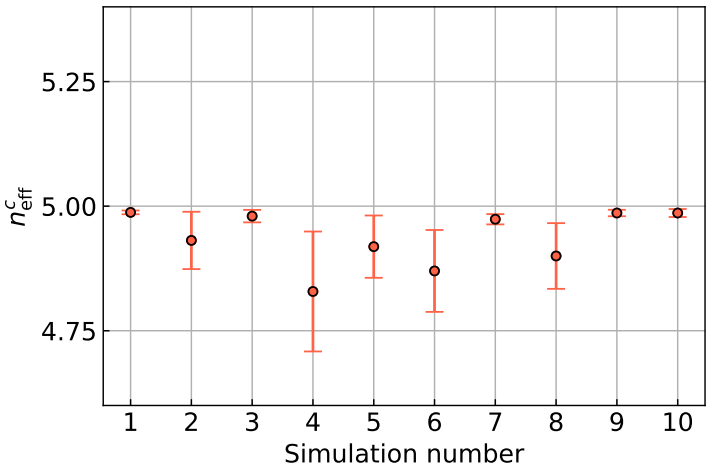


Figure 6.2: Computation of $n_{\text{eff}}^c \pm \alpha$ as defined in Alg. 2 using data from ten independent simulations using our custom noise model. It is the same data as used in Sec. 5.1.

It seems that both n_{eff}^c and its uncertainty α varies from one simulation to another as expected. This implies that $n_{\text{eff}}^c \pm \alpha$ is less system-robust than n_{eff}, which is a clear downside. Furthermore, all data points lie below $n_{\text{eff}}^c = 5$, indicating that n_{eff}^c is more conservative than n_{eff}, in the sense that $n_{\text{eff}}^c \leq n_{\text{eff}}$. This is clearly true in general since,

$$S_n^c(D(n)) \leq S_n(D(n)) \quad \Longrightarrow \quad n_{\text{eff}}^c = 1 + \sum_{n=2}^{n'} S_n^c(D(n)) \leq 1 + \sum_{n=2}^{n'} S_n(D(n)) = n_{\text{eff}}. \tag{6.2.1}$$

The variations in n_{eff}^c seems to be ≤ 0.2. The size of these variations determine the resolution of n_{eff}^c that we can hope to obtain. If this resolution is too rough, then using the system-robust discrete version n_{eff} might be a better option. This will be discussed further in Sec. 6.4.

Before concluding this section, let us discuss why α is smallest when $n_{\text{eff}} \approx 5$. To see why, note that our custom noise model performs very well on $n \in \{2, 3, 4\}$ qubits, slightly worse on $n = 5$ qubits and a lot worse on $n \geq 6$ qubits, as is seen in Fig. 6.1. This results in $S_n^c(D(n)) \approx 1$ for $n \in \{2, 3, 4\}$, $S_n^c(D(n)) < 1$ for $n = 5$ and $S_n^c(D(n)) = 0$ for $n \geq 6$. In Fig. 4.15 it is seen that $\alpha_\epsilon(n)$ decreases as $\Delta_{\text{loss}}(n)$ decreases. So since $\alpha_s(n)$ is proportional to $\alpha_\delta(n) = \alpha_\epsilon(n)$ by definition, it follows that the main contribution to α comes from $\alpha_s(n = 5)$. In conclusion, small values of $\Delta_{\text{loss}}(n)$ correspond to large values of $S_n^c(D(n))$ with small uncertainties $\alpha_s(n)$, which for our example implies that $n_{\text{eff}}^c \approx 5$ will have the smallest uncertainty α.

6.3 Noise Sensitivity of Continuous Effective Qubit Number

To test the sensitivity of n_{eff}^c to small changes in the underlying noise model, we parametrize our noise model with a continuous parameter $g \in \mathbb{R}$ as is done in Chapter 5. In Fig. 6.3 the results of measuring n_{eff}^c according to Alg. 2 with $n_\epsilon = 100$ for each $g \in \{0.8, 0.85, 0.9, \ldots, 1.4\}$ is plotted. The simulation data used is the same as in Fig. 5.2.

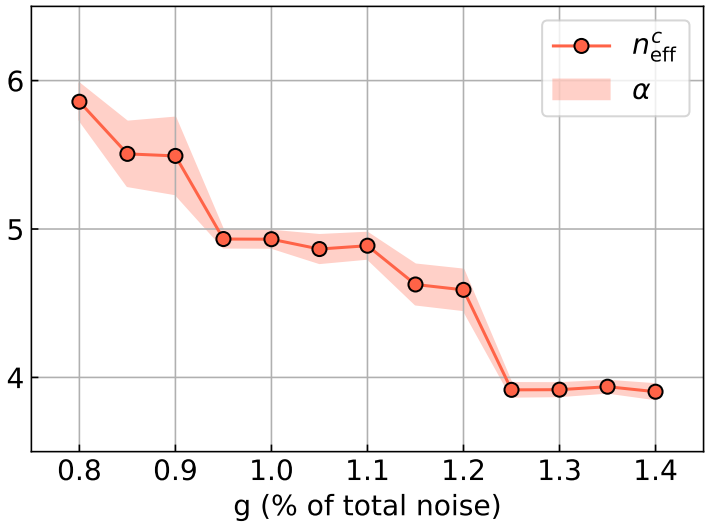


Figure 6.3: Measurement of n_{eff}^c for $g \in \{0.8, 0.85, 0.9, \ldots, 1.4\}$ and $n_\epsilon = 100$ using our custom noise model.

The abrupt changes at the plateaus of n_{eff} observed in Fig. 5.2 has indeed been smoothed out using the continuous extension n_{eff}^c as is seen in Fig. 6.3. It is also observed that α seems to be smallest when n_{eff}^c is approximately equal to an integer, and largest when n_{eff}^c lies in between two integers. The reason for this is the same as the one explaining that the data points in Fig. 6.2 with $n_{\text{eff}}^c \approx 5$ had the smallest uncertainties α.

6.4 Comparison With Discrete Effective Qubit Number

This section aims at comparing the effective qubit number n_{eff} as defined in Sec. 4.7 to its continuous extension as defined in Sec. 6.1. The main thing to note is that n_{eff}^c seems to have gained the desired property that small changes in the underlying model correspond to small changes in n_{eff}^c, as is seen by comparing the noise sensitivity tests in Fig. 5.2 and Fig. 6.3. This unfortunately comes at the cost of loosing several important properties of n_{eff}.

Firstly, it is observed in Sec. 6.2 that n_{eff}^c is not as system-robust as n_{eff}, which makes comparisons between measurements of n_{eff}^c on different devices more difficult. To counter this difficulty an uncertainty α for n_{eff}^c is computed in Sec. 6.1.

Secondly, the noise models underlying real quantum devices will, in general, vary slightly with time. The extension n_{eff}^c will be more sensitive to such variations than n_{eff}. To see this, note that the success score $S_n^c(D(n))$ depends directly on the numerical value of $\Delta_{\text{loss}}(n)$ which will change along with variations in

the underlying noise model, leading to changes in both n_{eff}^{c} and α. In contrast, the binary success criterion $S_n(D(n))$ only depends on whether or not the following holds,

$$\Delta_{\text{loss}}(n) + \alpha_{\delta}(n) < \delta_{\text{gain}}(n), \tag{6.4.1}$$

which depends to a lesser degree on the actual numerical value of $\Delta_{\text{loss}}(n)$, implying that n_{eff} is more stable against variations in the underlying noise model.

Lastly, the extension $n_{\text{eff}}^{c} \in \mathbb{R}$ does not have the same simple interpretation as n_{eff}, i.e., being the largest qubit count for which a device can run the test implementation of quantum phase estimation in Fig. 4.1 before noise "takes over". Since n_{eff} is supposed to be a single-number metric that is easy to read, share and compare, this matters a lot.

In conclusion, the properties lost when extending n_{eff} continuously to n_{eff}^{c} seems to outweigh the continuity gained concerning changes in the underlying noise model. It poses an interesting problem to see whether n_{eff} can be extended continuously in some other way, that avoids the loss of the properties mentioned above.

CHAPTER 7

Experiments on Quantum Hardware

In this chapter, the effective qubit number as defined in Sec. 4.7 will be tested on real quantum devices. Specifically, the two devices available on the free plan at IBM, which at the time of writing are Sherbrooke and Brisbane, as well as an anonymized device from the UK. All devices are based on a superconducting qubit architecture. The Sherbrooke and Brisbane devices from IBM are all 127 qubit Eagle r3 quantum processors. All relevant information regarding IBM's devices can be found on their online quantum computing platform. For the sake of convenience, the most essential information is presented in Table 7.1 alongside similar details for the anonymized device from the UK.

	Sherbrooke	Brisbane
Median $\sqrt{X}$ error	$2.429 \cdot 10^{-4}$	$2.504 \cdot 10^{-4}$
Median readout error	$2.124 \cdot 10^{-2}$	$1.782 \cdot 10^{-2}$
Median ECR error	$6.740 \cdot 10^{-3}$	$7.331 \cdot 10^{-3}$
Median T1	270.45μs	227.88μs
Median T2	211.03μs	137.67μs

	UK device
Median $\sqrt{X}$ error	$3.4 \cdot 10^{-3}$
Median readout error	$6.56 \cdot 10^{-2}$
Median $CNOT$ error	$4.17 \cdot 10^{-2}$
Median T_1	132.2μs
Median T_2	26.8μs

Table 7.1: Low-level information of the quantum devices used to test n_{eff} during this chapter. They all have the same basis gates $\{I, X, \sqrt{X}, R_z, ECR\}$. None of the devices has all-to-all connectivity.

There is a notable difference between IBM's devices and the anonymized UK device regarding qubit thermal relaxation times T_1 and T_2, as well as the median one- and two-qubit gate errors. If we let $n_{\text{eff}}(\text{Sherbrooke})$, $n_{\text{eff}}(\text{Brisbane})$, and $n_{\text{eff}}(\text{UK})$ be the effective qubit numbers measured for the Sherbrooke, Brisbane and UK devices respectively, then we expect,

$$\textbf{Hypothesis:} \quad n_{\text{eff}}(\text{Sherbrooke}) \approx n_{\text{eff}}(\text{Brisbane}) > n_{\text{eff}}(\text{UK}). \tag{7.0.1}$$

Besides testing this hypothesis, there is another important reason to test n_{eff} on hardware. Single-number metrics like n_{eff}, should enable scientists and industrial users to understand which quantum computing architectures outperform others. They do so by assigning a single number to each device, enabling a straightforward comparison between devices. Since $n_{\text{eff}} \in \mathbb{N}$, different devices with different underlying noise models might have equal effective qubit numbers, it is in this sense that n_{eff} classify quantum devices into classes of approximately equal quality. Comparing how n_{eff} classifies devices to other single-number metrics, e.g. quantum volume, enables us to understand how well n_{eff} distinguishes different devices.

7.1 Measurements of the Effective Qubit Number

The effective qubit number n_{eff} is measured on all three devices according to Alg. 1 with $n_\epsilon = 100$ for IBM's devices and $n_\epsilon = 70$ for the UK device. The results of the measurements are contained in Fig. 7.1, which immediately confirms the hypothesis in Eq. 7.0.1.

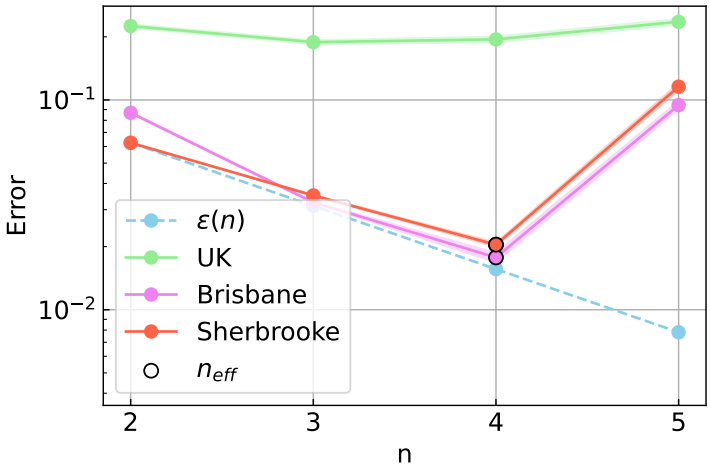


Figure 7.1: Measurements of n_{eff} according to Alg. 1 on the Sherbrooke and Brisbane devices at IBM as well the anonymized device in the UK. The solid lines are the means $\mu_\epsilon(n)$ and the shaded backgrounds are the corresponding standard errors $\pm\alpha_\epsilon(n)$ as defined in Eq. 4.6.8 and Eq. 4.6.16 respectively. The effective qubit number for the UK device is not visible since $n_{\text{eff}}(\text{UK}) = 1$.

In all experiments, data is collected for qubit counts $n \in \{2, 3, 4, 5\}$, which according to Eq. 4.7.3 results in a total number of QPE-calls of $n_{\text{QPE}} = 320\,000$ for each of IBM's devices and $n_{\text{QPE}} = 224\,000$ for the UK device. The difference in quality of data between devices in Fig. 7.1 can be understood by examining the empirical probability distributions $\hat{p}(n, n_s = 100)$ underlying each phase estimate $\phi_{\text{est.}}(n)$.

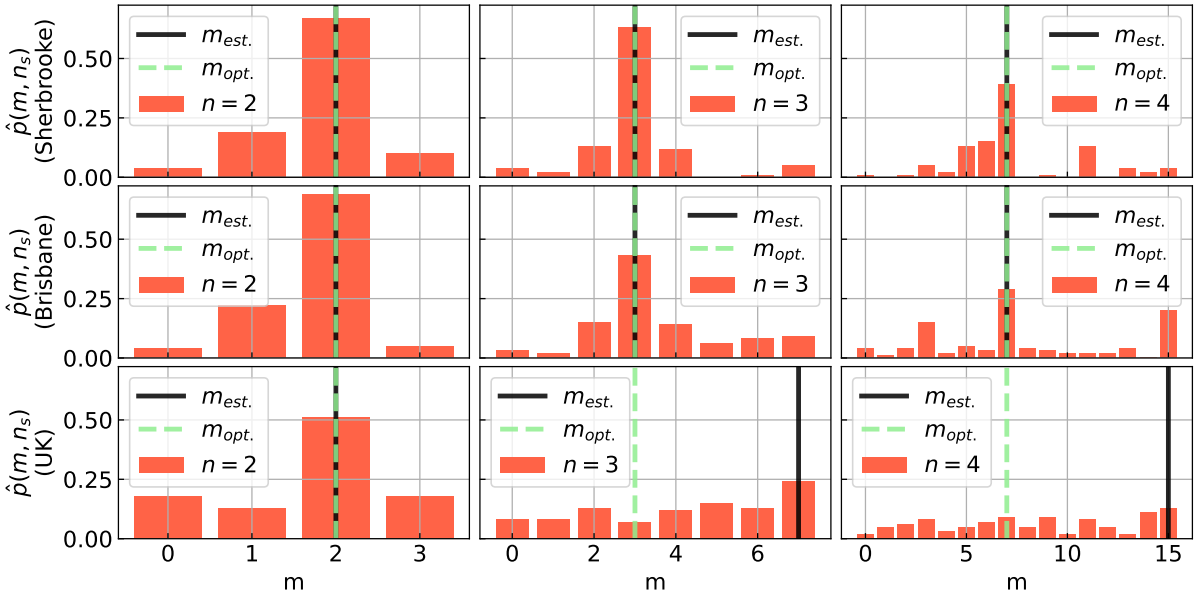


Figure 7.2: Examples of empirical probability distributions $\hat{p}(m, n_s = 100)$ with $n \in \{2, 3, 4\}$ and $\phi = \frac{5}{12}$ for all three devices computed according to Eq. 4.1.4, where $m_{\text{est.}} = \arg\max_m \{\hat{p}(m, n_s)\}$.

Empirical probability distributions with $\phi = \frac{5}{12}$, $n \in \{2, 3, 4\}$, and $n_s = 100$ is plotted in Fig. 7.2, where the modes of both the theoretical and empirical probability distributions are reported, i.e.,

$$m_{\text{est.}} = \arg\max_m \{\hat{p}(m, n_s)\} \quad \text{and} \quad m_{\text{opt.}} = \arg\max_m \{p(m)\}. \tag{7.1.1}$$

The empirical probability distributions for the UK device already seem to be uniformly distributed on $n = 3$ qubits. The error rates of the UK device are a lot larger than those of IBM's devices, implying that larger amounts of noise are present on the UK device for lower qubit counts, for which a uniform distribution of outcomes is expected, as is discussed in Sec. 2.3. These uniformly distributed outcomes explain the larger numerical errors observed in Fig. 7.2 for the UK device.

7.2 Comparison With Simulations

Firstly, note that the measurement $n_{\text{eff}}(\text{Sherbrooke}) = 4$ is one less than the measurements in Sec. 5.1 of n_{eff} on our custom noise model which imitates the behavior of the Sherbrooke device. The reason is that the custom noise model oversimplifies the noisy character of the Sherbrooke device, as it does not account for finite qubit connectivity or more complex types of noise, such as crosstalk. This results in the simulator having a larger effective qubit number than the real-world device.

Secondly, the distribution of data $\delta^i_{\text{loss}}(n) = \epsilon^i_{\text{est.}}(n) - \epsilon(n)$ obtained from the n_ϵ phase estimates for each $\phi \in \Phi$, is plotted for all devices and $n \in \{3, 4, 5\}$ in Fig. 7.3.

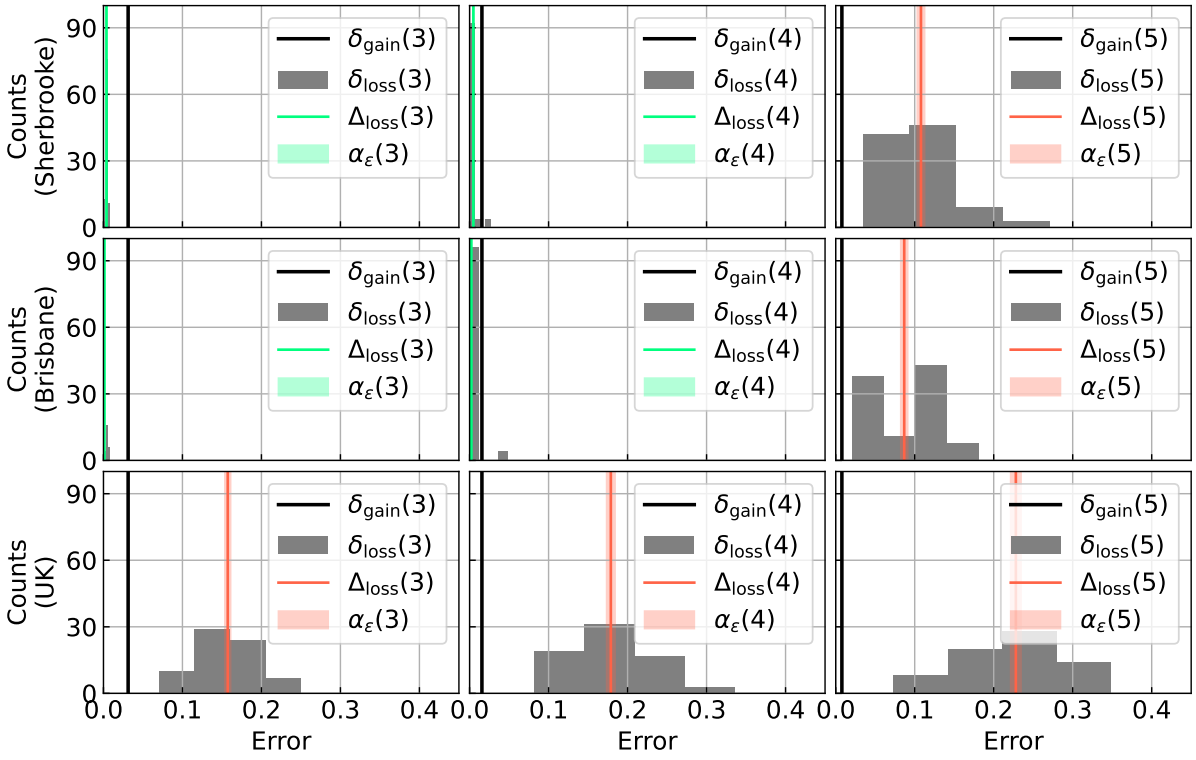


Figure 7.3: Distributions underlying data points in Fig. 7.1 at $n \in \{3, 4, 5\}$. The line $\Delta_{\text{loss}}(n)$ with shaded background $\pm\alpha_\delta(n)$ is colored green if noise has not yet "taken over", and red otherwise.

The distributions in Fig. 7.3 follow trends very similar to those observed in Fig. 4.15 for data obtained

from our custom noise model. The custom noise model thus appears to reproduce behavior similar to that observed experimentally, which assures us that behavior such as system robustness observed in simulations will likely carry over to experiments on real quantum devices.

7.3 Further Comparisons

Firstly, computing the continuous extension n_{eff}^{c} of the effective qubit number n_{eff} for the data in Fig. 7.1 results in the following values and uncertainties:

$$n_{\text{eff}}^{c}(\text{Sherbrooke}) = 3.57 \pm 0.04, \quad n_{\text{eff}}^{c}(\text{Brisbane}) = 3.43 \pm 0.08, \quad \text{and} \quad n_{\text{eff}}^{c}(\text{UK}) = 1 \pm 0 \quad (7.3.1)$$

Note that $n_{\text{eff}}^{c} < 4$ for both the Sherbrooke and Brisbane device, since $S_n^c(D(n)) < 1$ for all qubit counts $n \in \{2, 3, 4, 5\}$, while $n_{\text{eff}}^{c} = 1$ for the UK device, since $S_n^c(D(n)) = 0$ for all qubit counts $n \in \{2, 3, 4, 5\}$. This confirms that n_{eff}^{c} is more conservative than n_{eff} as discussed in Sec. 6.4.

Secondly, let us discuss some data that Ebbe obtained during his thesis. The data come from experiments on IBM's retired 127-qubit Osaka device, based on a superconducting qubit architecture, and AGT's 12-qubit Ibex device, based on a trapped-ion architecture [34]. The data consists of phase estimates $\phi_{\text{est.}}(n)$ as defined in Eq. 4.1.5, from which average prediction errors $\epsilon_{\text{est.}}^{\text{Ebbe}}(n)$ over Φ^{Ebbe} was computed according to Eq. 4.1.7. The results including $n_{\text{eff}}^{\text{Ebbe}}$ as defined in Eq. 4.1.8 is plotted in Fig. 7.4.

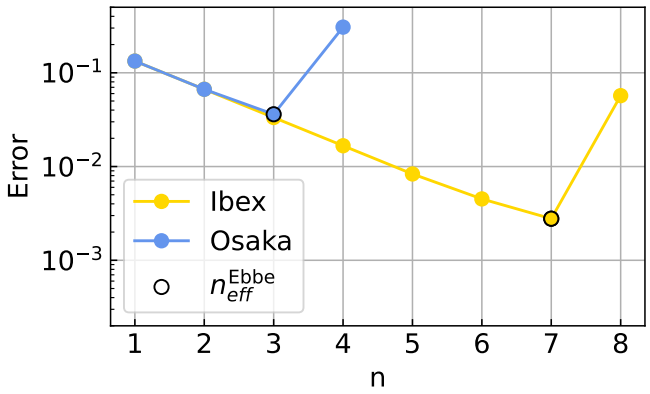


Figure 7.4: Data from Ebbe's thesis from experiments on IBM's Osaka device and AQT's Ibex device. Solid lines are average errors $\epsilon_{\text{est.}}^{\text{Ebbe}}(n)$ over Φ^{Ebbe} as defined in Eq. 4.1.7.

The key takeaway from Fig. 7.4 is that effective qubit numbers as defined by Ebbe, larger than $3-4$, exist on devices used one year ago. This indicates that the same should be true for our definition of the effective qubit number. Another takeaway concerns classical processing tools. He used a tool called Fire Opal, made by Q-Control [35], which employs hardware-aware mapping of logical qubits to physical qubits on hardware, error mitigation, and dynamical decoupling sequences —additional sequences of gates added to the circuit to counter phase damping and crosstalk. Using this, he increased $n_{\text{eff}}^{\text{Ebbe}}(\text{Osaka})$ from 3 to 4, thus proving that classical processing tools affect benchmarks like the effective qubit number.

Thirdly, the three IBM devices, Sherbrooke, Brisbane, and Osaka, as well as AQT's Ibex device, have a quantum volume of $V_Q = 128 = 2^7$. Quantum volume is thus unable to distinguish the four devices based on performance, i.e., it classifies them all into the same performance class. The effective qubit

number n_{eff} also classifies the Sherbrooke and Brisbane device into the same performance class. Even so, the measurement

$$n_{\text{eff}}^{\text{Ebbe}}(\text{Ibex}) = 7 \neq 3 = n_{\text{eff}}^{\text{Ebbe}}(\text{Osaka}) \tag{7.3.2}$$

indicates a significant difference in performance on quantum phase estimation between IBM's 127-qubit devices and AQT's Ibex device. It would thus be interesting to measure $n_{\text{eff}}(\text{Ibex})$, since the discussion above indicates that it might be larger than the values of IBM's Sherbrooke and Brisbane devices, i.e.,

$$n_{\text{eff}}(\text{Ibex}) \overset{?}{>} n_{\text{eff}}(\text{Sherbrooke}) = n_{\text{eff}}(\text{Brisbane}). \tag{7.3.3}$$

If this is so, then n_{eff} would separate devices into smaller performance classes, i.e., it would be better at distinguishing different devices.

Lastly, a second measurement of n_{eff} on IBM's Sherbrooke and Brisbane devices has been performed. The two measurements on the Brisbane device are performed within $10 - 20$ minutes of each other, while the ones on the Sherbrooke device are performed about 4.5 hours apart. The data is contained in Fig. 7.5.

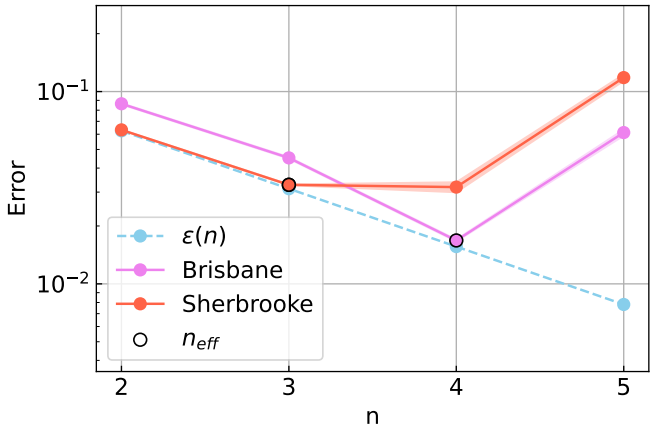


Figure 7.5: A second measurement of n_{eff} according to Alg. 1 on the Sherbrooke and Brisbane devices at IBM.

The results appear to be almost unchanged for the Brisbane device. In contrast, differences appear for the Sherbrooke device compared to Fig. 7.1, i.e.,

$$n_{\text{eff}}(\text{Sherbrooke}) = 4 \quad \longrightarrow \quad n_{\text{eff}}(\text{Sherbrooke}) = 3. \tag{7.3.4}$$

During the time following a device calibration, device characteristics such as qubit relaxation times and gate errors may drift in value. Alongside the difference in the periods over which the experiments are performed, this might explain the observed difference between the two devices. Due to the system-robustness of n_{eff}, it seems unlikely that such differences could occur by chance, but to conclude anything, more than two measurements will be needed needed.

CHAPTER 8

Discussion and Outlook

The effective qubit number n_{eff} can be interpreted as the largest qubit count n for which a given device can run the test implementation in Fig. 4.1 before noise "takes over" as defined in Eq. 4.6.18. Importantly, n_{eff} only measures the performance of a device on that test implementation, and can thus not be used to extrapolate the performance of that device on arbitrary implementations of quantum phase estimation. In conclusion, the intended use of n_{eff} is to compare the computing power of quantum devices to determine which device to use, and not to decide, for a specific device, how many qubits that should be used on quantum phase estimation, to obtain the best possible results.

The definition of $n_{\text{eff}}^{\text{Ebbe}}$ in terms of a minimum, whose existence is unknown, has been replaced in n_{eff} with an iterative definition in terms of a success criterion, making it more well-defined. Despite this, n_{eff} seems to be placed at the minima of the mean prediction error $\mu_\epsilon(n)$ in all experiments and simulations. To see why, note that,

$$\mu_\epsilon(n) \approx \epsilon(n) = \frac{1}{2^{n+2}}$$

for small qubit counts n due to low amounts of noise. Then at the qubit count n where noise takes over according to the definition in Eq. 4.6.18, the mean prediction error $\mu_\epsilon(n)$ increases sharply $\mu_\epsilon(n) \gg \epsilon(n)$ - this behavior is observed in both Fig. 7.1 and Fig. 4.13. Given such behavior, the effective qubit number n_{eff} will be equal to $\arg\min_n\{\mu_\epsilon(n)\}$. Since such behavior seems to be very common, it follows that in a lot of cases, n_{eff} will be equal to the qubit count n for which the best numerical accuracy is obtained on the test implementation in Fig. 4.1.

The effective qubit number $n_{\text{eff}}^{\text{Ebbe}}$ as defined in Sec. 4.1 did not satisfy all the criteria of an ideal benchmark as discussed in Sec. 4.1. This motivated the construction of an effective qubit number n_{eff}, satisfying the criteria of an ideal benchmark to a larger extend than $n_{\text{eff}}^{\text{Ebbe}}$. The result turned out to be more well-defined, well-motivated, and system-robust. Even so, satisfying the criteria of an ideal benchmark is not a binary decision, and most benchmarks do not excel in all criteria. The effective qubit number n_{eff} has an issue regarding implementation-robustness as discussed in Sec. 4.6. Let us quickly recap why this issue arises. The effective qubit number is measured according to Alg. 1 and is a function of the classical data,

$$D(n) := \{\phi_{\text{est.}}^i(n) \mid \phi \in \Phi \text{ and } i \in \{1, 2, \ldots, n_\epsilon\}\}$$

obtained by running the test implementation of quantum phase estimation in Fig. 4.1 for $\phi \in \Phi$ and

$n \in \{2, 3, \ldots, n'\}$ for suitable n'. Each phase estimate $\phi^i_{\text{est.}}(n)$ is computed according to Eq. 4.1.5 from an empirical probability distribution $\hat{p}^i(m, n_s)$ constructed from the classical data obtained from $n_s = 100$ QPE-calls. After all data $D(n)$ has been collected, it is possible to cheat by interchanging counts between the empirical probability distributions, $\hat{p}^i(m, n_s)$ and $\hat{p}^j(m, n_s)$, of two different phase estimates, $\phi^i_{\text{est.}}(n)$ and $\phi^j_{\text{est.}}(n)$, resulting different phase estimates. This could change the mean prediction error $\mu_\epsilon(n)$ and thus also n_{eff}. This involves data manipulation, which is strictly prohibited. Rules against issues like this is stated at the bottom of 1 and should always be followed strictly when measuring n_{eff}. It poses an interesting problem to construct an alternative definition of n_{eff}, for which such data manipulation is not possible, since this would improve its implementation-robustness, taking it one step closer to the definition of an ideal benchmark.

The effective qubit number n_{eff} is an application-oriented, holistic single-number metric. As such, it should be compared to benchmarks that fall into the same category. A widely used single-number metric is the quantum volume, which, being a randomized benchmark, falls outside this category. In contrast, the number of algorithmic qubits #AQ defined by IonQ [9] is also an application-oriented, holistic single-number metric - it is discussed in greater depth in Sec. 3.1.
The main difference between #AQ and n_{eff} is that the former is based on an entire suite of quantum algorithms, while the latter is based solely on quantum phase estimation. Some quantum algorithms might perform better in the absence of specific types of noise or on devices with particular qubit topologies. Single-number metrics based on a single algorithm, like n_{eff}, are thus naturally biased when used to determine which qubit architectures are better. The number of algorithmic qubits avoids this problem to a larger degree. Even so, specific algorithms in the suite defining #AQ might dominate in its measurement due to a larger susceptibility towards noise or larger gate counts for specific problem sizes. It would be interesting to see which devices n_{eff} favors, by measuring it on more quantum devices.
Besides this, n_{eff} is defined in terms of numerical accuracy, which is a natural quantity to optimize from a practical perspective, while #AQ is defined in terms of the slightly more abstract concept of fidelity. This makes n_{eff} easier to interpret and more well-motivated, since numerical accuracy is in most practical cases the main quantity of interest. Ultimately, the benchmark used to compare hardware depends on the purpose of the comparison. If you want to test for progress towards quantum utility, quantum volume might be a better option. If you are testing for computing power within quantum chemistry, such as to determine the ground states of molecules, then n_{eff} might be the better option. At the same time, #AQ might be the better option if you are testing for overall computing power.

The last topic for discussion is the continuous extension n^c_{eff} of the effective qubit number n_{eff} defined in Chapter 6. It was found that by extending n_{eff} continuously, small changes in noise implies only slight variations in the effective qubit number. Unfortunately, this comes at the cost of losing system-robustness, stability under minor variations in noise over time, and having a simple interpretation like n_{eff}, as being the largest qubit count for which a test implementation of quantum phase estimation can be run before noise "takes over". It poses an interesting problem to see whether n_{eff} can be extended continuously in some other way, that avoids the loss of the properties mentioned above.

CHAPTER 9

Conclusion

The primary objective of this thesis was to develop an application-oriented, holistic single-number metric based on quantum phase estimation, which is easy to read, share, and compare. The developed benchmark should be well-defined, well-motivated, system-robust, efficient, and implementation-robust. The secondary objective was to test it on hardware and compare it with other single-number metrics.

In this thesis, I have developed a novel, application-oriented, and holistic single-number metric based on quantum phase estimation, called the effective qubit number, n_{eff}. It has the straightforward interpretation of being the largest number of qubits, on which a device can run the test implementation of quantum phase estimation in Fig. 4.1 before noise "takes over", where noise "taking over" is defined in a precise mathematical way in Eq. 4.6.18. The effective qubit number is formulated in Sec. 4.7 both as a simple equation and as an algorithm summarizing all the steps involved in its measurement. Below the algorithmic formulation in Alg. 1, three rules are written down that should be followed strictly when measuring n_{eff}. These rules primarily concern the use of classical processing tools, such as transpilers and error mitigation, all of which are permitted under the condition of transparency regarding the specific optimization tools employed. This holistic view ensures that n_{eff} captures the computing power of the entire quantum computing stack, making it a more relevant benchmark for near-term devices.

Following the definition of the effective qubit number, it is argued to be well-defined and well-motivated, and shown to be efficient and quite system-robust as seen in Sec. 5.1. Additionally, it is found to be stable against slight variations in the underlying noise model in Section 5.2, which manifests itself numerically as the plateaus observed in Fig. 5.2. The exception to this statement appears to happen right at the edges of those plateaus, where slight variations in the underlying noise model corresponds to a rapid change in the effective qubit number, $n_{\text{eff}} \longrightarrow n_{\text{eff}} \pm 1$. This motivated the development of a continuous extension $n_{\text{eff}}^c \in \mathbb{R}$ of n_{eff} in Chapter 6, as to ensure that slight variations in noise always corresponds to small changes in n_{eff}^c. Measurements of this continuous extension should be done while following the same set of rules as for n_{eff} given below Alg. 1. Unfortunately, the gain in continuity with respect to noise observed in Fig. 6.3 comes at the cost of losing system-robustness, stability under minor drifts in the parameters of devices between calibrations, and having a straightforward interpretation, such as n_{eff}. Ultimately, the gains do not seem to outweigh the losses.

Finally, measurements of n_{eff} on quantum hardware have been performed. Specifically, on two of IBM's 127 qubit Eagle r3 quantum computers, called Sherbrooke and Brisbane, and on an anonymized quantum computer from the UK. The results are $n_{\text{eff}}(\text{Sherbrooke}) = n_{\text{eff}}(\text{Brisbane}) = 4$ and $n_{\text{eff}}(\text{UK}) = 1$ respectively. The observed differences in n_{eff} seems reasonable considering the low-level data of the devices in Tab. 7.1. The two devices from IBM both have a quantum volume of 128 [2], initially indicating that devices with equal quantum volumes have equal effective qubit numbers. In contrast, measurement data from a previous master's student [32] seems to indicate that AQT's Ibex device, which also has a quantum volume of 128, might have an effective qubit number larger than those of Sherbrooke and Brisbane. If this is the case, then n_{eff} might be depend more strongly on device characteristics like qubit lifetimes and gate errors than quantum volume.

The effective qubit number is a novel contribution to the sparse collection of application-oriented single-number metrics, since it offers a functionally meaningful assessment of quantum hardware performance, in addition to being easy to read, share, and compare. Unlike abstract benchmarks like quantum volume, n_{eff} measures computing power for problems of real-world significance, particularly in quantum chemistry, where quantum phase estimation is an essential subroutine, e.g., in determining the ground-state energy of molecules like FeMoCo [36]. As quantum devices advance, application-oriented benchmarks will be essential in guiding their development toward impactful applications. Lastly, during development, the criteria of a good benchmark outlined in Sec. 3.2 have routinely been revisited to ensure that n_{eff} is build to guide and not inhibit progress toward quantum utility.

An obvious limitation of n_{eff} is that it only measures the performance of a device on the test implementation of quantum phase estimation in Fig. 4.1, and can thus not be used to extrapolate the performance of that device on arbitrary implementations of quantum phase estimation. Furthermore, the computing power of a device for quantum phase estimation may not directly translate to other quantum algorithms. Even so, quantum phase estimation serves as a proxy for quantum advantage and is a vital subroutine within quantum chemistry, making it a well-motivated basis for algorithmic benchmarks. Furthermore, it seems reasonable to believe that if one device significantly outperforms another on the chosen test implementation of quantum phase estimation, then it will also outperform on general implementations.

Important future work include attempts to redefine n_{eff}, such that the problems regarding implementation-robustness discussed in Chapter 8, are avoided altogether. Doing so successfully, would take n_{eff} one step closer to satisfying the definition of an ideal benchmark in Sec. 3.2. Other topics for future work include, measuring n_{eff} on larger varieties of hardware, thereby gaining insight into which hardware architectures it tends to favor, as well as providing a continuous extension of n_{eff} without losing its key properties, as to avoid regions of unreasonable deviations in n_{eff} under minor variations in noise, such as at the edges of the plateaus in Fig. 5.2. Lastly, any universal quantum computer can implement quantum phase estimation, so extending n_{eff} to measurement-based quantum computers should be possible. Achieving this successfully would be valuable, as it is not yet known which quantum computing architectures will dominate in the future.

APPENDIX A

Properties of Quantum Phase Estimation

Given a unitary U and an eigenstate $|u\rangle$ with eigenvalue $e^{i2\pi\phi}$, we found that the quantum phase estimation algorithm on n qubits can provide us with approximations to ϕ of the form $\frac{m}{2^n}$ for $m = 0, 1, \ldots, 2^n - 1$. The probability of measuring each of these outcomes is given in Eq. 2.2.13, which is conveniently restated here:

$$p(m) = \frac{1}{2^{2n}} \left| \frac{1 - e^{i2\pi 2^n(\phi - \frac{m}{2^n})}}{1 - e^{i2\pi(\phi - \frac{m}{2^n})}} \right|^2 = \frac{1}{2^{2n}} \frac{\sin^2(\pi 2^n \delta_n)}{\sin^2(\pi \delta_n)},$$

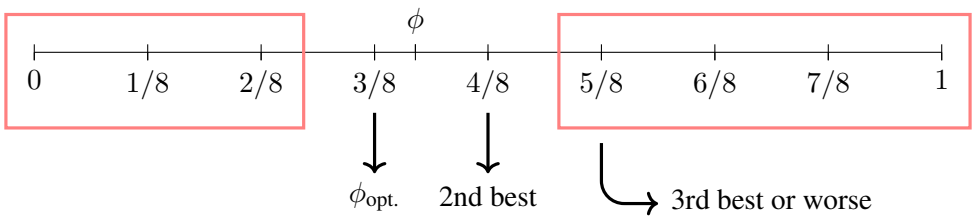
where $p(m) = 1$ if ϕ takes on this fractional form $\phi = \frac{m}{2^n}$. There will always be an optimal approximation $\phi_{\text{opt.}} = \frac{m_{\text{opt.}}}{2^n}$ that minimizes the distance to our input ϕ,

$$\phi_{\text{opt.}} = \frac{1}{2^n} \min_{m \in \{0,1,\ldots,2^n-1\}} |\phi - \frac{m}{2^n}|.$$

It was claimed in Chapter 2 that $\phi_{\text{opt.}}$ is the most probable outcome of the quantum phase estimation algorithm, in the sense that,

$$p(m_{\text{opt.}}) > p(m), \quad m \neq m_{\text{opt.}}.$$

The goal of this Appendix is to prove this statement. To see how we will do this, we will consider a quick example. Say we have $n = 3$ qubits, and choose $\phi = 0.4$. Then the possible phase approximations are $\frac{0}{2^3} = 0, \frac{1}{2^3} = \frac{1}{8}, \ldots, 1$. These are marked in the figure below.



In our case where $\phi = 0.4$, the optimal phase approximation $\phi_{\text{opt.}} = \frac{3}{8}$. The 2nd best approximation to ϕ is $\frac{4}{8}$, and the 3rd best approximation or worse are $\frac{2}{8}$, $\frac{5}{8}$, $\frac{1}{8}$ and so on. To ease notation, define $\phi_{\text{opt.}}(i) = \frac{m_{\text{opt.}}(i)}{2^n}$ the be the ith best approximation to ϕ, such that $\phi_{\text{opt.}}(1) = \phi_{\text{opt.}}$ is the best, $\phi_{\text{opt.}}(2) = \frac{4}{8}$ the 2nd best, and so on.

The first leap towards a proof follows from some great online lecture notes at IBM made by John Watrous [37]. In these notes, he proves using simple geometrical arguments that,

$$p(m_{\text{opt.}}) \geq \frac{4}{\pi^2} \quad \text{and} \quad p(m_{\text{opt.}}(i)) \leq \frac{1}{4},\ i \geq 3.$$

In words, the 3rd best approximation or worse is always less probable than the best one. All that is left to check for ourselves is whether it is possible that,

$$p(m_{\text{opt.}}) \overset{?}{<} p(m_{\text{opt.}}(2)),$$

i.e., can the 2nd-best approximation be more likely than the best one? The answer turns out to be no. To get some inspiration as to why this is so, we consider the special case sketched previously with $n = 3$ and $\phi = 0.4$. If we vary ϕ within the interval $[\frac{3}{8}, \frac{4}{8}]$, then we can investigate how the probabilities $p(m = 3)$ and $p(m = 4)$ changes. When ϕ is in the lower half of the interval $m_{\text{opt.}} = 3$, and when ϕ is in the upper half of the interval $m_{\text{opt.}} = 4$. For the best approximation to be the most likely one, we need to have that $p(m = 3) > p(m = 4)$ on the lower half of the interval and $p(m = 4) > p(m = 3)$ on the upper half of the interval.

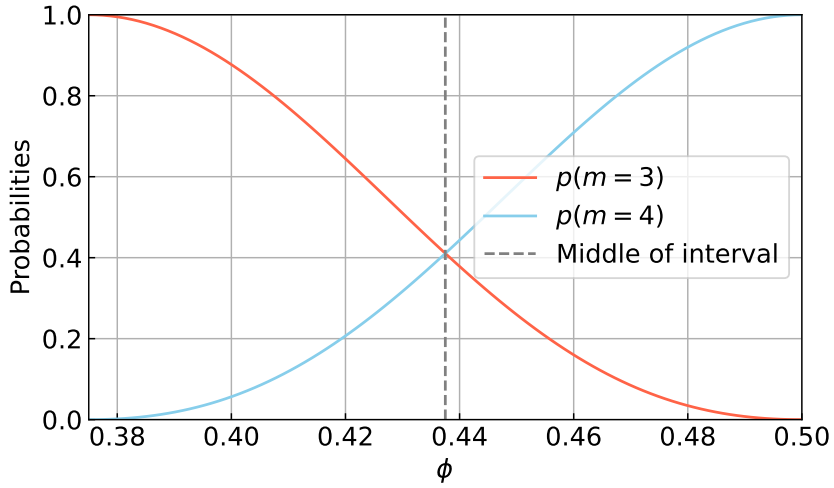


Figure A.1: Visualization of $p(m)$ as given in Eq. 2.2.13, for $m = 3, 4$ and ϕ is varied in the interval $[\frac{3}{8}, \frac{4}{8}]$.

In Fig. A.1 we indeed observe what we hoped for! By playing around with this, making similar plots for different values of n, and for different intervals $[\frac{m}{2^n}, \frac{m+1}{2^n}]$, you will get very similar graphs, which strongly indicates that our statement is true. Even so, we still need to prove it.

In the following, we will explicitly write $p(m, \phi)$ as a function of ϕ, to clarify the arguments. The fact that all graphs like Fig. A.1 for different intervals $[\frac{m}{2^n}, \frac{m+1}{2^n}]$ look the same is a manifestation of the fact that,

$$p(m, \phi) = \frac{1}{2^{2n}} \frac{\sin^2(\pi 2^n \delta_n)}{\sin^2(\pi \delta_n)},$$

only depends on the difference $\delta_n = \phi - \frac{m}{2^n}$. More precisely,

$$p(m, \phi) = p(m - 1, \phi - \frac{1}{2^n}).$$

Due to this, we might as well restrict ourselves to the case where $\phi \in [0, \frac{1}{2^n}]$. Another feature of all these graphs is that they seem to be symmetrical around the vertical line going through the middle of the interval. In terms of Fig. A.1, it appears that the probability $p(m = 3, \phi)$ is the same as the probability $p(m = 4, \frac{4}{8} - \phi)$, i.e., given a point on the red graph at ϕ, the point on the blue graph you get by reflecting ϕ around the middle axis ($\frac{4}{8} - \phi$) is of equal height. To show it in general, remember that we might as well assume that $\phi \in [0, \frac{1}{2^n}]$:

$$p(m = 1, \frac{1}{2^n} - \phi) = \frac{1}{2^{2n}} \frac{\sin^2(\pi 2^n(\frac{1}{2^n} - \phi - \frac{1}{2^n})))}{\sin^2(\pi(\frac{1}{2^n} - \phi - \frac{1}{2^n}))} = \frac{1}{2^{2n}} \frac{\sin^2(\pi 2^n(\phi - \frac{0}{2^n})))}{\sin^2(\pi(\phi - \frac{0}{2^n}))} = p(m = 0, \phi).$$

This reflection symmetry is the last hint needed to finish the proof. To see why, note that it immediately implies that,

$$p(m = 0, \phi = \frac{1}{2}\frac{1}{2^n}) = p(m = 1, \phi = \frac{1}{2}\frac{1}{2^n}).$$

We further have that $p(m = 0, \phi)$ starts at 1 and $p(m = 1, \phi)$ starts at 0 at $\phi = 0$, they meet in the middle at $\phi = \frac{1}{2}\frac{1}{2^n}$, after which they cross and end at 0 and 1 respectively at $\phi = \frac{1}{2^n}$. From graphs like in Fig. A.1 it seems to be the case that $p(m = 0, \phi)$ is strictly decreasing on $[0, \frac{1}{2^n}]$ and that $p(m = 1, \phi)$ is strictly increasing on the same interval. If this is so, then we immediately get that the best approximation is always the most probable one, and we are done.

By reflection symmetry it follows that if $p(m = 0, \phi)$ is strictly decreasing on $[0, \frac{1}{2^n}]$, then $p(m = 1, \phi)$ is strictly increasing on the same interval. Let's prove that $p(m = 0, \phi)$ is strictly decreasing by induction in the number of qubits n. The base case $n = 1$ holds since,

$$p(m = 0, \phi) = \frac{1}{4}\frac{\sin(2\pi\phi)^2}{\sin(\pi\phi)^2} = \frac{\sin(\pi\phi)^2}{\sin(\pi\phi)^2}\cos(\pi\phi)^2 = \cos(\pi\phi)^2$$

which is strictly decreasing for $\phi \in [0, \frac{1}{2}]$. Now, assume that case$(n-1)$ holds, and let's prove that case(n) holds:

$$\begin{aligned} p(m = 0, \phi) &= \frac{1}{2^{2n}}\frac{\sin(2^n\pi\phi)^2}{\sin(\pi\phi)^2} \\ &= \frac{1}{2^{2n}}\frac{\sin\big(22^{n-1}\pi\phi\big)^2}{\sin(\pi\phi)^2} \\ &= (\frac{1}{2^{2(n-1)}}\frac{\sin\big(2^{n-1}\pi\phi\big)^2}{\sin(\pi\phi)^2})\cos\big(2^{n-1}\pi\phi\big)^2. \end{aligned}$$

By induction assumption, the first term is strictly decreasing on $[0, \frac{1}{2^n}] \subset [0, \frac{1}{2^{n-1}}]$, and the second term is also strictly decreasing on the same interval since $2^{n-1}\pi\phi \in [0, \frac{\pi}{2}]$. This finishes the induction argument, and we have proved our desired result.

APPENDIX **B**

Constructing a Custom Noise Model

The goal of this Appendix is to construct a realistic noise model using the Qiskit SDK [26], which can be used throughout the thesis to perform simulations that test results and ideas. Gates on real-world quantum devices are noisy, which we model using depolarization channels,

$$\rho \to (1-\lambda)\rho + \frac{\lambda}{2^n}\mathbb{I}, \tag{B.0.1}$$

where λ determines the rate at which errors occur. Furthermore, qubits interact with the environment, leading to amplitude and phase damping, happening over the characteristic time-scales T_1 and T_2 respectively [17]. Qubit information is thus lost over the finite time it takes to implement any given quantum gate. These types of noise are modeled using thermal relaxation quantum channels [17].

The noise model will consist of ten fully connected qubits. To make it realistic, device characteristics are borrowed from Qiskit's FakeSherbrooke mock-backend, which imitates IBM's Sherbrooke device [38], including its native gates,

$$\mathcal{B} = \{\mathbb{I}, X, \sqrt{X}, R_z, \text{ECR}, \text{Measure}\}. \tag{B.0.2}$$

Measurements have been included explicitly in $\mathcal{B}$ to avoid restating everything in this chapter for measurements individually. Errors for R_z gates will be ignored since it is a virtual gate. To model gate errors, each basis gate $U \in \mathcal{B}$ is modulated by a depolarization channel, with parameter $\lambda(U)$ equal to the average gate error on the Sherbrooke device. The values used are listed below:

$$\lambda(\mathbb{I}) = \lambda(X) = \lambda(\sqrt{X}) = 4.3 \cdot 10^{-4},\ \lambda(\text{ECR}) = 5.2 \cdot 10^{-2},\ \text{and } \lambda(\text{Measure}) = 2.7 \cdot 10^{-2}. \tag{B.0.3}$$

Gate implementation times $T(U)$ for each $U \in \mathcal{B}$ are chosen to be equal to the given value on the Sherbrooke device and are listed below:

$$T(\mathbb{I}) = T(X) = T(\sqrt{X}) = 56.8\text{ns},\ T(\text{ECR}) = 540.6\text{ns},\ \text{and } T(\text{Measure}) = 1.2\mu\text{s}. \tag{B.0.4}$$

Qubit decoherence is modeled using thermal relaxation quantum channels with parameters T_1 and T_2 equal to the average values over all qubits of the Sherbrooke device. These values are listed below:

$$\mu(T_1) = 271.7\mu\text{s} \quad \text{and} \quad \mu(T_2) = 188.2\mu\text{s}. \tag{B.0.5}$$

All this information is retrieved directly from Qiskit's FakeSherbrooke mock-backend within the Python program. All values are listed to one decimal precision.

The noise model will be parametrized by a parameter $g \in \mathbb{R}^+$ as follows,

$$\lambda(U) \to g \cdot \lambda(U) \quad \text{and} \quad T_i \to \frac{1}{g} \cdot T_i. \tag{B.0.6}$$

This parametrization will allow us to test concepts under slight variations in the noise strength of the simulator. This noise model is likely to perform better than IBM's Sherbrooke device, due to the naive modeling of gate errors using depolarization channels and the absence of complex noise characteristics, such as crosstalk.

Bibliography

[1] R. Barends et al. “Superconducting quantum circuits at the surface code threshold for fault tolerance”. In: *Nature* 508.7497 (Sept. 2014), pp. 500–503. ISSN: 1476-4687. DOI: 10.1038/nature13171. URL: http://dx.doi.org/10.1038/nature13171.

[2] IBM. *IBM Quantum Computers*. https://quantum.ibm.com/services/resources, Last accessed on 07-06-2025. 2025.

[3] AQT. *AQT Quantum Computers*. https://www.aqt.eu/products/, Last accessed on 07-06-2025. 2025.

[4] IonQ. *IonQ Quantum Computers*. https://ionq.com/quantum-systems/compare, Last accessed on 07-06-2025. 2025.

[5] Quantinuum. *Quantinuum System Model H2 Quantum Computer*. https://www.quantinuum.com/products-solutions/quantinuum-systems/system-model-h2, Last accessed on 07-06-2025. 2025.

[6] E. Knill et al. “Randomized benchmarking of quantum gates”. In: *Physical Review A* 77.1 (Jan. 2008). ISSN: 1094-1622. DOI: 10.1103/physreva.77.012307. URL: http://dx.doi.org/10.1103/PhysRevA.77.012307.

[7] Andrew W. Cross et al. “Validating quantum computers using randomized model circuits”. In: *Physical Review A* 100.3 (2019). ISSN: 2469-9934.

[8] Timothy Proctor et al. “Measuring the capabilities of quantum computers”. In: *Nature Physics* 18.1 (Dec. 2021), pp. 75–79. ISSN: 1745-2481. DOI: 10.1038/s41567-021-01409-7. URL: http://dx.doi.org/10.1038/s41567-021-01409-7.

[9] IonQ. *Algorithmic Qubits: A Better Single-Number Metric*. https://ionq.com/resources/algorithmic-qubits-a-better-single-number-metric, Last accessed on 05-04-2025. 2025.

[10] Timothy Proctor et al. “Benchmarking Quantum Computers”. In: *Nature Reviews Physics 7* 7 (2025), pp. 105–118.

[11] Richard P. Feynman. “Simulating Physics with Computers”. In: *International Journal of Theoretical Physics* 21.6 (June 1, 1982), pp. 467–488. ISSN: 1572-9575. DOI: 10.1007/BF02650179.

[12] Charles H. Bennett and Gilles Brassard. “Quantum cryptography: Public key distribution and coin tossing”. In: *Theoretical Computer Science* 560 (Dec. 2014), pp. 7–11. ISSN: 0304-3975. DOI: 10.1016/j.tcs.2014.05.025. URL: http://dx.doi.org/10.1016/j.tcs.2014.05.025.

[13] David Deutsch. “Quantum theory, the Church–Turing principle and the universal quantum computer”. In: *Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences* 400 (1985), pp. 117–97. URL: https://api.semanticscholar.org/CorpusID:1438116.

[14] Lov K. Grover. *A fast quantum mechanical algorithm for database search*. 1996. arXiv: quant-ph/9605043 [quant-ph]. URL: https://arxiv.org/abs/quant-ph/9605043.

[15] Peter W. Shor. “Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer”. In: *SIAM Journal on Computing* 26.5 (Oct. 1997), pp. 1484–1509. ISSN: 1095-7111. DOI: 10.1137/s0097539795293172. URL: http://dx.doi.org/10.1137/S0097539795293172.

[16] Michael A. Nielsen. “Cluster-state quantum computation”. In: *Reports on Mathematical Physics* 57.1 (Feb. 2006), pp. 147–161. ISSN: 0034-4877. DOI: 10.1016/s0034-4877(06)80014-5. URL: http://dx.doi.org/10.1016/S0034-4877(06)80014-5.

[17] Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press, 2010.

[18] Artur Ekert et al. *Introduction to Quantum Information Science*. Lecture notes. 2024.

[19] Christopher C. Gerry and Peter L. Knight. *Introductory Quantum Optics*. Cambridge University Press, 2005.

[20] QuEra. *QuEra Aquila Quantum Computer*. https://www.quera.com/aquila, Last accessed on 08-06-2025. 2025.

[21] Atom Computing. *Atom Comptutings Quantum Computer*. https://atom-computing.com/quantum-computing-technology/, Last accessed on 08-06-2025. 2025.

[22] Xanadu. *Xanadu X-series Quantum Computer*. https://www.xanadu.ai/products/x-series, Last accessed on 08-06-2025. 2025.

[23] National Academies of Sciences, Engineering, and Medicine. *Quantum Computing: Progress and Prospects*. Washington, DC: The National Academies Press, 2019. ISBN: 978-0-309-48424-5. DOI: 10.17226/25196. URL: https://doi.org/10.17226/25196.

[24] Frederic Chong, Diana Franklin, and Margaret Martonosi. “Programming languages and compiler design for realistic quantum hardware”. In: *Nature* 549 (Sept. 2017), pp. 180–187. DOI: 10.1038/nature23459.

[25] John Preskill. “Quantum Computing in the NISQ era and beyond”. In: *Quantum* 2 (Aug. 2018), p. 79. ISSN: 2521-327X. DOI: 10.22331/q-2018-08-06-79. URL: http://dx.doi.org/10.22331/q-2018-08-06-79.

[26] IBM. *Qiskit SDK*. https://www.ibm.com/quantum/qiskit, Last accessed on 07-06-2025. 2025.

[27] A. Blais et al. “Circuit Quantum Electrodynamics”. In: *Reviews of Modern Physics* 93 (2021), p. 025005. DOI: 10.1103/RevModPhys.93.025005.

[28] Jan Frahm. *Lie groups*. Lecture notes. 2020.

[29] Christoph Dankert et al. “Exact and approximate unitary 2-designs and their application to fidelity estimation”. In: *Physical Review A* 80.1 (July 2009). ISSN: 1094-1622.

[30] Scott Aaronson and Lijie Chen. *Complexity-Theoretic Foundations of Quantum Supremacy Experiments*. 2016. arXiv: 1612.05903 [quant-ph]. URL: https://arxiv.org/abs/1612.05903.

[31] Quantinuum. *Debunking algorithmic qubits*. https://www.quantinuum.com/blog/debunking-algorithmic-qubits, Last accessed on 08-06-2025. 2024.

[32] Ebbe Maru Storm. "The effective number of qubits". MA thesis. Aarhus University, Department of Physics and Astronomy, 2024.

[33] Ifan G. Hughes and Thomas P. A. Hase. *Measurements and Their Uncertainties: A Practical Guide to Modern Error Analysis*. Oxford, 2010.

[34] AQT. *IBEX Q1*. https://www.aqt.eu/products/ibex-q1/, Last accessed on 06-06-2025. 2025.

[35] Q-Control. *Fire Opal by Q-Control*. https://docs.q-ctrl.com/fire-opal, Last accessed on 06-06-2025. 2025.

[36] Markus Reiher et al. "Elucidating reaction mechanisms on quantum computers". In: *Proceedings of the National Academy of Sciences* 114.29 (2017), pp. 7555–7560. ISSN: 0027-8424. DOI: 10.1073/pnas.1619152114. URL: https://www.pnas.org/content/114/29/7555.

[37] John Watrous at IBM. *Fundamentals of Quantum Algorithms*. https://learning.quantum.ibm.com/course/fundamentals-of-quantum-algorithms, Last accessed on 15-05-2025. 2025.

[38] IBM. *IBM's Sherbrooke device*. https://quantum.ibm.com/services/resources?system=ibm_sherbrooke, Last accessed on 07-06-2025. 2025.