Approaches for Utilizing Quantum Computing with Limited Resources

February, 2025

Graduate School of Science and Technology, Keio University

Kentaro Tamura

A Thesis for the Degree of Ph. D. in Engineering

Approaches for Utilizing Quantum Computing with Limited Resources

February, 2025

Graduate School of Science and Technology, Keio University

Kentaro Tamura

Abstract

Quantum computers are expected to solve certain problems more efficiently than conventional computers by leveraging the laws of quantum mechanics. With rapid advancements in quantum computer development, these devices have become widely accessible through cloud platforms. However, current quantum computers are constrained by limited resources. They only have a few hundred noisy quantum bits (qubits) available, which restricts the range of algorithms that can be run on these devices. Those working with current quantum computers are faced with the problem of utilizing a restricted number of noisy qubits with limited access to hardware parameters. This thesis aims to solve the following problems: the shortage of a simple method for benchmarking qubits of quantum computing devices, and the limited understanding of the noise robustness of optimization algorithms using Quantum Random Access Code (QRAC). Solving these problems provides tools and insight to facilitate the utilization of devices with limited resources.

The first challenge is the need to select stable qubits from quantum computers. Especially with cloud-based quantum computers that can only be accessed remotely, the only information that is guaranteed access is the input and output. To address this challenge, we propose a method for evaluating the stability of qubits of a quantum computing device based only on the input and output. We focus on the task of quantum random number generation, an inherently quantum mechanical task for quantum computers that leverages the Born rule. We find that existing statistical tests for random number generators (RNGs) cannot be directly applied to assess the stability of the qubits of current quantum computing devices because their qubits do not satisfy the unbiased assumption. To address this issue, the proposed method uses a polyline approximation algorithm on the cumulative distribution of the output, allowing for a pure evaluation of stability. We demonstrate through experiments on a real device that the method can detect variations in stability of qubits over time.

The second challenge is the need for noise robust algorithms. Real devices are affected by noise, whose intensity varies over time. To consistently obtain useful results, it is vital to choose algorithms that are robust to noise. In this study, we focus on the noise robustness of a particular combinatorial optimization algorithm referred to as Quantum Random Access Optimization (QRAO). Combinatorial optimization is a task with a wide range of applications, and it is one of the promising applications for quantum computers. QRAO is expected to efficiently solve Quantum Unconstrained Binary Optimization (QUBO) problems with fewer qubits using Quantum Random Access Code (QRAC), but its noise robustness was not sufficiently understood. In particular, the measurement process called quantum state rounding, which extracts classical solutions from quantum states, had unclear effects on the quality of the solutions under noise. In this study, we use the (3,1)-QRAC Hamiltonian, which encodes up to three classical bits per qubit, and numerically demonstrate its higher noise robustness compared to the Ising Hamiltonian, with the MaxCut problem as an example. We also analyze the mechanisms behind the noise robustness and derive the number of measurements required to accurately decode classical bits with a given precision.

The findings of this research provide a method for easily evaluating the stability of noisy qubits and offer insight into the noise-robustness of combinatorial optimization using QRAC. These contributions are expected to help address key challenges for utilizing quantum computers with limited resources.

Contents

List of Figures

List of Tables

Acknowledgements

This thesis is the culmination of the support and encouragement of many individuals and institutions. I am forever grateful to my supervisor, Professor Naoki Yamamoto, for his tremendous patience, support, and honesty. He has provided me with the kind of freedom that is every researcher's dream. I have observed his responsibilities growing with each passing day, and I sincerely hope he can find moments of rest. I also thank the members of my dissertation committee, Professor Hiroshi Watanabe, Associate Professor Shu Tanaka, and Professor Masahiro Takeoka, for their constructive feedback and insightful suggestions. Their feedback not only refined my thesis, but also made me realize new perspectives that I had not considered before.

Meeting Professor Shikano was a defining event in my life. He was a person of true integrity in his approach to research. He generously took the time to engage in discussions with me even when I was still an undergraduate student. I now believe that being given time is the greatest gift of all. Collaborating with him remains one of the greatest honors of my research endeavor. I could not have wished for greater fortune in my research journey.

When I entered the doctoral program, I went through a difficult period for some time. During that time, the encouragement and advice I received from Dr. Yohichi Suzuki proved to be an incredible stroke of luck. His patience in research and emphasis on methodically following each step in the process had a profound impact on me. Had I not met him, my doctoral journey would not have been as meaningful as it turned out to be.

It was Dr. Rudy Raymond who provided me with the initial insight into the theme I would later pursue in my research. He supported me in every aspect, from coding to research discussions. One of the things I learned from him was the importance of getting the work done. He always provided constructive advice, for which I am deeply grateful.

Eventually, I came across yet another savior. Dr. Yuki Sato was not only a researcher worthy of great respect but also a person to be admired. His kindness was an expression of compassion that went beyond personal interests. I aspire to become a person like him: kind, caring, and wise.

I would like to thank Professor Ken Umeno, Associate Professor Atsushi Iwasaki, and Dr. Hidetoshi Okutomi for their valuable advice and support, despite my affiliation with a different laboratory. This experience had made me feel, for the first time, truly connected to a research community.

I am also grateful to the members at the Keio quantum computing center, particularly Associate Professor Hiroshi Watanabe, Dr. Ruho Kondo, Dr. Michihiko Sugawara, Dr. Takashi Abe, Dr. Tomoki Tanaka, Dr. Shunpei Uno, Mr. Kohei Oshio, Dr. Kohei Nakaji, Dr. Hiroyuki Tezuka, Associate Professor Takahiko Satoh, Dr. Takashi Imamichi, Dr. Toshinari Itoko, Ms. Naoko Oue, and Mr. Shinichi Niimi. Their advice extended beyond the scope of my research and into broader questions of life.

The presence of my colleagues was also invaluable during my research pursuit. I am indebted to Mr. Hiroyuki Harada and Mr. Taihei Takahashi for the feedback they provided on my dissertation and for the suggestions to improve my presentation. I was also honored to receive advice from Mr. Kaito Wada, Dr. Hiroshi Yano, and Mr. Jumpei Kato. I am truly grateful to Dr. Ryuki Suzuki for reaching out to me and illuminating my doctoral journey.

I am deeply grateful to Ms. Chiaki Miyamoto and Dr. Toshiaki Sakai, for their encouragement and guidance both on and off the court. Whenever I play tennis, I find myself thinking about life; whenever I'm not, I find myself thinking about tennis. Tennis is a microcosm of life.

I would like to express my gratitude to the JST Doctoral Program Student Support Project and the Keio University Fujiwara Scholarship for their generous financial support, which allowed me to focus on my research.

I would like to convey my utmost thanks to my family. Everything I have done so far stands firmly on the foundation they have provided. I have always had a place to return to, and now I truly understand how invaluable that is.

Chapter 1

Introduction

The idea of leveraging the laws of quantum mechanics to perform computations was proposed more than 40 years ago by pioneers such as Paul Benioff [7], Yuri Manin [8], and Richard Feynman [9]. With advances in theory and engineering, what was once an idea has now become a reality. Today, real quantum computers can be accessed through the cloud. Quantum computers are computational devices that harness the laws of quantum mechanics to perform computations [10]. Unlike classical computers that perform computations via classical bits that take a value of either 0 or 1 at all times, quantum computers perform computations using qubits, which take a value of either 0 or 1 only after measurement. Prior to measurement, qubits exist in quantum states, allowing inherently quantum phenomena such as superposition and entanglement. These phenomena can be leveraged to construct quantum algorithms that are more efficient than their classical counterparts. Some promising quantum algorithms have been devised, including Shor's factorization algorithm [11] and Grover's database search algorithm [12]. The former achieves an exponential speedup compared to the best classical algorithms, and the latter achieves a quadratic speedup compared to any classical algorithm.

Despite remarkable progress in the field of quantum computing, some of the most promising quantum algorithms cannot be implemented on current quantum computers. This is because current quantum computers are constrained by limited resources, specifically by having only a few hundred noisy qubits available. These devices are referred to as Noisy Intermediate-Scale Quantum (NISQ) [13] devices. This constraint hinders the implementation of some of the groundbreaking algorithms. For instance, Shor's factorization algorithm has been estimated to require 20 million qubits to factor 2024 bit RSA integers in 8 hours [14], and Grover's database search algorithm fails to achieve a quadratic speedup with noisy qubits [15]. The limitation on the number of qubits and the influence of noise give rise to two significant challenges for the utilization of current devices: first, the effect of noise limits the number of operations that can be performed before the accumulated impact of noise compromises the computational results, and second, the limited number of qubits narrows the size of problems that can be solved.

Given the circumstances of current quantum computers, research is underway to utilize the limited number of noisy qubits available. One promising approach is to perform computations with the joint effort of classical computers and quantum computers [16]. Quantum algorithms that achieve this are referred to as hybrid quantum-classical algorithms [17]. A prime example of a hybrid quantum-classical algorithm is the Variational Quantum Eigensolver (VQE) [18–20]. VQE is a quantum algorithm to obtain the ground state energy of a Hamiltonian. It was proposed as an alternative to algorithms that are difficult to implement on NISQ devices, such as Quantum Phase Estimation (QPE) [21–23]. A Hamiltonian is an operator that represents the total energy of a system, and its ground state energy provides valuable information in quantum chemistry [24, 25] and condensed matter physics [26]. VQE can also be applied to combinatorial optimization [27, 28], a task with numerous applications in industry [29, 30] and finance [31, 32].

Another approach to utilizing current devices is to characterize the performance of the devices and

their qubits through benchmarks [33]. With the advent of cloud-based platforms, we are now confronted with the challenge of selecting which devices and qubits to perform computations on. Benchmarks serve as a valuable tool in this sense as they allow us to make informed decisions about which devices and qubits are suitable for their needs. Various benchmarks have been proposed [34–40], some of which focus on the characterization of noise, while others highlight the ability to perform specific tasks. The main challenges in developing benchmarks for current quantum computing devices have been identified, including the uncertainty surrounding their most useful applications, the diversity of physical implementations, and the difficulty of characterizing quantum noise [37]. These challenges are even more pronounced with cloud-based quantum computers, where access to hardware parameters is limited.

The work aims to address two key challenges arising from the resource limitations of current quantum computing devices: the difficulty of selecting stable qubits for computation and the limited understanding of how specific combinatorial algorithms perform under noise conditions. First, it introduces a benchmarking scheme based on quantum random number generation, offering a simple and portable method of analyzing the stability of qubits. Second, we provide an analysis of the noise-robustness of a quantum algorithm for approximate combinatorial optimization, providing a deeper understanding of how the reduction of qubits leads to an increase in noise-robustness.

First, we outline our approach to benchmarking qubits. Qubits are a fundamental building block of quantum computers, and benchmarking of their performance is critical [37, 41]. We focus on quantum random number generation, a simple but inherently quantum mechanical task that can be executed on a quantum computer. Quantum random number generation is the production of random bits arising from the measurement of a quantum state [42]. Due to the Born rule, which states that the measurement results of a quantum state are inherently probabilistic, the resulting bits can be considered truly random [43]. To implement quantum random number generation on a quantum computer, the simplest method is to apply the Hadamard gate to all qubits, creating a balanced superposition state. In principle, the output sequence of each qubit should be truly random. However, we found that the output random number sequence of the qubits of real quantum computing devices fails to pass conventional tests for random number generators [2, 44]. Further investigation revealed that the output is not only biased, but also non-identically distributed [2]. The fact that the output random number sequence is non-identically distributed indicates that the device is unstable. We leveraged this fact to analyze the stability of qubits via quantum random number generation. Stability is a crucial characteristic of a quantum computing device to produce reproducible results [39, 45–53]. Because conventional tests for RNGs cannot isolate and test the non-identically distributed characteristic, we propose an alternative way of checking this assumption by applying a polyline algorithm on the cumulative distribution of the output of each qubit.

Second, we summarize the result of analyzing the noise-robustness an algorithm referred to as QRAO [27]. The algorithm utilizes an encoding method called QRAC [54–57] to embed multiple classical variables per qubit. The key idea of QRAC is that by allowing the probabilistic retrieval of classical bits, the Holevo bound [58], which states that at least n qubits are required to encode and decode n classical bits, can be alleviated. QRAC was first originally proposed in the context of quantum finite automata [55, 56, 59] and quantum communication complexity [60–62]. In the context of combinatorial optimization, QRAC can be leveraged to reduce the number of qubits required to solve a problem by at best 1/3 compared to other methods such as VQE or the Quantum Approximate Optimization Algorithm (QAOA) [63, 64]. Although the algorithm cuts back on the use of qubits, a valuable resource for quantum computers, its effect on the quality of the classical solution was unclear. In particular, the effect of reducing the use of qubits on the quality of the classical solution was a significant issue that warranted further investigation. We addressed this issue by simulating the algorithm under fake noise, which is a noise model that incorporates the single-qubit depolarizing error, single-qubit thermal relaxation error, two-qubit depolarizing error, and single-qubit readout error, whose parameters are tuned based on real system snapshots [4]. We found that the use of QRAC leads to a more robust quality of the solution in terms of the approximation ratio [5]. Additionally, we investigated the reason behind the robustness of

QRAO by assuming depolarizing noise, and found that there exists an intersection point in the problem size where the lower bound on the expected approximation of QRAO exceeds that of the Ising type encoding. We also found that QRAO is also efficient with regard to the number of measurements required to obtain the classical solution with a given precision.

The goal of this thesis is to provide tools and insight to maximize the utility of the limited number of noisy qubits that current devices offer. The structure of this thesis is as follows.

- **Chapter 2** outlines the foundations of quantum computing, covering topics such as qubits, quantum states, quantum gates, quantum circuits, and descriptions of noise in quantum computing.
- **Chapter 3** describes our method for benchmarking the stability of qubits via quantum random number generation. To lay the groundwork for the method, we review relevant methods for benchmarking qubits and quantum computing devices. We also introduce the development of various methods of random number generation and statistical tests for generators.
- **Chapter 4** discusses the noise-robustness of QRAO with simulation results under noise and theoretical analysis. Related topics such as QRAC, VQE, and QAOA are also covered.
- **Chapter 5** provides a conclusion and outlook of the thesis.

In the appendix, we offer an introduction to probability distributions for testing random number generators, additional results of real devices, and the derivation of the results in Chapter 4.

Chapter 2

Preliminaries

A quantum computer is a computational device that operates by utilizing the principles of quantum mechanics [10]. In this chapter, we establish the foundation for understanding the key components of quantum computing, including qubits, quantum states, quantum gates, quantum circuits, and models of quantum noise.

2.1 Fundamentals of Quantum Computing

The fundamental unit of computation in a classical computer is the classical bit, which can assume a value of either 0 or 1. In a quantum computer, the fundamental unit is a qubit. A quantum computer performs computations by applying quantum gates to its qubits. In this section, we explain the concept of qubits, quantum gates, and quantum circuits to provide an introduction to quantum computers. We assume a basic understanding of quantum information and linear algebra here.

2.1.1 Quantum Bits (Qubits) and Quantum States

Let us denote the quantum state corresponding to the classical bits 0 and 1 by $|0\rangle$ and $|1\rangle$, respectively. $|0\rangle$ and $|1\rangle$ are vectors in a two-dimensional complex vector space that form an orthonormal basis. They are often called computational basis states. $|0\rangle$ and $|1\rangle$ can be written as

$$|0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \qquad |1\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix}.$$

A qubit is different from a classical bit in that a qubit can assume a quantum state that is described as a linear combination (superposition) of the states $|0\rangle$ and $|1\rangle$. The state of a qubit can be written as

$$|\psi\rangle = c_1 |0\rangle + c_2 |1\rangle = \begin{bmatrix} c_1 \\ c_2 \end{bmatrix}, \tag{2.1}$$

where c_1 and c_2 are complex numbers that fulfill the relation $|c_1|^2 + |c_2|^2 = 1$. The notation $|\cdot\rangle$ is called Dirac notation, which allows for easily expressing operations such as Hermitian conjugation

$$\langle\psi| = (|\psi\rangle)^\dagger = [c_1^*, c_2^*] \tag{2.2}$$

and inner product

$$|\psi\rangle \langle\phi| = [c_1^*, c_2^*] \begin{bmatrix} d_1 \\ d_2 \end{bmatrix} = c_1^* d_1 + c_2^* d_2. \tag{2.3}$$

Upon measuring the state $|\psi\rangle$, the state $|0\rangle$ or $|1\rangle$ is observed. This corresponds to one of the eigenstates of the system. This measurement is called a projective measurement. According to the Born rule, an individual quantum measurement is unpredictable in principle [65]. The probability that the state $|0\rangle$ is observed is given by $|c_1|^2$, and the probability that the state $|1\rangle$ is observed is given by $|c_2|^2$. This unpredictability fundamentally distinguishes quantum computers from classical computers [44]. A qubit can be geometrically represented by the Bloch sphere. The Bloch sphere representation of a qubit is obtained by describing its state as

$$|\psi\rangle = \cos\left(\frac{\theta}{2}\right)|0\rangle + e^{i\varphi}\sin\left(\frac{\theta}{2}\right)|1\rangle \tag{2.4}$$

and by interpreting (θ, φ) as polar coordinates. Note that $0 \leq \theta \leq \pi$ and $0 \leq \varphi \leq 2\pi$. The state can now be plotted on the Bloch sphere seen in Fig. 2.1.

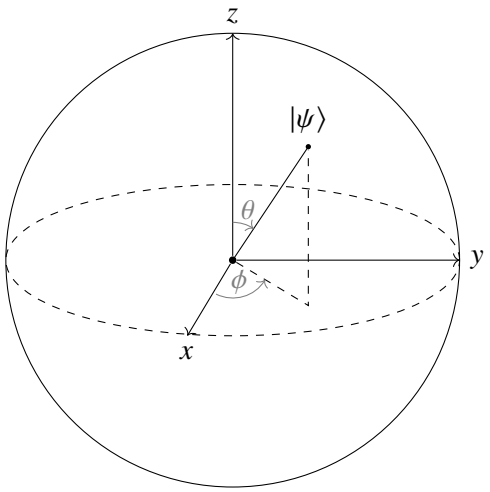


Fig. 2.1. Bloch sphere representation of the quantum state of a qubit.

The state of multiple qubits is described by the tensor product ($\otimes$) of the states involved. For example, the two-qubit state where the first qubit is in the state $|0\rangle$ and the second qubit is in the state $|1\rangle$ is denoted $|0\rangle \otimes |1\rangle$, which can be abbreviated as $|0\rangle|1\rangle$ or $|01\rangle$. In matrix form, the tensor product is written as

$$|\psi\rangle \otimes |\phi\rangle = \begin{bmatrix} c_{11} \\ c_{12} \end{bmatrix} \otimes \begin{bmatrix} c_{21} \\ c_{22} \end{bmatrix} = \begin{bmatrix} c_{11}c_{21} \\ c_{11}c_{22} \\ c_{12}c_{21} \\ c_{12}c_{22} \end{bmatrix}. \tag{2.5}$$

An n-qubit state is therefore described as a 2^n-dimensional vector.

2.1.2 Quantum Gates

A quantum computer performs computations by applying quantum gates on its qubits [10, 66]. This corresponds to a rotation of the state vector on the Bloch sphere. A single-qubit gate is a quantum gate that operates on a single qubit. It is represented by a 2×2 complex matrix. The state of a qubit after the operation of a gate U is

$$U|\psi\rangle, \tag{2.6}$$

where U is a unitary matrix that fulfills the relation

$$U^\dagger U = UU^\dagger = I. \tag{2.7}$$

Here, I is the identity matrix. The matrix representing the gate must be a unitary matrix for the state after gate operation to be normalized, that is,

$$(U|\psi\rangle)^\dagger (U|\psi\rangle) = \langle\psi| U^\dagger U |\psi\rangle = 1. \tag{2.8}$$

The Pauli-X gate is a common single-qubit gate often denoted X. The X gate is a quantum computer equivalent of the NOT operation in classical computers. The result of applying the X gate is the following:

$$X|0\rangle = |1\rangle, \tag{2.9}$$
$$X|1\rangle = |0\rangle. \tag{2.10}$$

This corresponds to a rotation around the X-axis by π on the Bloch sphere. The Pauli-Z gate is another common gate that flips the phase of the qubit:

$$Z|0\rangle = |0\rangle,$$
$$Z|1\rangle = -|1\rangle$$

This corresponds to a rotation around the Z-axis by π. It is useful to know that the X gate and Z gate can also be written as

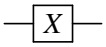
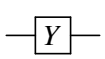
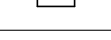
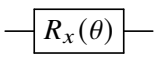
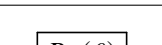
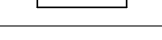
$$X = |0\rangle\langle 1| + |1\rangle\langle 0|, \tag{2.11}$$
$$Z = |0\rangle\langle 0| - |1\rangle\langle 1|. \tag{2.12}$$

The Hadamard gate creates superposition. It is a $\pi/2$ rotation around the Y-axis followed by a π rotation around the Z-axis on the Bloch sphere. The operation of the gate is as follows:

$$H|0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) = |+\rangle, \tag{2.13}$$
$$H|1\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) = |-\rangle. \tag{2.14}$$

Measurement of state $|+\rangle$ in the computational basis $\{|0\rangle, |1\rangle\}$ gives the measurement result $|0\rangle$ and $|1\rangle$ with the same probability. By repeatedly preparing and measuring this state, true random numbers can be generated based on the inherently probabilistic nature of quantum mechanics [67]. A list of quantum gates is provided in Table 2.1.

Table 2.1. Common quantum gates and their circuit representation, matrix representation, and property.

Name	**Circuit**	**Matrix**	**Property**
X Gate (Pauli-X)		$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$	Bit-flip (rotates state by π around the X-axis).
Y Gate (Pauli-Y)		$\begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}$	Rotates state by π around the Y-axis.
Z Gate (Pauli-Z)		$\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$	Phase-flip (rotates state by π around the Z-axis).
Hadamard Gate		$\frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$	Creates superposition (rotates state by $\pi/2$ around the Y-axis and π around the Z-axis).
Phase Gate		$\begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix}$	Adds a phase of $\frac{\pi}{2}$ (quarter-turn in the phase).
T Gate		$\begin{bmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{bmatrix}$	Adds a phase of $\frac{\pi}{4}$ (eighth-turn in the phase).
$R_x(\theta)$		$\begin{bmatrix} \cos\frac{\theta}{2} & -i\sin\frac{\theta}{2} \\ -i\sin\frac{\theta}{2} & \cos\frac{\theta}{2} \end{bmatrix}$	Rotates state by θ around the X-axis.
$R_y(\theta)$		$\begin{bmatrix} \cos\frac{\theta}{2} & -\sin\frac{\theta}{2} \\ \sin\frac{\theta}{2} & \cos\frac{\theta}{2} \end{bmatrix}$	Rotates state by θ around the Y-axis.
$R_z(\theta)$		$\begin{bmatrix} e^{-i\theta/2} & 0 \\ 0 & e^{i\theta/2} \end{bmatrix}$	Rotates state by θ around the Z-axis.
CNOT Gate (Controlled-NOT)		$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$	Flips target qubit if control qubit is $\|1\rangle$.

Two-qubit gates are quantum gate operations that operate on two qubits. The most fundamental two-qubit gate is the controlled-NOT gate. The controlled-NOT gate is represented by the matrix

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}, \tag{2.15}$$

whose application on the states two-qubit states $|00\rangle, |01\rangle, |10\rangle, |11\rangle$ result in the following:

$$\text{CNOT}(|00\rangle) = |00\rangle, \tag{2.16}$$
$$\text{CNOT}(|01\rangle) = |01\rangle, \tag{2.17}$$
$$\text{CNOT}(|10\rangle) = |11\rangle, \tag{2.18}$$
$$\text{CNOT}(|11\rangle) = |10\rangle. \tag{2.19}$$

The first qubit is called the control qubit and the second qubit the target qubit. When the control qubit is in the state $|1\rangle$, the X gate is applied to the target qubit, and when the control qubit is in the state $|0\rangle$,

the state of the target qubit is left unchanged. The gate can be used to create what is called the Bell state. Bell states are the following:

$$|\Phi^+\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle), \tag{2.20}$$

$$|\Phi^-\rangle = \frac{1}{\sqrt{2}}(|00\rangle - |11\rangle), \tag{2.21}$$

$$|\Psi^+\rangle = \frac{1}{\sqrt{2}}(|01\rangle + |10\rangle), \tag{2.22}$$

$$|\Psi^-\rangle = \frac{1}{\sqrt{2}}(|01\rangle - |10\rangle). \tag{2.23}$$

The first Bell state $|\Phi^+\rangle$ can be created by applying the Hadamard gate on the first qubit of the state $|00\rangle$, followed by the controlled-NOT gate with the first qubit as the control qubit and the second qubit as the target qubit. The Bell states represent the maximal quantum entanglement between qubits. For example, consider the Bell state $|\Phi^+\rangle$. Measurement of the first qubit yields $|0\rangle$ or $|1\rangle$. If the first qubit is found to be in state $|0\rangle$, the second qubit will also be in the state $|0\rangle$. Similarly, if the first qubit is found to be in the state $|1\rangle$, the second qubit will be in the state $|1\rangle$. This inherently quantum phenomenon occurs regardless of the physical distance between the two qubits forming the Bell state and is the foundation of technologies such as quantum teleportation.

2.1.3 Quantum Circuits

A quantum circuit is a sequence of instructions for a quantum computer, specifying which quantum gate operations to apply, in what order, and to which qubits. A quantum computer performs computations based on the given quantum circuit. An example of a quantum circuit is provided in Fig. 2.2.

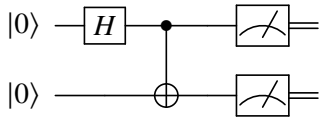


Fig. 2.2. A quantum circuit for creating and measuring a Bell state.

A quantum computer executes a quantum circuit from the left to right. Given the quantum circuit in Fig. 2.2, the quantum computer first prepares the initial state $|00\rangle$. The first qubit is then subjected to the Hadamard gate, upon which the entire state becomes $(|00\rangle + |10\rangle)/\sqrt{2}$. The next gate in the circuit is the controlled-NOT gate, whose control qubit is the first qubit and whose target qubit is the second qubit. The state after the controlled-NOT gate becomes $(|00\rangle + |11\rangle)/\sqrt{2}$, which is a Bell state. Finally, each qubit is measured. The outcome is the classical bits, either 00 or 11. Typically, a quantum circuit is executed repeatedly on a quantum computer to obtain measurement results that are used to infer the probabilities of different quantum states. The number of measurements is often referred to as shots.

2.2 Description of Noise in Quantum Computing

The quantum states and quantum gates introduced so far assume ideal conditions, where quantum systems and operations are free from noise. However, real quantum devices are prone to noise and require a density matrix formulation for an accurate description. In this chapter, the density matrix formulation is introduced as a tool for describing quantum states in the presence of noise. Subsequently, quantum noise models that utilize the density matrix formulation are introduced to account for the imperfections present in real quantum devices.

2.2.1 Density Matrix Formulation

The density matrix formulation is a convenient way of describing a quantum state under the influence of noise. Quantum states that can be described as a linear combination and tensor product of basis states, such as $|0\rangle$ or $|1\rangle$ are called pure states. For example, the state $\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ is a pure state. The measurement outcome of this pure state in the computational basis yields $|0\rangle$ or $|1\rangle$ with equal probability due to the Born rule. On the other hand, let us consider a set of qubits consisting of equal numbers of qubits in the $|0\rangle$ and $|1\rangle$ states. The measurement outcome of a qubit taken from this set uniformly at random also yields the result of $|0\rangle$ or $|1\rangle$ statistically with equal probability. States like the latter, that cannot be described as a pure state and are represented by a probabilistic mixture of pure states are called mixed states.

A mixed state is represented by the density matrix ρ. The mixed state where the pure states $|0\rangle$ and $|1\rangle$ are mixed with equal probability is described as $\rho = 0.5\,|0\rangle\langle 0| + 0.5\,|1\rangle\langle 1|$. Generally, the mixed state where the pure states $|\psi_k\rangle$ are mixed with the probabilities p_k is denoted

$$\rho = \sum_k p_k\,|\psi_k\rangle\langle\psi_k|\,. \tag{2.24}$$

The density matrix fulfills the following conditions:

1. Trace condition: $\mathrm{Tr}(\rho) = 1$,
2. Positivity condition: $\langle\psi|\,\rho\,|\psi\rangle \geq 0$,
3. Hermiticity: $\rho = \rho^\dagger$.

The probability of obtaining the measurement result corresponding to the state $|\psi_k\rangle$ by measuring the density matrix ρ is given by $\mathrm{Tr}(\rho\,|\psi_k\rangle\langle\psi_k|)$.

Using the density matrix formalism, the previously introduced projective measurement can be generalized into what is called the positive operator valued measure (POVM). A POVM is a set of positive semi-definite operators $\{E_i\}$ that fulfill:

$$\sum_i E_i = I, \tag{2.25}$$

where E_i represents the possible outcome of measurement. The probability of obtaining the i-th measurement result from the state ρ is given by

$$\Pr[i] = \mathrm{Tr}(\rho E_i). \tag{2.26}$$

A POVM is a general description of measurement that includes projective measurements and is essential for describing measurements in a noisy environment.

2.2.2 Depolarizing Noise

A single-qubit quantum state ρ is depolarized when it is reduced to the completely mixed state $I/2$. Geometrically, this corresponds to the state at the center of the Bloch sphere. Depolarizing noise causes the state to depolarize with probability $(1 - p)$ and leaves the state unaffected with probability p. This causes a shrinking effect on the Bloch sphere towards the center as seen in Fig. 2.3(b).

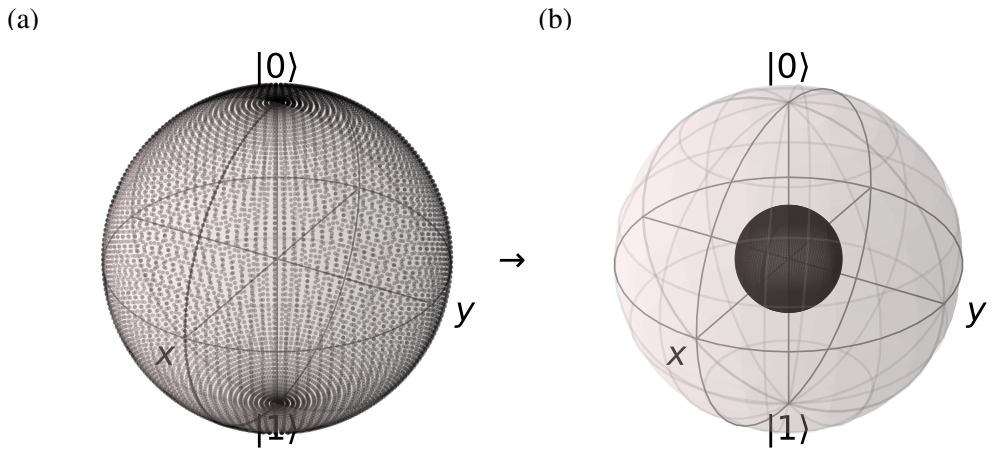


Fig. 2.3. (a) The Bloch sphere before depolarizing noise, where the states are distributed over the full surface. (b) The Bloch sphere after depolarizing noise, showing the shrinking of the points towards the center, representing a mixed state.

Let us denote the n-qubit density matrix ρ after applying the depolarization noise with error parameter $1-p$ as $\mathcal{D}(\rho)$. The state can then be written as

$$\mathcal{D}(\rho) = p\rho + (1-p)\frac{I}{2^n}. \tag{2.27}$$

The state under depolarizing noise can also be represented by the Pauli matrices as

$$\mathcal{D}(\rho) = p\rho + \frac{1-p}{3}(X\rho X + Y\rho Y + Z\rho Z), \tag{2.28}$$

which gives the intuition that the noise is equivalent to applying the gates X, Y, Z on the state with probability $(1-p)/3$ each, and leaving the state alone with probability p. When depolarizing noise is applied a second time on the state ρ, the resulting state is

$$\begin{aligned}
\mathcal{D}^2(\rho) &= \mathcal{D}(\mathcal{D}(\rho)) \\
&= \mathcal{D}\left(p\rho + (1-p)\frac{I}{2^n}\right) \\
&= p\left(p\rho + (1-p)\frac{I}{2^n}\right) + (1-p)\frac{I}{2^n} \\
&= p^2\rho + p(1-p)\frac{I}{2^n} + (1-p)\frac{I}{2^n} \\
&= p^2\rho + \left[p(1-p) + (1-p)\right]\frac{I}{2^n} \\
&= p^2\rho + \left(1-p^2\right)\frac{I}{2^n}
\end{aligned} \tag{2.29}$$

The result of applying depolarizing noise N times to the state ρ is therefore,

$$\mathcal{D}^N(\rho) = p^N\rho + \left(1-p^N\right)\frac{I}{2^n}. \tag{2.30}$$

2.2.3 Amplitude Damping Noise

Amplitude damping noise causes a system that is initially in an excited state to spontaneously dissipate energy, transitioning to its ground state [68]. The state ρ under amplitude damping noise becomes the

state ρ' after some time t. The state ρ' is given by

$$\rho' = E_0 \rho' E_0^\dagger + E_1 \rho E_1^\dagger \tag{2.31}$$

$$= \begin{pmatrix} 1 & 0 \\ 0 & \sqrt{1-\lambda} \end{pmatrix} \rho \begin{pmatrix} 1 & 0 \\ 0 & \sqrt{1-\lambda} \end{pmatrix}^\dagger + \begin{pmatrix} 0 & \sqrt{\lambda} \\ 0 & 0 \end{pmatrix} \rho \begin{pmatrix} 0 & \sqrt{\lambda} \\ 0 & 0 \end{pmatrix}^\dagger, \tag{2.32}$$

where $\lambda = 1 - e^{-t/T_1}$. e^{-t/T_1} represents the probability that a state after some time t under thermal relaxation noise is found in the state $|1\rangle$. The expression $\rho' = E_0 \rho E_0^\dagger + E_1 \rho E_1^\dagger$ is called the Kraus representation [69]. The Kraus matrices E_0 and E_1 describe the transformation of the density matrix under noise. Any physically valid quantum operation can be described by the Kraus representation as $\rho' = \sum_i E_i \rho E_i^\dagger$. The Kraus matrices have the following properties:

1. (Complete Positivity) The density matrix resulting from the transformation $\rho' = \sum_i E_i \rho E_i^\dagger$ always has non-negative eigenvalues.

2. (Trace-Preserving) The sum of operators $\sum_i E_i^\dagger E_i = I$.

These properties ensure that the Kraus representation is suitable for describing the evolution of quantum systems.

Geometrically, the amplitude damping noise corresponds to the shrinking of the Bloch sphere towards the ground state $|0\rangle$ as seen in Fig. 2.4(b).

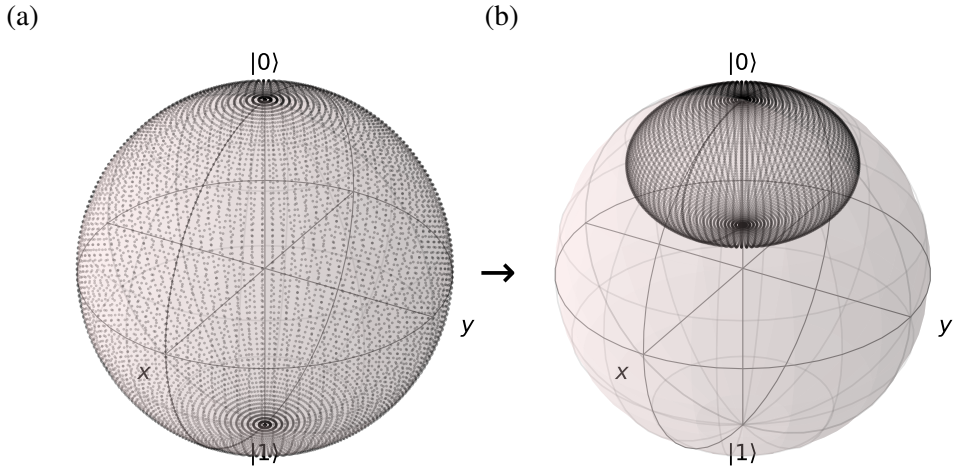


Fig. 2.4. (a) The Bloch sphere before amplitude damping noise, where the states are distributed across the full surface. (b) The Bloch sphere after amplitude damping noise, showing the contraction of the points towards the state $|0\rangle$.

The T_1 relaxation time describes the rate at which the excited state dissipates energy and becomes the ground state.

2.2.4 Phase Damping Noise

The phase damping noise causes the state to lose its quantum information while sustaining its energy. The state, after some time t under the phase damping noise, is given by

$$\rho' = M_0 \rho M_0^\dagger + M_1 \rho M_1^\dagger \tag{2.33}$$

$$= \begin{pmatrix} 1 & 0 \\ 0 & \sqrt{1-\gamma} \end{pmatrix} \rho \begin{pmatrix} 1 & 0 \\ 0 & \sqrt{1-\gamma} \end{pmatrix}^\dagger + \begin{pmatrix} 0 & 0 \\ 0 & \sqrt{\gamma} \end{pmatrix} \rho \begin{pmatrix} 0 & 0 \\ 0 & \sqrt{\gamma} \end{pmatrix}^\dagger \tag{2.34}$$

where $\gamma = 1 - e^{-t/T_2}$. Geometrically, phase damping noise corresponds to the shrinking of the Bloch sphere towards the z-axis, as seen in Fig. 2.5(b).

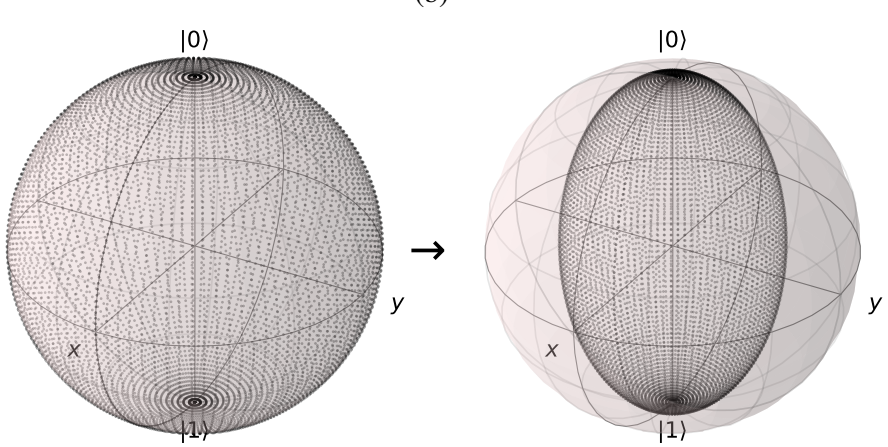


Fig. 2.5. (a) The Bloch sphere before phase damping noise, where the states are distributed across the full surface. (b) The Bloch sphere after phase damping noise, showing the shrinking of the points towards the z-axis.

2.2.5 Readout Error

Readout error is the misinterpretation of the classical bits obtained by the measurement of a quantum state. In the single-qubit case, the misinterpretation is either $0 \rightarrow 1$ or $1 \rightarrow 0$. The probabilities associated with each misinterpretation can be written as $\Pr[0|1]$ and $\Pr[1|0]$, respectively. By considering the probabilities of correctly and incorrectly interpreting the output, we obtain the following matrix representing the single-qubit readout error:

$$A = \begin{bmatrix} \Pr[0|0] & \Pr[0|1] \\ \Pr[1|0] & \Pr[1|1] \end{bmatrix}. \tag{2.35}$$

Similarly, the two-qubit readout error can be represented by the following matrix:

$$A = \begin{bmatrix} \Pr[00|00] & \Pr[00|01] & \Pr[00|10] & \Pr[00|11] \\ \Pr[01|00] & \Pr[01|01] & \Pr[01|10] & \Pr[01|11] \\ \Pr[10|00] & \Pr[10|01] & \Pr[10|10] & \Pr[10|11] \\ \Pr[11|00] & \Pr[11|01] & \Pr[11|10] & \Pr[11|11] \end{bmatrix}. \tag{2.36}$$

The main sources of readout error are the following [70].

1. Decoherence of qubits during measurement.
2. Overlapping support in probability distributions of measurement results corresponding to the states $|0\rangle$ and $|1\rangle$.

One common method to mitigate readout noise is through matrix inversion [70]. In this approach, the inverse of the readout error probability matrix is applied as follows:

$$y = A^{-1}x, \tag{2.37}$$

where x represents the noisy output due to readout error, and y denotes the corrected output.

Chapter 3

Stability Analysis of Quantum Computers via Quantum Random Number Generation

In the present chapter, we propose a method for benchmarking the stability of quantum computing devices via quantum random number generation. We also present the results of its application to a real device. To do so, we review methods for benchmarking quantum computing devices and qubits. We then explain the crucial concepts in random number generation, such as the definition of randomness, randomness extractors, and classification of random number generators. We then elaborate on the limitations of applying conventional tests for random number generators to the output of quantum computing devices. Finally, we present our method and results.

3.1 Benchmarking Quantum Computing Devices

Current quantum computing devices are NISQ devices. NISQ devices are quantum computing devices with 50 to a few hundred noisy qubits [13]. While these devices have limited computational capabilities compared to fault-tolerant quantum computers, which are still in development and expected to perform error-free quantum operations, research is being conducted to harness the computational capabilities of NISQ devices as much as possible [71]. When performing computations on such a quantum computing device, selecting which qubits and devices to use is a critical issue that affects the accuracy of the computational results. As cloud-based NISQ quantum computers become increasingly accessible, the demand to remotely compare, select, and utilize qubits and devices is growing.

Benchmarking methods for quantum computing devices evaluate the performance and characteristics of devices [72], enabling informed decisions regarding qubit selection and device usage. There are mainly four reasons why benchmarking quantum computing devices is difficult [37]:

1. There is a wide range of implementations of quantum hardware;
2. Quantum hardware is still under development and is not yet capable of performing practical computations;
3. The most useful applications of quantum computing remain uncertain;
4. Characterizing quantum noise is challenging, as it is not yet fully comprehended.

To address these challenges comprehensively, a variety of benchmarking methods are necessary.

Benchmarking methods for quantum computing devices can be categorized along two axes: complexity and level of abstraction [72]. The complexity of a benchmarking method relates to the level of detail of the object under evaluation, ranging from single logic gates to entire computing systems. The

level of abstraction, on the other hand, indicates how closely the measure aligns with practical applications. Benchmarking methods with low complexity involve isolating the performance of qubits and gates from classical components, like compilers, within an integrated quantum computing system. These include methods such as randomized benchmarking [73–75], or methods that involve solving small problems [76–79]. Benchmarking methods with high complexity focus on evaluating the overall performance of integrated quantum computers, such as quantum volume [72]. In this section, we review some key methods of benchmarking quantum computing devices and their qubits.

3.1.1 Gate Fidelity

The gate fidelity is a measure of how close to ideal a gate operation on a quantum state is. It provides an important benchmark for qubits because it characterizes the accuracy of quantum operations [80]. Let us define the quantum state prior to gate operation as ρ and the quantum state after gate operation as ρ'. The gate fidelity is then defined as

$$F(\rho, \rho') = \mathrm{Tr}\left(\sqrt{\sqrt{\rho}\rho'\sqrt{\rho}}\right)^2. \tag{3.1}$$

When the state ρ is a pure state $|\psi\rangle\langle\psi|$, the gate fidelity can be written as

$$F(|\psi\rangle, \rho') = \sqrt{\langle\psi|\rho'|\psi\rangle}. \tag{3.2}$$

The gate fidelity has several important properties, as listed below.

1. $F(\rho, \rho') = F(\rho', \rho)$;
2. $0 \leq F(\rho, \rho') \leq 1$;
3. $F(\rho, \sigma) \geq 0$ with equality if and only if $\rho\sigma = 0$;
4. $F(\rho, \sigma) \leq 1$ with equality if and only if $\rho = \sigma$.

There are various techniques for estimating the gate fidelity or the average gate fidelity to benchmark qubits, including quantum state tomography or randomized benchmarking [33].

3.1.2 Quantum Process Tomography

Quantum process tomography is the characterization of a quantum process achieved by subjecting a set of known input states $\{\rho_i\}$ to a quantum process and by measuring the set of corresponding output states $\{\rho_i'\}$. The quantum process is reconstructed using the set of input states and output states. As stated earlier, a quantum process can be described as a map

$$\mathcal{E}(\rho) = \sum_i E_i \rho E_i^\dagger = \rho', \tag{3.3}$$

where E_i are Kraus operators. These operators can be expressed as a linear combination of Pauli operators P_i defined as:

$$P_j = P_{j_1} \otimes P_{j_2} \otimes \cdots \otimes P_{j_n}, \tag{3.4}$$

where $P_{j,k} \in \{X, Y, Z, I\}$ for $k \in \{1, 2, \cdots, n\}$ for an n-qubit system. The expansion of E_i is then given by

$$E_i = \sum_{j=1}^{2^n} a_{i,j} P_j, \tag{3.5}$$

where $a_{i,j}$ are complex coefficients. Therefore, a quantum process can be written as

$$\mathcal{E}(\rho) = \sum_{j,k=1} \chi_{j,k} P_j \rho P_k, \tag{3.6}$$

where $\chi_{j,k} = a_{i,j} a^*_{i,k}$. In quantum process tomography, the goal is to estimate the elements of the $\chi-$matrix (often called the process matrix), which completely determines the quantum process. Due to the completeness condition, we have

$$\mathcal{E}(I) = \sum_{j,k=1} \chi_{j,k} P_j P_k = I, \tag{3.7}$$

and thus the process matrix has $4^n(4^n - 1)$ free parameters [81]. In quantum process tomography, the $\chi-$matrix is obtained in the following steps: In quantum process tomography, the χ-matrix is obtained in the following steps:

1. **Prepare input states**: Prepare a set of input states $\rho_1, \rho_2, \rho_3, \rho_4$ that form a complete basis for the Hilbert-Schmidt space.

2. **Apply the quantum process**: Apply the quantum process to each input state, resulting in output states $\rho'_1, \rho'_2, \rho'_3, \rho'_4$.

3. **Perform quantum state tomography**: Measure the output states $\rho'_1, \rho'_2, \rho'_3, \rho'_4$ using quantum state tomography to determine their density matrices.

4. **Obtain elements of χ**: Based on the input-output pairs (ρ_i and ρ'_i), we solve Eq.(3.5) to obtain the elements of the process matrix.

We now provide an example of how this is done. Let us consider the case where the Hadamard gate H is applied to a single qubit. The aim is thus to obtain the process matrix that represents the Hadamard gate. This quantum process is denoted

$$\mathcal{E}(\rho) = H\rho H^\dagger. \tag{3.8}$$

We prepare the four input states

$$\rho_1 = |0\rangle\langle 0|, \tag{3.9}$$
$$\rho_2 = |1\rangle\langle 1|, \tag{3.10}$$
$$\rho_3 = |+\rangle\langle +|, \tag{3.11}$$
$$\rho_4 = |i+\rangle\langle i+|, \tag{3.12}$$

where $|i+\rangle = (|0\rangle + i|1\rangle)/\sqrt{2}$. We then apply the Hadamard gate to each input state. The output states should be:

$$\mathcal{E}(\rho_1) = H|0\rangle\langle 0|H^\dagger = |+\rangle\langle +|, \tag{3.13}$$
$$\mathcal{E}(\rho_2) = H|1\rangle\langle 1|H^\dagger = |-\rangle\langle -|, \tag{3.14}$$
$$\mathcal{E}(\rho_3) = H|+\rangle\langle +|H^\dagger = |0\rangle\langle 0|, \tag{3.15}$$
$$\mathcal{E}(\rho_4) = H|i+\rangle\langle i+|H^\dagger = |i-\rangle\langle i-|, \tag{3.16}$$

where $|-\rangle = (|0\rangle - |1\rangle)/\sqrt{2}$ and $|i-\rangle = (|0\rangle - i|1\rangle)/\sqrt{2}$. Suppose that we prepared ρ_1, applied the quantum gate, and obtained ρ'_1. To identify the state ρ'_1, we measure the state along the basis $\{|0\rangle, |1\rangle\}$,

$\{|+\rangle, |-\rangle\}$, and $\{|i+\rangle, |i-\rangle\}$, respectively. To recover the state ρ_1', we use the fact that a single-qubit density matrix can be represented by

$$\rho_1' = \frac{1}{2}(I + S_1 X + S_2 Y + S_3 Z). \tag{3.17}$$

Here, S_1, S_2, and S_3 are parameters that can be determined from measurements. The values of each parameter are calculated as follows:

- S_1: Apply the Hadamard gate H to the qubit and then measure in the computational basis. The parameter is given by $S_1 = 2\Pr[|0\rangle] - 1$, where $\Pr[|1\rangle]$ represents the probability of observing $|1\rangle$.
- S_2: Apply the inverse phase gate $S^\dagger$ followed by the Hadamard gate H. Then, measure in the computational basis. The parameter is given by $S_2 = 2\Pr[|1\rangle] - 1$.
- S_3: Measure directly in the computational basis. The parameter is given by $S_3 = 2\Pr[|0\rangle] - 1$.

In this example, $\rho_1' = |+\rangle\langle+|$, so, the parameters should be

$$S_1 = 1, \tag{3.18}$$
$$S_2 = 0, \tag{3.19}$$
$$S_3 = 0. \tag{3.20}$$

The state is recovered as

$$\rho_1' = \frac{1}{2}(I + X) = |+\rangle\langle+|. \tag{3.21}$$

Likewise, we obtain the remaining output states

$$\rho_2' = \frac{1}{2}(I - X) = |-\rangle\langle-|, \tag{3.22}$$
$$\rho_3' = \frac{1}{2}(I + Z) = |0\rangle\langle 0|, \tag{3.23}$$
$$\rho_4' = \frac{1}{2}(I - Y) = |i-\rangle\langle i-|. \tag{3.24}$$

Finally, we solve Eq. (3.5) to obtain

$$\chi = \frac{1}{2}\begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \end{pmatrix}. \tag{3.25}$$

Quantum process tomography remains a benchmark standard for quantum computing devices [82]. However, its time complexity is known to scale doubly exponentially in the number of qubits n, and is therefore a costly benchmark [83]. The method is also known to be susceptible to state preparation and measurement (SPAM) errors [36].

3.1.3 Randomized Benchmarking

Randomized benchmarking is a method to obtain the average gate fidelity of a quantum computing device. The method scales polynomially to the number of qubits [84] and is robust to SPAM error [75]. There are various versions of randomized benchmarking, such as interleaved randomized benchmarking [34] or approximate randomized benchmarking [35]. Here, we focus on what is often called the standard Clifford randomized benchmarking [75]. Randomized benchmarking is carried out in the following order:

1. Prepare the system in the initial state $|0\rangle^{\otimes n}$.
2. Apply a sequence of N_c randomly chosen Clifford gates: $C_{N_c} \cdots C_2 C_1 |0\rangle^{\otimes n}$.
3. Apply the inverse sequence of Clifford gates so that the system returns to the initial state in the ideal case: $C_1^{-1} \cdots C_{N_c}^{-1} C_{N_c} \cdots C_2 C_1 |0\rangle^{\otimes n}$.
4. Apply measurement to the system to check whether it returns to the initial state $|0\rangle^{\otimes n}$.
5. Repeat 1 through 4 for various values of N_c. Take the average of the results to estimate the fidelity of the gates.

Clifford gates include the Hadamard gate, Pauli gate, controlled-NOT gate, and phase gate. These gates transform the tensor products of the Pauli matrices into other tensor products of the Pauli matrices by conjugation [10, 85]. An example of a randomized benchmarking circuit for a single qubit is provided in Fig. 3.1.

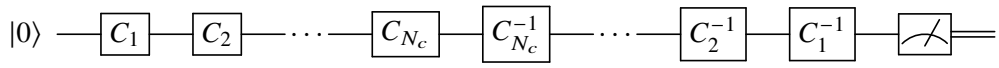


Fig. 3.1. A random sequence of Clifford gates is applied to the initial state $|0\rangle$, followed by a sequence of inverse gates.

To illustrate the process, we now present an example of randomized benchmarking for a single qubit. For example, let N_c be $\{1, 2, 3\}$, and the corresponding gate sequences be X, HZ, and YHZ , respectively. For $N_c = 1$, we first apply the gate X to the initial state $|0\rangle$. We then apply the inverse gate X^{-1} and measure in the computational basis. By repeatedly preparing and measuring this circuit, we obtain the probability of recovering the initial state $|0\rangle$. For instance, let the probability of observing the initial state be:

- $N_c = 1$: $\Pr[N_c = 1] = 0.9$;
- $N_c = 2$: $\Pr[N_c = 2] = 0.7$;
- $N_c = 3$: $\Pr[N_c = 3] = 0.5$.

We now compute the survival probability $\Pr[N_c]$ of measuring the initial state with the function:

$$\Pr[N_c] = A_0 \lambda^{N_c} + B_0, \tag{3.26}$$

where A_0 is a scaling factor, λ is the decay rate, and B_0 is the offset. The result of fitting the values $\Pr[N_c = 1], \Pr[N_c = 2], \Pr[N_c = 3]$ with the survival probability is shown in Fig. 3.2.

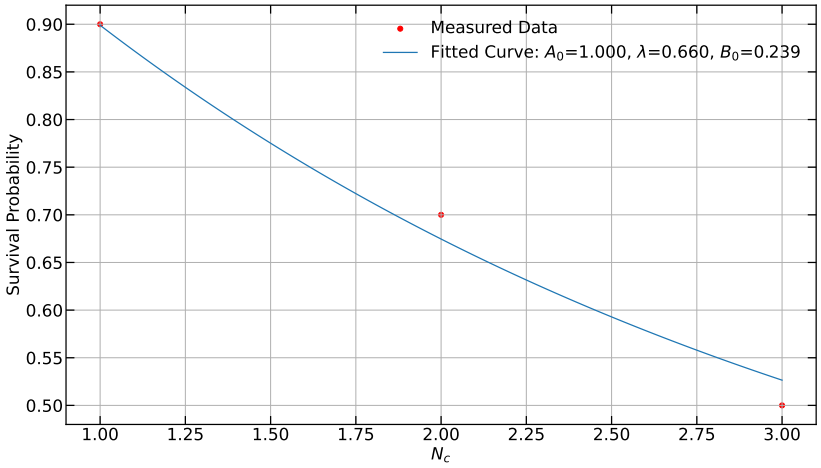


Fig. 3.2. The measured survival probability for different sequence lengths N_c and the fitted curve using the function $\Pr[N_c] = A_0\lambda^{N_c} + B_0$.

We now have the decay rate $\lambda = 0.66$. This decay rate is equivalent to the average gate fidelity. This result can also be interpreted in the context of depolarizing noise. The decay rate λ is related to the depolarizing error rate r through the equation:

$$\lambda = 1 - \frac{2^n - 1}{2^n} r, \tag{3.27}$$

where n is the number of qubits. Solving for $n = 1$, we obtain:

$$r = 2(1 - \lambda). \tag{3.28}$$

For $\lambda = 0.66$, we get $r = 0.68$.

3.1.4 Quantum Volume

Quantum volume [40] is a single numerical metric that measures the largest random circuit with the same width and depth that a quantum computer can execute successfully [86]. The method is carried out by executing random circuits and by comparing the output distribution to what is expected from a random unitary ensemble [38].

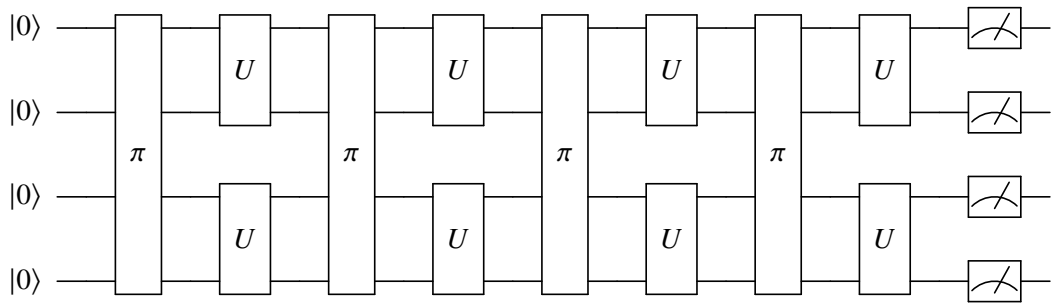


Fig. 3.3. Quantum circuit for measuring quantum volume. The depth of the circuit is $d = 4$. U represents random two-qubit operations, and π denotes random permutations of the qubit labels. When d is an odd number, one qubit is left out in each layer of the circuit.

The metric is computed in the following steps:

1. Construct n_c random quantum circuits with the number of qubits equal to its depth. For example, the circuit in Fig. 3.3 illustrates the circuit with the number of qubits and depth $d = 4$.
2. Execute each random quantum circuit n_c times to obtain the measurement results.
3. Compare the output distributions with the theoretical output distributions to evaluate the heavy output generation rates.
4. If the success rate is at least $\frac{2}{3}$, increase d by one and repeat from Step 1.
5. The largest depth where the success rate meets the threshold is denoted d^*. The quantum volume is then computed as 2^{d^*}.

For example, we illustrate a sample circuit for $d = 2$ in Fig. 3.4.

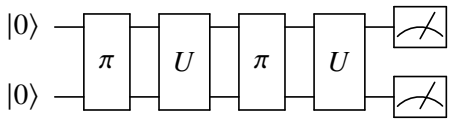


Fig. 3.4. A quantum volume circuit with $d = 2$.

First, the ideal output distribution of the random circuit is computed classically. We then execute the circuit n_s times to obtain the histogram of the resulting states, capturing the frequency of occurrence for each state. The states are re-ordered from the least probable to the most probable to find the median state. The number of observations of states larger than the median is recorded as the heavy output. The process is repeated in $n_c > 100$ random circuits. Let the sum of the number of observations of states larger than the median be denoted n_h. Then, the following value is computed:

$$C = \frac{n_h - z\sqrt{n_h\left(n_s - \frac{n_h}{n_c}\right)}}{n_c n_s}, \tag{3.29}$$

where z is the $z-$score set at 2, which corresponds to the 97.5 % one-sided confidence interval [86]. If $C > 2/3$, then the device passes the test for $d = 2$, and is tested with random circuits with larger values for d, until the test fails. The quantum volume is given by the largest value of d^* for which the device passes the test, expressed as

$$QV = 2^{d^*}. \tag{3.30}$$

Because quantum volume requires the classical computation of the ideal output distribution for every random circuit, it is difficult to implement for $d^* > 30$ [38].

3.1.5 Benchmarking Stability

We have introduced methods to benchmark quantum computing devices so far. However, these methods characterize the device at a specific point in time, and their values are not constant. In reality, the state of a quantum computing device continuously changes, and the ability of a device to produce reproducible results is a highly important topic [39, 45–53]. A stable quantum computing device consistently produces the same probability distribution over time (temporal stability) and across qubits (spacial stability) [45]. Here, we review one of the basic methods for benchmarking stability.

One way to characterize stability is to assess the device metrics in terms of moment-based distance (MBD) [45]. The MBD denoted $d(f, g)$ between two histograms f and g is defined as:

$$d(f, g) = \sum_{j=0}^{\infty} S_j(f, g) \tag{3.31}$$

where

$$S_j(f, g) = \frac{1}{(j)!} \int_a^b \left| \left(\frac{x}{\gamma}\right)^j (f(x) - g(x)) \right| dx \tag{3.32}$$

and

$$\gamma = \max(|a|, |b|). \tag{3.33}$$

Here, a and b are the minimum and maximum values of x, which can be derived from theoretical considerations or empirical histogram data. $d(f, g)$ is known to have the following properties [45]:

1. $d(f, g) \geq 0$, which follows directly from the definition of d;
2. $d(f, g) = d(g, f)$, demonstrating the symmetry of the distance;
3. $d(f, g) = 0$ if and only if $f(x) = g(x)$, ensuring that the distance is zero only when the two distributions are identical;
4. $d(f, g) \leq d(f, h) + d(h, g)$, satisfying the triangle inequality;
5. The series $d = S_0 + S_1 + S_2 + \cdots$ converges.

$j = 4$ is reported to be sufficient to characterize stability [45].

To illustrate, let us consider the case of assessing the temporal stability of a qubit in terms of its gate fidelity. The gate fidelity takes values ranging from 0 to 1, so we have $\gamma = 1$. S_j for $j = 0, 1, 2, 3$, and 4 can be calculated as

$$S_0(f, g) = \int_a^b |(f(x) - g(x))| \, dx, \tag{3.34}$$

$$S_1(f, g) = \int_a^b |x(f(x) - g(x))| \, dx, \tag{3.35}$$

$$S_2(f, g) = \frac{1}{2} \int_a^b \left|x^2(f(x) - g(x))\right| \, dx, \tag{3.36}$$

$$S_3(f, g) = \frac{1}{6} \int_a^b \left|x^3(f(x) - g(x))\right| \, dx, \tag{3.37}$$

$$S_4(f, g) = \frac{1}{24} \int_a^b \left|x^4(f(x) - g(x))\right| \, dx. \tag{3.38}$$

Thus, we obtain $d_4 = S_0 + S_1 + S_2 + S_3 + S_4$. The reference distribution f is taken at an initial time, while the compared distribution g is taken at a subsequent time. The metric d_4 is calculated multiple times to obtain the fluctuation of the metric over time. To assess spatial stability, a qubit is selected as the reference qubit and d_4 is calculated between all pairs of qubits.

3.2 Fundamentals of Random Number Generation

True random number generators (TRNGs) produce sequences of unbiased and independent 0s and 1s [42]. Due to the uniformity and lack of patterns in their output, the results cannot be predicted with greater accuracy than by chance. Quantum computers, devices that leverage the laws of quantum mechanics to perform computations, can, in principle, function as TRNGs [43]. In contrast, classical computers cannot function as TRNGs. This is because they are bound by the laws of classical mechanics, where the variables within a closed system can be determined at any point in time from the initial conditions [87]. In the present section, we review basic concepts in random number generation, including various definitions of randomness, randomness extraction, and the classification of random number generators. The relationship between the concepts is illustrated in Fig. 3.5, where a physical system is measured to produce raw bits, which are then fed into a randomness extractor to extract a uniform random number sequence.

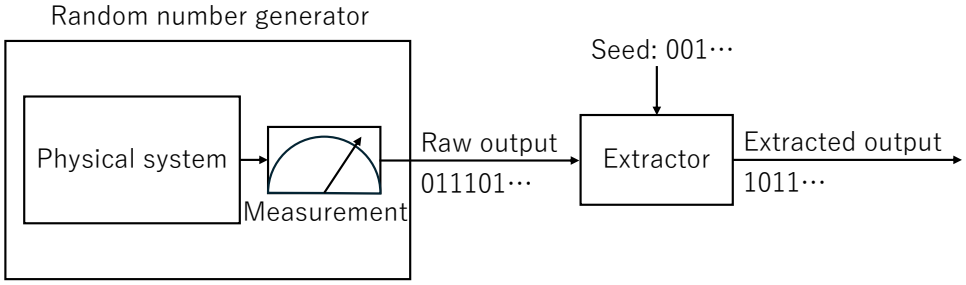


Fig. 3.5. A diagram of random number generation where a physical system is measured to produce raw output, which is then post-processed by a randomness extractor using a seed to generate the final output.

Random number generation is a physical process. Depending on the type of physical process involved, a random number generator can be classified as shown in Fig. 3.6. Pseudorandom Number Generators (PRNGs) are also physical, as they are implemented on classical computers whose underlying processes are governed by classical mechanics.

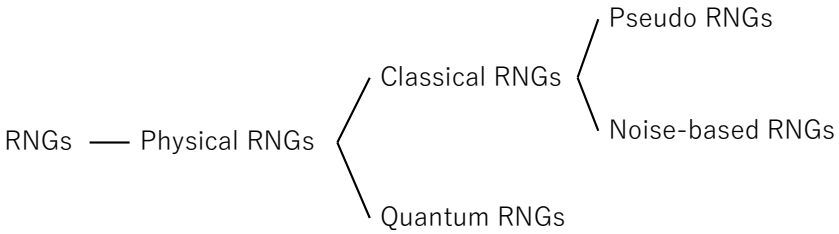


Fig. 3.6. A diagram illustrating the classification of random number generators based on the underlying physical process.

3.2.1 Randomness

The definitions of randomness can be classified into two categories: process randomness and product randomness. Process randomness is the randomness of the physical process, whereas product randomness is the randomness of the output sequence [88]. An example of process randomness is a balanced beamsplitter experiment, where a photon is directed towards a beamsplitter, and it has an equal probability of being transmitted or reflected. Process randomness in this case arises from the Born rule in quantum mechanics, which states that an individual quantum measurement is unpredictable in principle [65]. Product randomness, on the other hand, has several definitions, outlined below:

1. **von Mises' definition [88–90]**
   An infinitely long sequence $a_1, a_2, \ldots$ is considered random when its independently selected subsequences have the same limit relative frequencies of outcomes as the entire sequence. Namely, the following conditions must be met for a sequence to be considered random:

   (a) Let $f(r)$ be the number of 1s in the first r terms of $a_1, a_2, \cdots$. Then $f(r)/r$ approaches a limit p as $r \to \infty$.

   (b) Let $a_{n_1}, a_{n_2}, \cdots$ be any infinite subsequence of $a_1, a_2, \cdots$, formed by deleting terms according to a rule that depends only on n and $a_1, a_2, \cdots, a_{n-1}$. Let $g(r)$ be the number of 1s in the first r terms of $a_{n_1}, a_{n_2}, \cdots$. Then, $g(r)/r$ also approaches the same limit p as $r \to \infty$.

2. **Church's definition [88, 90]**
   Church made von Mises' definition more precise by defining the selection of subsequences as recursive functions. In Church's definition, a random sequence fulfills the following conditions:

   (a) Let $f(r)$ be the number of 1s in the first r terms of $a_1, a_2, \cdots$. Then $f(r)/r$ approaches the limit p as $r \to \infty$.

   (b) If ϕ is any effectively calculable function of positive integers, if $b_1 = 1$, $b_{n+1} = 2b_n + a_n$, $c_n = \phi(b_n)$, and the integers n such that $c_n = 1$ form an infinite sequence $n_1, n_2, \cdots$, and if $g(r)$ is the number of 1s in the first r terms of $a_{n_1}, a_{n_2}, \cdots$, then $g(r)/r$ approaches the same limit p as $r \to \infty$.

3. **Martin-Löf's definition [88, 91–94]**
   Martin-Löf used computable hypothesis tests to define randomness. Namely, an infinite sequence is random if it passes any hypothesis test assuming ideal probability distribution at arbitrarily small levels of significance.

4. **Kolmogorov, Chaitin, and Solomonoff's definition [95–98]**
   Kolmogorov, Chaitin, and Solomonoff defined randomness from an algorithmic perspective. They defined a random sequence as a sequence with no shorter way (program) to describe it than the sequence itself. That is, a string x is considered random if its shortest possible description $d(x)$ is at least as long as the string itself. These strings cannot be compressed.

In practice, product randomness is statistically checked by performing a finite number of hypothesis tests on a finite number of sequences of finite length with the aim of detecting as many patterns as possible.

3.2.2 Randomness extractors

In random number generation, it is often the case that the RNG does not output completely unbiased and independent sequences of 0s and 1s. Depending on the characterization of the generator, methods exist to extract random sequences from such generators. These methods are called randomness extractors. The relationship between an RNG and a randomness extractor is illustrated in Fig. 3.5. A physical system is measured to obtain the raw output, which is then fed into the randomness extractor to distill random bits. For example, one of the most primitive randomness extractors is the von Neumann extractor [99]. It works by taking pairs of consecutive bits from a random sequence and applying the following rule:

1. If the two bits don't match, keep the first bit ($01 \to 0, 10 \to 1$);

2. If the two bits match, discard the pair (00, 11 are discarded).

Assuming that the generator produces biased but independent bits, the extractor removes the bias. For example, let the probability of 0s and 1s before extraction be denoted p and $1 - p$, respectively. Then, the probabilities of observing the pairs are:

$$\Pr(00) = p^2, \tag{3.39}$$
$$\Pr(01) = p(1 - p), \tag{3.40}$$
$$\Pr(10) = (1 - p)p, \tag{3.41}$$
$$\Pr(11) = (1 - p)^2. \tag{3.42}$$

The probability of 0s and 1s in the extracted sequence is balanced. Randomness extractors, such as the von Neumann extractor, are called deterministic extractors. Deterministic extractors are extractors that do not require external sources of randomness [100]. It is known that deterministic extractors cannot extract a single bit from certain kinds of distribution. An example of such a distribution is the Santha-Vazirani source [101]. A Santha-Vazirani source is a source of randomness in which for every $1 \leq i \leq n$ and $x_1, \ldots, x_{i-1} \in \{0, 1\}$, the following relation holds:

$$p_0 \leq \Pr[X_i = 1 \mid X_1 = x_1, \ldots, X_{i-1} = x_{i-1}] \leq 1 - p_0, \tag{3.43}$$

where p_0 is the minimum probability of observing $X_i = 1$. This source does not allow for randomness extraction through deterministic extractors.

The limitation of deterministic extractors leads to the development of seeded extractors, which are extractors that use external sources of randomness for extraction [100]. An important metric related to seeded randomness extractors is the min-entropy. The min-entropy H_∞ of a random variable X on $\{0, 1\}^n$ is defined by

$$H_\infty(X) = -\log_2 \left(\max_{x \in \{0,1\}^n} \Pr[X = x] \right). \tag{3.44}$$

It takes values ranging from 0 to 1, and as it approaches 1, the distribution becomes closer to uniform. The min-entropy can be interpreted as the proportion of random bits contained in a sequence on average. For example, a 100-bit long sequence from a generator whose min-entropy is 0.8 would suggest that the sequence contains at most 80 random bits on average. The Shannon entropy $H(X)$, on the other hand, is a less conservative measure of randomness defined as

$$H(X) = -\sum_{x \in \{0,1\}^n} \Pr[X = x] \log_2 \left(\Pr[X = x] \right). \tag{3.45}$$

It captures the average uncertainty in the outcome of X. We have the following inequality:

$$H_\infty(X) \leq H(X), \tag{3.46}$$

as seen in Fig. 3.7. This inequality holds because the min-entropy measures the unpredictability of the most probable outcome, while Shannon entropy takes into account the distribution of all possible outcomes.

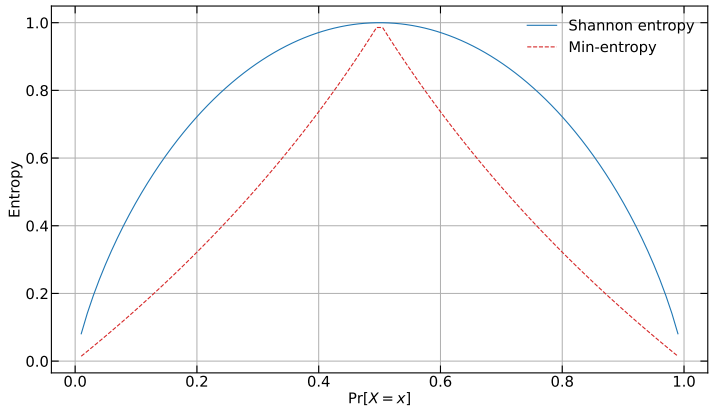


Fig. 3.7. Comparison of Shannon Entropy and min-entropy. The min-entropy is always no larger than Shannon entropy.

The min-entropy is a metric for process randomness, but in most cases where we only have access to the output, we estimate it from the observed output by various statistical means. NIST SP 800-90B [102] is a collection of such statistical means. To illustrate the process of min-entropy estimation, we explain the most-common-value estimate.

The most-common-value estimate is a method of min-entropy estimation that finds the proportion $\hat{p}$ of the most frequently occurring value in the sequence and computes a confidence interval for this proportion. The test can also be applied to non-binary sequences; here, we provide a description for binary sequences. Given a binary sequence $x_1, x_2, \ldots, x_n$, we computing the proportion of the most common bit $\hat{p}$. The upper bound on $\hat{p}$ is provided by

$$p_u = \min\left(1, \hat{p} + 2.576\sqrt{\frac{\hat{p}(1-\hat{p})}{n-1}}\right),$$

where 2.576 corresponds to the 99 % confidence interval. The min-entropy is then estimated as $-\log_2(p_u)$. Note that a sequence such as $0101010101\ldots$ results in a min-entropy of 1. Therefore, it is recommended to perform various estimates and to use the minimum (worst case) of all estimates as the estimated min-entropy [102].

To define seeded randomness extractors, another important concept is the notion of ε-closeness. A distribution X is said to be ε-close to a distribution Y when the statistical distance $|X-Y|$ has the relation:

$$\|X - Y\| = \frac{1}{2}\sum_{x\in\{0,1\}^n} |\Pr[X = x] - \Pr[Y = x]| \leq \varepsilon. \tag{3.47}$$

Now we provide a definition of seeded extractors. A (k, ϵ)-extractor is defined as a function $E : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ that for every distribution X over $\{0,1\}^n$ with $H_\infty(X) \geq k$, $E(X, Y)$ is ϵ-close to uniform, where Y is independent of X [103]. The loss of entropy of a seeded extractor is given by $k+d-m$, which is always nonnegative. The entropy loss is proven to be at least $2\log_2(1/\varepsilon) - O(1)$ [104]. As examples, we present two seeded extractors: the Toeplitz extractor and Trevisan's extractor.

The Toeplitz extractor and the Trevisan extractor are strong (k, ϵ)-extractors, meaning that their output is independent of the seed [105, 106]. Let us consider the input $x_1, x_2, \ldots, x_n$ with a min-entropy of at least k. This means that at most nk bits can be extracted. The seed required to construct the $m \times n$

Toeplitz matrix is denoted $a_1, a_2, \ldots, a_{m+n-1}$. The Toeplitz matrix is constructed as follows:

$$\begin{bmatrix} a_m & a_{m+1} & \cdots & a_{m+n-2} & a_{m+n-1} \\ a_{m-1} & a_m & a_{m+1} & \cdots & a_{m+n-2} \\ \vdots & a_{m-1} & \ddots & \vdots & \vdots \\ a_2 & \vdots & \ddots & a_n & a_{n+1} \\ a_1 & a_2 & \cdots & a_{n-1} & a_n \end{bmatrix}.$$

The extracted sequence $y_1, y_2, \ldots, y_m$ is obtained by multiplying the raw sequence by the Toeplitz matrix:

$$\begin{bmatrix} a_m & a_{m+1} & \cdots & a_{m+n-2} & a_{m+n-1} \\ a_{m-1} & a_m & a_{m+1} & \cdots & a_{m+n-2} \\ \vdots & a_{m-1} & \ddots & \vdots & \vdots \\ a_2 & \vdots & \ddots & a_n & a_{n+1} \\ a_1 & a_2 & \cdots & a_{n-1} & a_n \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ \vdots \\ y_m \end{bmatrix}.$$

Note that k is the maximum extraction rate, and it is customary to extract at a lower rate for safety. The Toeplitz extractor requires a seed length of only linear scaling of the input length [107].

Trevisan's extractor [108] is a strong extractor that requires a seed length that scales only poly-logarithmically with the input length [107]. It is also known to be secure in the presence of quantum side information [109]. The idea of Trevisan's extractor is to use a specific class of error-correcting codes called list-decodable codes [110] to expand the seed into multiple packets allowing overlap, and to choose indices of the input sequence based on the packets [111]. Each packet is used to extract a single bit from the input sequence by performing the XOR operation on the bits corresponding to the chosen indices.

3.2.3 Pseudorandom Number Generators (PRNGs)

PRNGs are deterministic functions implemented on classical computers that output bits that are apparently random [112]. The output is completely determined by the function and the seed, and the resulting sequence always has patterns and periodicity [113]. However, because the sequence can be quickly and consistently reproduced from the seed and function, PRNGs are widely used in scenarios where true random number generators (TRNGs) are not necessary. Next, we will introduce several examples of PRNGs: the linear congruential generator and the Mersenne Twister generator. We also illustrate the Blum Blum Shub (BBS) generator, which is a type of cryptographically secure PRNGs (CSPRNGs).

The linear congruential generator (LCG) [114] is a simple generator of the form:

$$X_{n+1} = (aX_n + c) \mod m, \tag{3.48}$$

where X_n is the current state, a is the multiplier, c is the increment, and m is the modulus. Each output is typically normalized by dividing it by m or converting it to its binary representation. The sequence of pseudorandom numbers is determined by the initial seed X_0 and the choice of parameters a, c, and m. The combination of seed and parameters has been extensively researched [115–117]. The linear congruential generator (LCG) has the advantage of being simple and easy to implement. With properly chosen parameters, it can achieve a maximum period of m, where m is the modulus. However, LCGs have notable disadvantages, such as correlations between successive values and patterns in higher dimensions [118, 119]. The choice of parameters achieves the maximum period of m if and only if it satisfies the following properties [120]:

1. The parameters c and m are coprime.

2. All the prime factors of m divide $a - 1$.

3. If 4 divides m, then 4 also divides $a - 1$.

A poor choice of parameters may lead to patterns as seen in Fig. 3.8.

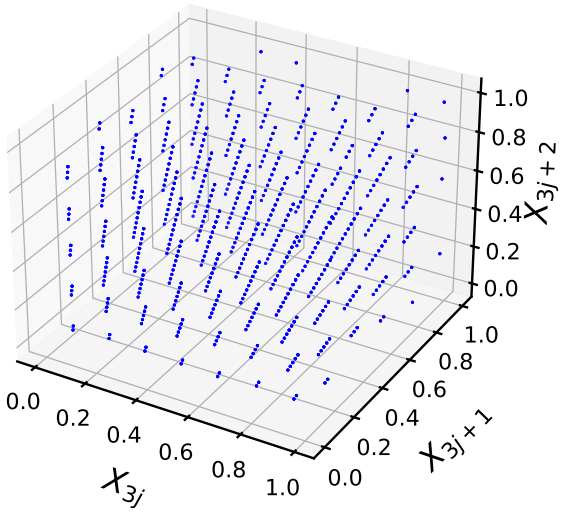


Fig. 3.8. An example of periodic patterns in LCG with $a = 122, c = 321, m = 1234, X_0 = 37$. Note that i is an integer.

The Mersenne Twister is a PRNG with 623-dimensionally equidistributed uniform pseudorandom number generator with a maximum period of $2^{19937} - 1$ [121, 122]. The generator is defined by the function:

$$x_{k+n} = x_{k+m} + (x_k^u | x_{k+1}^l)A \quad (j = 0, 1, 2, \ldots), \tag{3.49}$$

where $x_0, x_1, \ldots$ are 32-dimensional binary vectors, A a 32×32 binary matrix, n, m non-zero positive integers, and | the bit-wise OR operation. The notation x_k^u means the upper r bits of x_k and x_k^l the lower r bits. Matrix A is defined as

$$\begin{pmatrix} 0 & I_{w-1} \\ a_{w-1} & (a_{w-2}, \ldots, a_0) \end{pmatrix},$$

where I_{w-1} is a $(w-1) \times (w-1)$ identity matrix. The parameters of the generator so far are: integers n, m, r, the word size w, and the coefficients for the matrix A $\mathbf{a} = (a_{w-1}, a_{w-2}, \ldots, a_0)$. The seed for the generator is a w-bit integer x_0, from which $x_1, \ldots, x_{n-1}$ are obtained by the recurring function

$$x_i = f \times (x_{i-1} \oplus (x_{i-1} \gg (w-2))) + i, \quad i = 1, 2, \ldots, n-1,$$

where f is a w-bit integer. To improve the distribution of the output, each generated vector x in Eq. (3.49) is multiplied by a tempering matrix T from the right, which is equivalent to performing the computations

$$\begin{aligned} x &= x \oplus (x \gg u), \\ x &= x \oplus ((x \ll s)\&b), \\ x &= x \oplus ((x \ll t)\&c), \\ x &= x \oplus (x \gg l), \end{aligned}$$

where u, s, t, l are integers, b, c are w-bit integers. $x \gg u$ means to shift the w-bit integer x u bits to the right, and $x \ll u$ to the left. The bit-wise operations | and & are defined as follows:

Table 3.1. The truth table of the bit-wise | and & operations between x and y.

x	y	$x\|y$	$x\&y$
0	0	0	0
0	1	1	0
1	0	1	0
1	1	1	1

The generator is called MT19937 when the following parameters are used [121]:

- $w = 32$,
- $n = 624$,
- $m = 397$,
- $r = 31$,
- $u = 11$,
- $s = 7$,
- $t = 15$,
- $l = 18$,
- $a = 0x9908B0DF$,
- $b = 0x9D2C5680$,
- $c = 0xEFC60000$,
- $f = 1812433253$.

MT19937 has been implemented in many programming languages and software tools. For instance, it is available in the <random> header of C++11 (and later C++14) standard template library (STL) [123].

CSPRNGs are PRNGs that meet the following criteria [124]: they pass statistical tests for randomness and remain secure under intense attacks, even if an attacker gains partial knowledge of their initial or ongoing state. The BBS generator [125] is a type of CSPRNG. The generator function of the BBS generator is given by

$$x_{n+1} = x_n^2 \mod M,$$

where x_0 is the seed and M the product of two integers p, q, such that $p = q = 3 \mod 4$ and $gcd(p - 1, q - 1)$ is small [125, 126]. Instead of using the numbers directly, the bit parity of the numbers is used [127]:

$$X_i = x_0 \mod 2.$$

The ith bit of the BBS generator is given by

$$x_i = \left(x_0^{2^i \mod (p-1)(q-1)}\right) \mod M.$$

To predict the ith bit, not only the knowledge of M, but also the knowledge of its prime factors p and q is required. The security foundation of the BBS generator is that when M is a large prime, it is computationally difficult to derive p and q with the known factoring algorithms on classical computers. However, quantum algorithms that run in polynomial time could threaten the security of this generator [128].

3.2.4 Quantum Random Number Generators (QRNGs)

QRNGs are a type of TRNGs whose output arises from quantum events [42]. Depending on which components of the generator are trusted, QRNGs can be classified into five categories, as shown in Table 3.2.

Table 3.2. Classification of QRNGs with required assumptions and typical implementation.

Type	Assumptions	Implementation
Device-independent (DI)	Violation of a Bell inequality.	Entangled photons created and measured along randomly chosen basis at two spatially separated locations [129].
Semi-device-independent (SDI)	The dimension of the physical system or some components of the device are trusted	An untrusted laser source measured by trusted heterodyne detection system [130].
Trusted device (TD)	All components of the device are fully trusted	Radioactive source measured by detector [1].

The earliest quantum random number generators (QRNGs) belonged to the trusted device (TD) category. TD-QRNGs are systems whose physical processes are thoroughly characterized and are considered reliable [131]. A notable example of a TD-QRNG is one that uses radioactive decay to generate randomness [1]. A typical setup of such a radioactivity-based TD-QRNG is illustrated in Fig. 3.9.

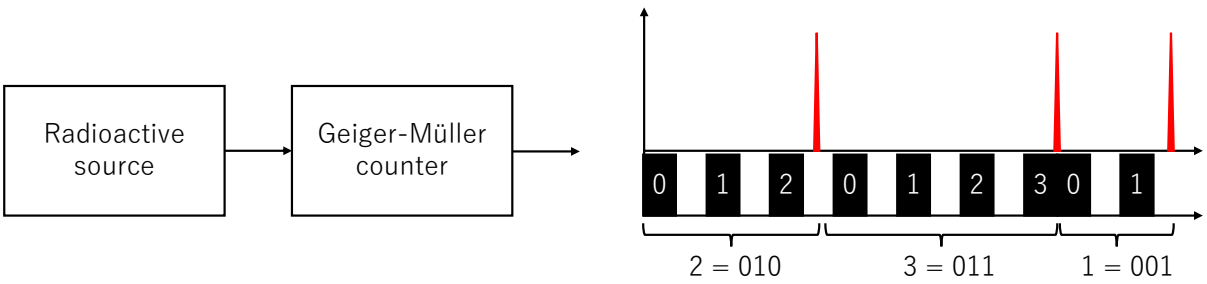


Fig. 3.9. A diagram of a radioactivity-based QRNG. The β radiation of a radioactive source is detected by a Geiger-Müller (GM) counter, which then outputs binary signals based on the value of the counter at the time of detection. On the right, the black bars represent the values of the counter, which is reset every time a particle is detected. The red spikes indicate detection events corresponding to the counts [1].

The probability of finding k pulses in an observation period of T seconds is given by

$$P_k(T) = \frac{(\lambda T)^k}{k!} e^{-\lambda T}, \tag{3.50}$$

where λ gives the mean number of pulses we detect in one second [42]. To convert the resulting sequence into uniform bits, the least significant bit is taken [132]. While radioactive-based QRNGs are still in use, they are known to have several drawbacks such as slow bit rate and non-uniformity of bits arising from the detection dead-time of GM counters [42].

The most common type of QRNGs is based on optics [42]. A simple implementation of optical QRNGs measures photons that are in a superposition of two or more possible paths [133]. A typical setup of an optical QRNG is shown in Fig. 3.10.

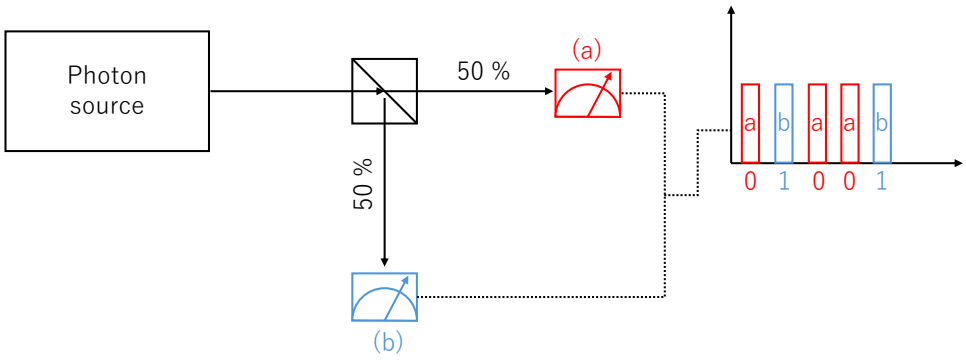


Fig. 3.10. A diagram of an optical QRNG. When a photon is emitted from a photon source, a balanced beamsplitter creates a superposition of two possible paths with equal probability. When the detector (a) detects a photon, 0 is output, and when (b) detects a photon, 1 is output.

A photon emitted from the photon source forms a superposition state:

$$\frac{1}{\sqrt{2}}(|0\rangle_a + |1\rangle_b), \tag{3.51}$$

where $|0\rangle_a$ represents the path leading to detector (a), and $|1\rangle_b$ represents the path leading to detector (b). Given that the beamsplitter is balanced, the probability of detecting the photon at either detector is equal. With these kinds of optical QRNGs, a bias in the beamsplitter and the dead time of detectors cause a non-uniform output [42]. These problems can be mitigated using randomness extractors or by tuning the detection threshold of detectors [134].

Device-independent quantum random number generators (DI-QRNGs) are a revolutionary approach to random number generation, as they enable the certification of process randomness without relying on the trustworthiness of the underlying devices [135]. DI-QRNGs leverage a form of the Bell inequality, where its violation ensures that the output is not predetermined [129]. Here, we introduce a type of DI-QRNG that exploits the Clauser–Horn–Shimony–Holt (CHSH) form of the Bell inequality [136].

The CHSH inequality is used to distinguish between classical and quantum correlations. It is based on the correlation function defined by

$$I = \sum_{x,y} (-1)^{xy} \left[\Pr(a \oplus b = 0|xy) - \Pr(a \oplus b = 1|xy)\right], \tag{3.52}$$

where (x, y) represent the inputs chosen for two spatially separated measurement devices, and (a, b) denote their respective binary outputs. In deterministic scenarios such as classical mechanics, the correlation function has the inequality

$$I \leq 2, \tag{3.53}$$

whereas in quantum mechanics, this inequality can be violated [129]. To illustrate this fact, let us consider the CHSH game. The CHSH game involves two participants, Alice and Bob, and a host, Charlie. Charlie distributes the classical bits x, y to Alice and Bob, respectively. Upon receiving a bit, Alice and Bob each output a classical bit a, b. The aim of Alice and Bob is to maximize the probability of fulfilling the equation

$$a \oplus b = xy. \tag{3.54}$$

Alice and Bob can freely exchange information until the game starts.

Table 3.3. The winning combinations of (a, b) corresponding to (x, y) in the CHSH game.

(x, y)	(a, b)
$(0, 0)$	$(0, 0), (1, 1)$
$(0, 1)$	$(0, 0), (1, 1)$
$(1, 0)$	$(0, 0), (1, 1)$
$(1, 1)$	$(0, 1), (1, 0)$

The optimal combinations of (x, y) and (a, b) that maximize Eq. (3.54) are presented in Table 3.3. The best classical scheme is simple: both Alice and Bob output 0 regardless of which classical bit they receive from Charlie. This achieves a winning probability of 0.75. The respective best classical scheme is as follows.

Alice Alice outputs $a = 0$, regardless of the input x:

- When $x = 0$, output $a = 0$,
- When $x = 1$, output $a = 0$.

Bob Bob also outputs $b = 0$, regardless of the input y:

- When $y = 0$, output $b = 0$,
- When $y = 1$, output $b = 0$.

The resulting winning probability can be calculated as follows.

Winning probability The winning probability is the sum of the winning probabilities of all combinations of (x, y) and (a, b):

$$\begin{aligned}\Pr(\text{win}) &= \Pr(\text{win}|xy = 00) + \Pr(\text{win}|xy = 01) \\ &\quad + \Pr(\text{win}|xy = 10) + \Pr(\text{win}|xy = 11) \\ &= 0.25 + 0.25 + 0.25 + 0 \\ &= 0.75.\end{aligned}$$

The CHSH value can be calculated as

$$\begin{aligned}I &= \{\Pr(a \oplus b = 0|xy = 00) - \Pr(a \oplus b = 1|xy = 00)\} \\ &\quad + \{\Pr(a \oplus b = 0|xy = 01) - \Pr(a \oplus b = 1|xy = 01)\} \\ &\quad + \{\Pr(a \oplus b = 0|xy = 10) - \Pr(a \oplus b = 1|xy = 10)\} \\ &\quad - \{\Pr(a \oplus b = 0|xy = 11) - \Pr(a \oplus b = 1|xy = 11)\} \\ &= (1 - 0) + (1 - 0) + (1 - 0) - (1 - 0) \\ &= 2.\end{aligned}$$

Here, we observe that the best classical scheme cannot violate the CHSH inequality. The best quantum scheme, on the other hand, involves Alice and Bob performing measurements on entangled qubits, where the types of measurement are chosen based on the classical bits they receive from Charlie. Alice and Bob share the qubits in the entangled state:

$$\begin{aligned}|\psi\rangle &= |\psi_{\text{Alice}}\rangle \otimes |\psi_{\text{Bob}}\rangle \\ &= \frac{1}{\sqrt{2}} \left(|00\rangle + |11\rangle\right).\end{aligned} \tag{3.55}$$

Upon receiving a classical bit from Charlie, Alice and Bob perform measurements on their qubits in the basis $\{|\phi_0(\theta)\rangle, |\phi_1(\theta)\rangle\}$, where

$$|\phi_0(\theta)\rangle = \cos\theta\,|0\rangle + \sin\theta\,|1\rangle\,, \tag{3.56}$$
$$|\phi_1(\theta)\rangle = -\sin\theta\,|0\rangle + \cos\theta\,|1\rangle\,, \tag{3.57}$$

and the measurement angles θ are determined by x, y. This scheme achieves a winning probability of approximately 0.85, as illustrated below.

Alice Alice decides the measurement basis according to x as follows:

- When $x = 0$, measure along the basis $\{|\phi_0(\theta_{A0})\rangle, |\phi_1(\theta_{A0})\rangle\}$.
- When $x = 1$, measure along the basis $\{|\phi_0(\theta_{A1})\rangle, |\phi_1(\theta_{A1})\rangle\}$.

Bob Bob also decides the measurement basis according to y as follows:

- When $y = 0$, measure along the basis $\{|\phi_0(\theta_{B0})\rangle, |\phi_1(\theta_{B0})\rangle\}$.
- When $y = 1$, measure along the basis $\{|\phi_0(\theta_{B1})\rangle, |\phi_1(\theta_{B1})\rangle\}$.

Winning probability The winning probability is given by the sum of the winning probabilities of all combinations of (x, y) and (a, b) as

$$\begin{aligned}\Pr(\text{win}) = \frac{1}{4}\{&\Pr(ab = 00|xy = 00) + \Pr(ab = 11|xy = 00)\\ &+ \Pr(ab = 00|xy = 01) + \Pr(ab = 11|xy = 01)\\ &+ \Pr(ab = 00|xy = 10) + \Pr(ab = 11|xy = 10)\\ &+ \Pr(ab = 01|xy = 11) + \Pr(ab = 10|xy = 11)\ \}\,.\end{aligned}$$

The respective probabilities are:

$$\Pr(ab = 00|xy = 00) = \frac{1}{2}\cos^2(\theta_{A0} - \theta_{B0}), \tag{3.58}$$
$$\Pr(ab = 11|xy = 00) = \frac{1}{2}\cos^2(\theta_{A0} - \theta_{B0}), \tag{3.59}$$
$$\Pr(ab = 00|xy = 01) = \frac{1}{2}\cos^2(\theta_{A0} - \theta_{B1}), \tag{3.60}$$
$$\Pr(ab = 11|xy = 01) = \frac{1}{2}\cos^2(\theta_{A0} - \theta_{B1}), \tag{3.61}$$
$$\Pr(ab = 00|xy = 10) = \frac{1}{2}\cos^2(\theta_{A1} - \theta_{B0}), \tag{3.62}$$
$$\Pr(ab = 11|xy = 10) = \frac{1}{2}\cos^2(\theta_{A1} - \theta_{B0}), \tag{3.63}$$
$$\Pr(ab = 01|xy = 11) = \frac{1}{2}\sin^2(\theta_{A1} - \theta_{B1}), \tag{3.64}$$
$$\Pr(ab = 10|xy = 11) = \frac{1}{2}\sin^2(\theta_{A1} - \theta_{B1}). \tag{3.65}$$

We therefore have

$$\Pr(\text{win}) = \frac{1}{4}\left\{\cos^2(\theta_{A0} - \theta_{B0}) + \cos^2(\theta_{A0} - \theta_{B1})\right. \tag{3.66}$$
$$\left. + \cos^2(\theta_{A1} - \theta_{B0}) + \sin^2(\theta_{A1} - \theta_{B1})\ \right\}. \tag{3.67}$$

When θ is chosen as

$$\{\theta_{A0}, \theta_{A1}, \theta_{B0}, \theta_{B1}\} = \left\{0, \frac{\pi}{4}, \frac{\pi}{8}, -\frac{\pi}{8}\right\}, \tag{3.68}$$

the winning probability is maximized as

$$\Pr(\text{win}) = \frac{3}{4}\cos^2\left(\frac{\pi}{8}\right) + \frac{1}{4}\sin^2\left(\frac{3\pi}{8}\right) \tag{3.69}$$

$$= 0.85355\ldots \approx 0.85. \tag{3.70}$$

The CHSH value is

$$\begin{aligned}
I &= \{\Pr(a \oplus b = 0|xy = 00) - \Pr(a \oplus b = 1|xy = 00)\} \\
&+ \{\Pr(a \oplus b = 0|xy = 01) - \Pr(a \oplus b = 1|xy = 01)\} \\
&+ \{\Pr(a \oplus b = 0|xy = 10) - \Pr(a \oplus b = 1|xy = 10)\} \\
&- \{\Pr(a \oplus b = 0|xy = 11) - \Pr(a \oplus b = 1|xy = 11)\} \\
&= \left\{\cos^2(\theta_{A0} - \theta_{B0}) - \sin^2(\theta_{A0} - \theta_{B0})\right\} \\
&+ \left\{\cos^2(\theta_{A0} - \theta_{B1}) - \sin^2(\theta_{A0} - \theta_{B1})\right\} \\
&+ \left\{\cos^2(\theta_{A1} - \theta_{B0}) - \sin^2(\theta_{A1} - \theta_{B0})\right\} \\
&- \left\{\cos^2(\theta_{A1} - \theta_{B1}) - \sin^2(\theta_{A1} - \theta_{B1})\right\} \\
&= 2\sqrt{2} = 2.82842\ldots.
\end{aligned}$$

We observe that in quantum mechanics it is possible to violate the CHSH inequality by using entangled qubits.

The relationship between the CHSH inequality and QRNGs lies in the fact that when the inequality is violated, the values of (a, b) result from the measurement of entangled quantum states, which, according to the Born rule, are inherently probabilistic [42]. The random number sequence is obtained in the following steps [129]:

1. Prepare two spatially separated and entangled qubits.
2. Generate (x, y) uniformly at random.
3. Measure the respective qubits with the basis chosen by the values of (x, y).
4. Obtain the measurement results (a, b).
5. Repeat steps 1 through 4 for n_s rounds to obtain the output $r = (a_1, b_1, a_2, b_2, \ldots, a_{n_s}, b_{n_s})$.

Once the above steps are completed, the correlation function I is estimated as

$$I = \frac{1}{n_s}\sum_{x,y}(-1)^{xy}[N(a \oplus b = 0|xy) - N(a \oplus b = 1|xy)]/\Pr[xy], \tag{3.71}$$

where N corresponds to the number of observations of each event, and n_s represents the number of times steps 1 through 3 were repeated. The resulting random number sequence $s = (a_1, b_1, a_2, b_2, \ldots, a_{n_s}, b_{n_s})$ contains random bits if the CHSH inequality is violated. The conditional min-entropy $H_\infty(R|S)$ is defined by

$$H_\infty(R|S) = -\log_2\left[\max_r P(r|s)\right], \tag{3.72}$$

where $P(r|s)$ is the conditional probability of obtaining the outcomes r when the measurements s are made, and the maximum is taken over all possible values of the output string r [129]. The conditional min-entropy $H_\infty(R|S)$ has the lower bound

$$H_\infty(R|S) \geq f(I-\varepsilon), \tag{3.73}$$

with probability $(1-\sigma)$, where f is a function of the correlation function I, and $\varepsilon = O(\sqrt{-\log(\sigma/q^2 n_s)})$. Here, q represents the probability of the least probable input pair $\min_{xy} \Pr[xy]$. In previous research, the value of the function f is obtained numerically [129]. Finally, the output $s = (a_1, b_1, a_2, b_2, \ldots, a_{n_s}, b_{n_s})$ is subjected to a randomness extractor at an extraction rate given by the lower bound of conditional min-entropy.

For a violation of the CHSH inequality to be considered valid, all loopholes must be closed. The loophole refers to reasons other than non-local correlations that could cause the CHSH inequality to exceed 2. Some of the common loopholes include [137, 138]:

1. **Locality loophole**: This occurs if the two measurement settings are not space-like separated. This allows for the possibility that Alice's measurement could influence Bob's through local means.

2. **Detection loophole**: This happens if the detection efficiency of the measuring devices is too low, causing a biased subset of measurement results.

3. **Freedom-of-choice loophole**: This takes place if the measurement settings of Alice and Bob are influenced by hidden variables or factors that are correlated with the system being measured.

4. **Memory loophole**: This occurs if the measurement results are correlated, affecting the subsequent measurement results.

The freedom-of-choice loophole suggests that the choice of (x, y) needs to be generated by TRNGs. In a previous study, the freedom-of-choice loophole was closed by using random numbers generated by 100,000 human participants based on choices made in a video game [138].

3.3 Stability Analysis of Quantum Computers

In the present section, we propose a method for benchmarking the stability of the qubits of a quantum computing device. To do so, we first explain the relationship between quantum random number generation and quantum computers. We then review conventional tests for RNGs. The characteristics of the random number sequence generated by the qubits of real devices are presented, along with the reason why conventional tests are limited in their ability to benchmark current quantum computing devices. Finally, we describe our proposed method and the results obtained using a real device.

3.3.1 Quantum Computers as Random Number Generators (RNGs)

Unlike classical computing devices, quantum computing devices are capable of functioning as TRNGs by leveraging the Born rule in quantum mechanics. However, implementing DI-QRNGs on these devices is challenging due to the difficulty in closing various loopholes in the Bell inequality, particularly the locality loophole [43]. Especially with cloud-based quantum computing devices, there are limits to the extent to which users can characterize the physical system, such as how the qubits are spatially located. Furthermore, the random numbers generated by a quantum computing device are subject to noise. Evidence of this is that the raw bits generated by quantum computing devices as TD-QRNGs fail to pass statistical tests without post-processing [44]. Under the circumstances, there are two possible approaches for random number generation on quantum computing devices. One is to employ less noise-sensitive

algorithms and post-processing to enhance the quality of the resulting random number sequence [43]. The other is to leverage the fact that the random number sequence generated by a noisy quantum computing device is influenced by noise, using the sequence as a performance-oriented benchmark for evaluating the quality of its qubits.

3.3.2 Conventional Tests for RNGs

Conventional tests for random number generators consist of a series of hypothesis tests that assume that the generator produces unbiased, independent, and identically distributed (i.i.d.) outputs. Various collections of such tests exist, including the NIST SP 800-22 [6] or the TestU01 [139]. In this subsection, the NIST SP 800-22 [6] is reviewed as an example of how RNGs are tested.

NIST SP 800-22 [6] is a collection of statistical tests aimed to detect random or PRNGs that output binary sequences that deviate from the unbiased and i.i.d. assumption. From a collection of binary random number sequences, the test suite computes the P-values corresponding to each of the 16 tests and decides whether the generator is unbiased and i.i.d. based on two factors: pass rate and P-value uniformity. This process is described in the following diagram.

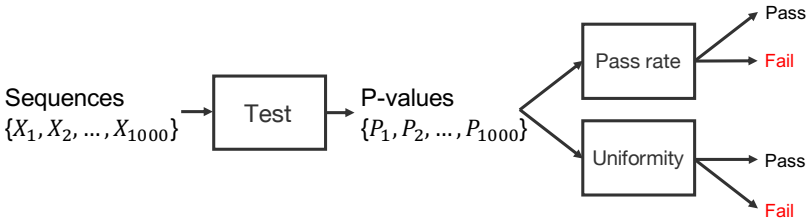


Fig. 3.11. Diagram of how N binary random number sequences go through a single test from NIST SP 800-22.

Subjecting N binary random number sequences to a single test in NIST SP 800-22 [6] will result in N P-values. The pass rate for the P-values is the proportion of the P-values no less than the level of significance α, whose default setting in NIST SP 800-22 [6] is 1 %. Under the assumption that the sequences were obtained from an unbiased and i.i.d random number generator, the pass rate follows the normal distribution whose mean is $\mu = 1 - \alpha$ and standard deviation is $\sigma = \sqrt{\alpha(1-\alpha)/N}$. If the pass rate is within the 99.7 % confidence interval of this distribution, it follows that the generator passes the test in terms of the pass rate. The 99.7 % confidence interval is given as $[\mu - 3\sigma, \mu + 3\sigma]$. This means that while the individual sequences are evaluated under the level of significance of 1 %, the pass rate is evaluated under the level of significance of 0.3 %. The following figure shows the theoretical distribution of the pass rate of 1000 sequences, which is the minimum number of sequences required for the pass rate test. The corresponding 99.7 % confidence interval is colored blue.

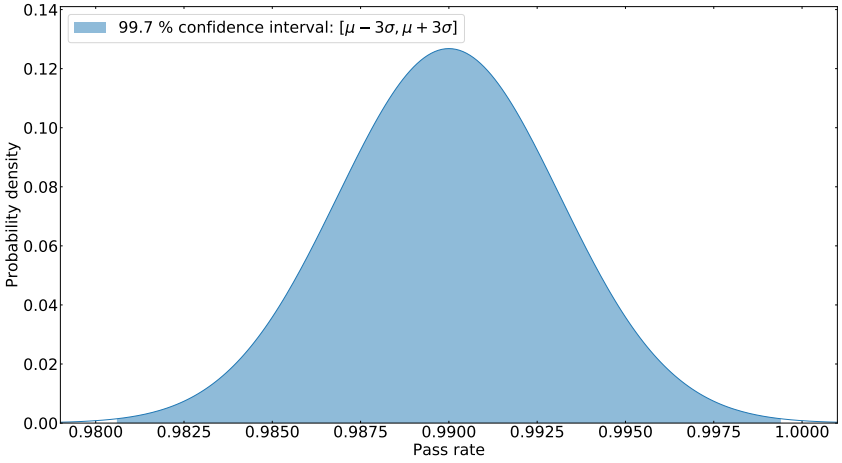


Fig. 3.12. The distribution of the pass rate for $N = 1000$, where the level of significance for each sequence is $\alpha = 0.01$. The corresponding 99.7 % confidence interval is colored in blue.

The second factor is the uniformity of the N P-values, which should be uniform under the unbiased and independent assumption. The uniformity of the P-values is tested by the χ^2 test, where the observed and theoretical numbers of P-values in the regions $[0.1j, 0.1j + 0.1]$ for $j = 0, 1, \ldots, 9$ are compared. If F_j P-values are within the range $[0.1j, 0.1, +0.1]$, the test statistic for the χ^2 test is:

$$\chi^2 = \sum_{j=0}^{9} \frac{F_j - 0.1N}{0.1N},$$

and the P-value is:

$$\text{P-value} = \text{igamc}\left(\frac{10-1}{2}, \frac{\chi^2}{2}\right),$$

where igamc stands for the incomplete gamma function defined previously as Eq. (A.2). If the P-value for the uniformity of the P-values is no less than the level of significance 0.1 %, the sequences pass the P-value uniformity test. Note that the P-value uniformity test requires at least 55 P-values. Below are the default parameters for the tests.

Table 3.4. The default parameters for NIST SP 800-22 [6].

Test Name	Parameters
Frequency Test within a Block	$M = 128$
Binary Matrix Rand Test	$M = Q = 32$
Non-overlapping Template Matching Test	$m = 9$
Overlapping Template Matching Test	$m = 9$
Approximate Entropy Test	$m = 9$
Serial Test	$m = 10$
Linear Complexity Test	$M = 500$

The common assumption behind the 16 tests is the unbiased i.i.d. assumption. Each test is targeted at a specific behavior expected from the assumption. A biased or non-i.i.d. random number generator, therefore, should fail all 16 tests. Refer to Appendix A for details regarding the probability distributions used in tests for RNGs.

3.3.3 Characteristics of the Output of a Real Device

The noise in a quantum computing device can influence the quality of the random number sequence in various ways, one of which is bias. We generated 4,743,168 random bits from each qubit of the IBM 20Q Poughkeepsie device by preparing each qubit in the state $(|0\rangle + |1\rangle)/\sqrt{2}$, and measuring them in the computational basis. The proportion of 1s in the resulting random number sequence for each qubit is shown in Fig. 3.13. The results indicate that without post-processing, the random number sequences produced by the qubits of a quantum computing device are biased.

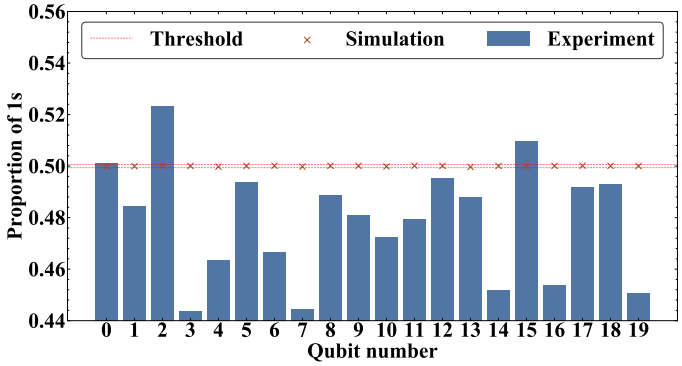


Fig. 3.13. The proportion of 1s of qubit[0] ~ [19]. The acceptable range under the level of significance $\alpha = 0.01$ is between the two dotted lines. The blue bars are the experimental results and the red plots the noisy simulation results. (Reprinted figure from [2]).

We then proceeded to test the random number sequences output by each qubit by dividing them into samples of length 8192. Among the 16 tests in NIST SP 800-22 [6], 8 tests were applicable given the limited length of samples as shown in Table 3.5.

Table 3.5. The minimum length n required for each test in order to obtain meaningful results. The tests not employed in the present study are shaded in grey. Note that the tests will be referred to by their numbers hereafter (Reprinted table from [2]).

Test number	Test name	Minimum length
1	Frequency	$n \geq 100$
2	Frequency within a block	$n \geq 100$
3	Runs	$n \geq 100$
4	Longest run of ones	$n \geq 128$
	Binary matrix rank	$n \geq 38912$
5	DFT	$n \geq 1000$
	Non-overlapping T. M.	$n \geq 8m - 8$
	Overlapping T. M.	$n \geq 10^6$
	Maurer's Universal	$n > 387840$
	Linear complexity	$n \geq 10^6$
	Serial	$n > 16$
6	Approximate entropy	$n > 64$
7	Cumulative sums (forward)	$n \geq 100$
8	Cumulative sums (backward)	$n \geq 100$
	Random excursions	$n \geq 10^6$
	Random excursions variant	$n \geq 10^6$

As a result of applying the 8 tests, the proportion of passed samples under the level of significance

$\alpha = 0.01$ was computed. We found that none of the qubits passed the 8 tests, as shown in Fig. 3.15. Furthermore, we computed the min-entropy of each sample from the proportion of 0s and 1s contained per sample to find that the min-entropy fluctuates across samples and qubits (see Fig. 3.14). To further analyze the characteristics of the output, we plotted $y = \sum_i 2x_i - 1$ for the random number sequence $x_1, x_2, \cdots$ generated by each qubit in Fig. 3.16. The resulting plots should be roughly aligned with the horizontal line $y = 0$, as is the case with the simulation results obtained by pseudorandom number generations. However, we observed breaks in the slope of y resulting from real qubits.

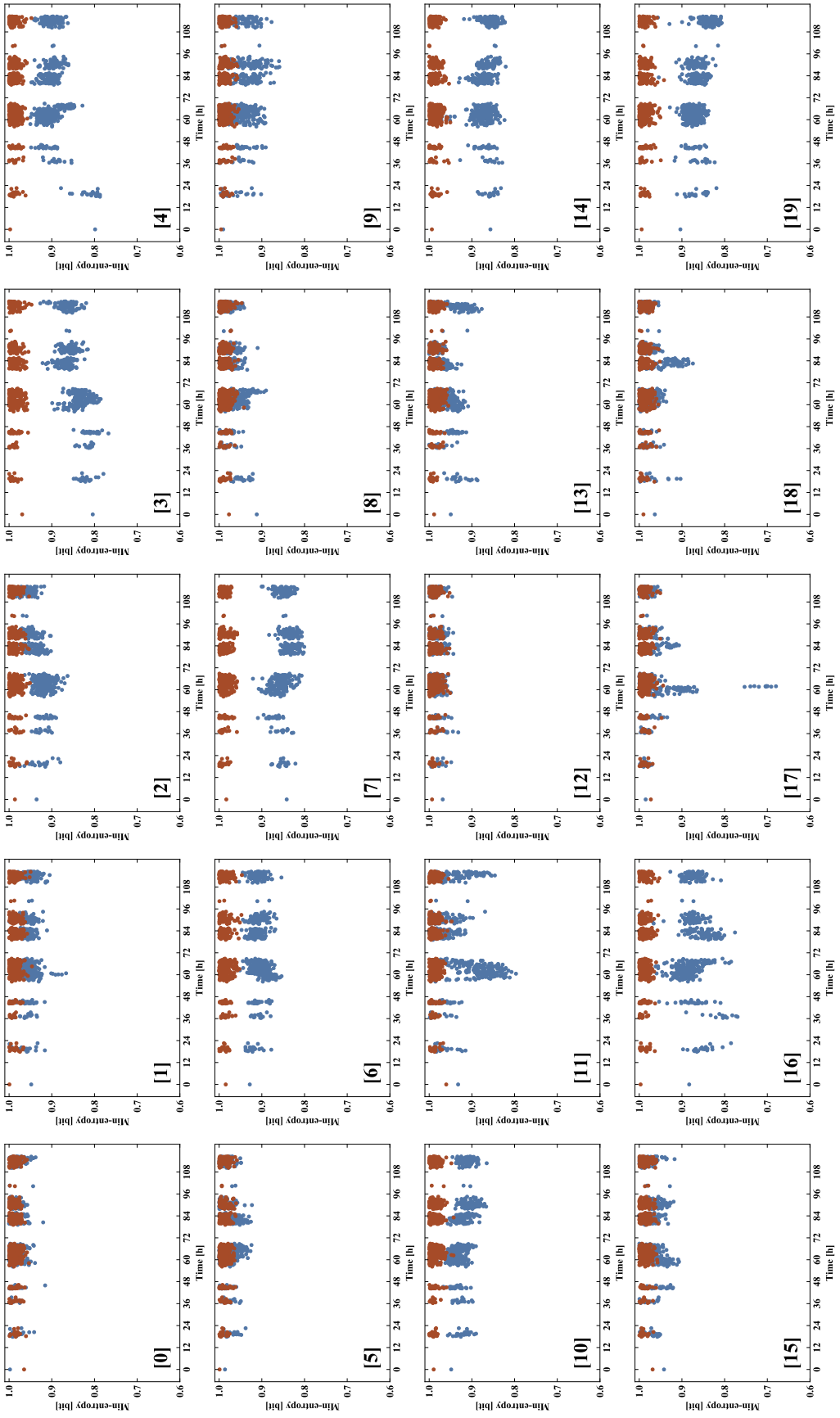


Fig. 3.14. Min-entropy transition of qubit [0]~[19]. The blue plots are the experimental results and the red plots the noisy simulation results. The figure has been rotated 90 degrees. The horizontal axis ranges from 2019/05/09 11:24 GMT to 2019/05/14 07:54 GMT (Reprinted figure from [2]).

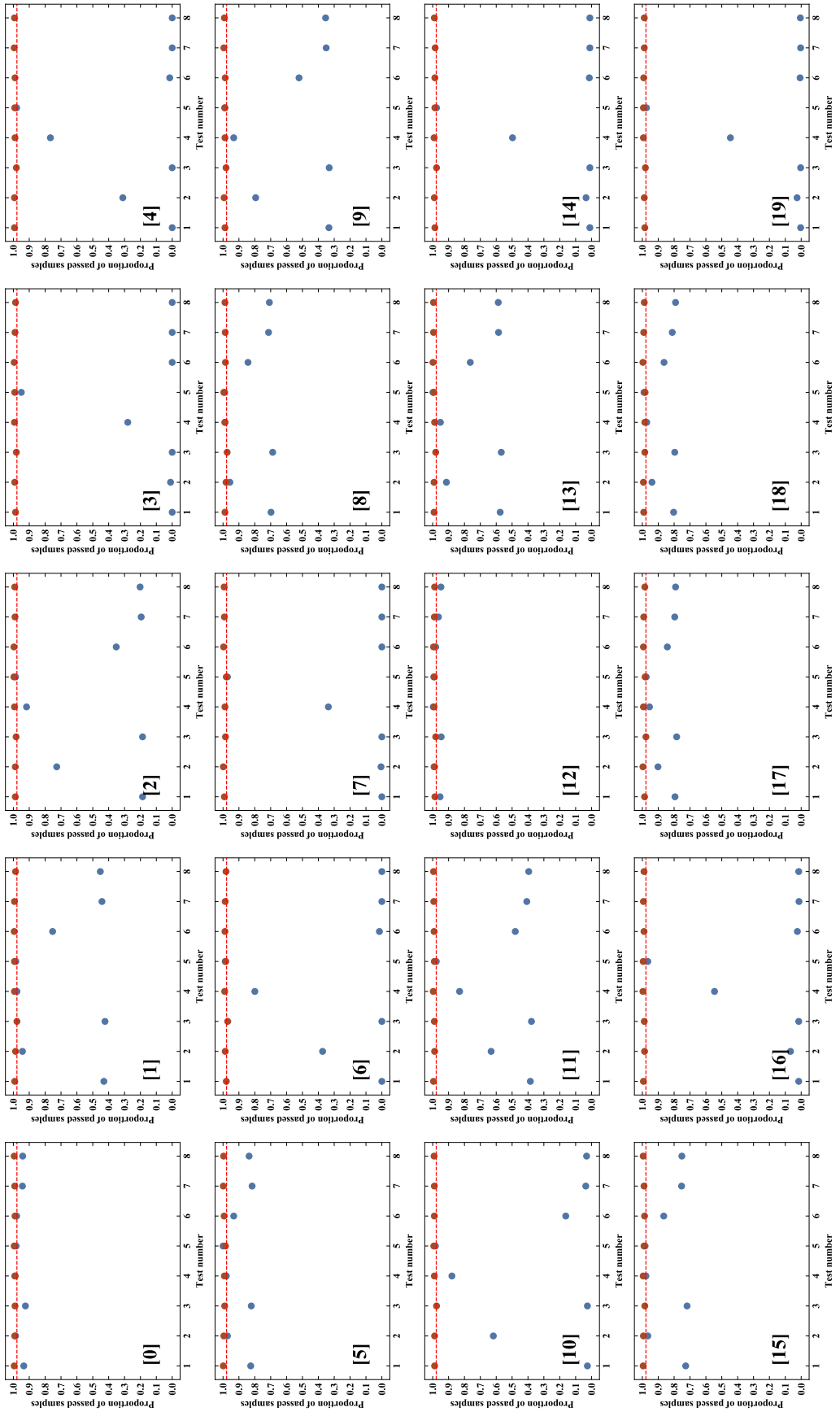


Fig. 3.15. The proportion of passed samples for each test. The test names corresponding to the test numbers can be found in Table 3.5. The acceptable range provided by the NIST is above the red line marking the proportion 0.977595. The blue plots are the experimental results and the red plots the noisy simulation results. The figure has been rotated 90 degrees (Reprinted figure from [2]).

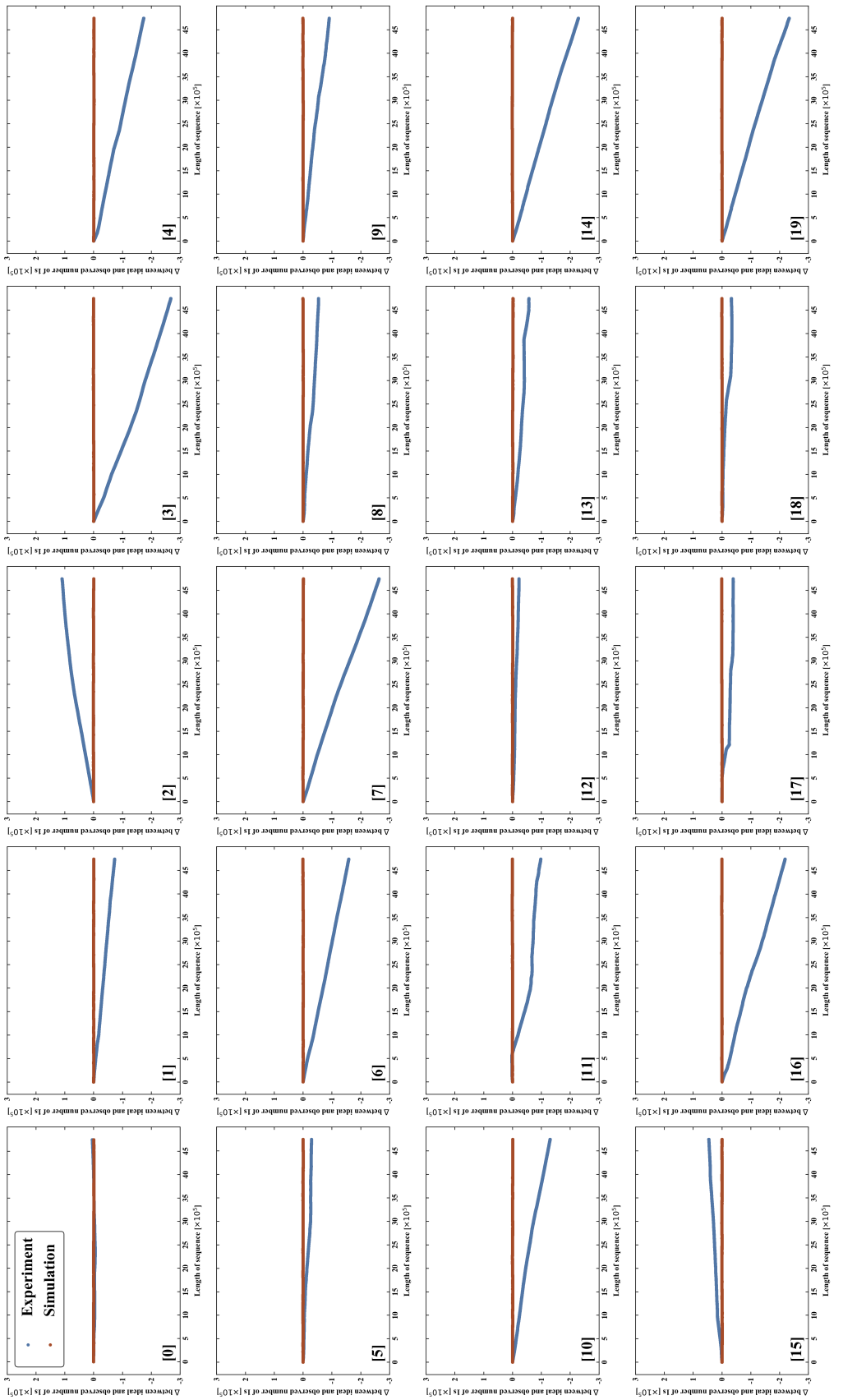


Fig. 3.16. The difference between the ideal and observed increase in the number of 1s of qubit [0]~[19]. The blue plots are the experimental results and the red plots the noisy simulation results. The figure has been rotated 90 degrees (Reprinted figure from [2]).

The results of statistical tests, min-entropy analysis, and slope analysis reveal that the random number sequences generated by real qubits are biased and that the level of bias varies over time. The random number generation output of real qubits is not identically distributed and, therefore, unstable. This fact can be leveraged to analyze the stability of qubits via quantum random number generation. However, conventional tests for random number generators cannot be used for this purpose, as they assume unbiased as well as i.i.d. generators.

3.3.4 Proposed Method

Provided that the raw output of the qubits of real quantum computing devices is non-identically distributed, we propose a method to detect the change in bias by examining the slope of the cumulative distribution of the output, thereby detecting any sign of instability. Here, the cumulative distribution of a sample is defined as the cumulative number of 1s at each point in the sample, as illustrated in Fig. 3.17.

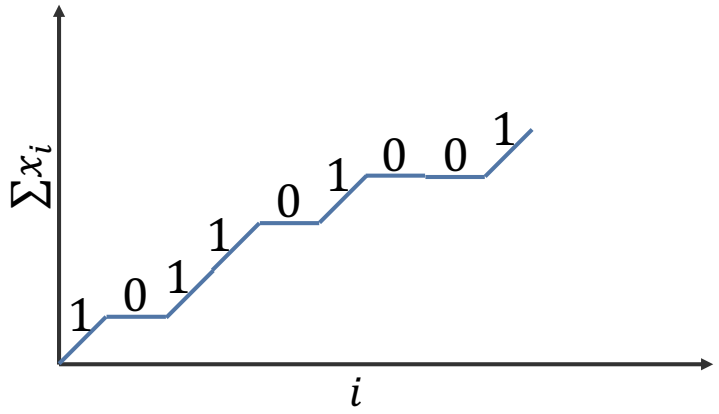


Fig. 3.17. The relationship between the random number sequence and its cumulative distribution. For the sequence $x_1, x_2, \ldots, x_n$, the cumulative distribution is $\sum_{i=1} x_i$.

The cumulative distribution can be approximated by a single line when the probability distribution of the generator is stable within the sample. In conventional tests for RNGs, the i.i.d. assumption is typically treated as a single assumption. However, the assumption in which we are interested is the identically distributed assumption. We provide a table of the correspondence between the independent and identically distributed assumption of a sample and the slopes of its approximate cumulative distribution in Table 3.6.

Table 3.6. The relationship between the i.i.d. assumption and the slope of the cumulative distributions.

Identically distributed	Independent	Cumulative Distribution
○	○	Single slope
○	×	Single slope
×	○	Multiple slopes
×	×	Multiple slopes

As illustrated in Table 3.6, non-identically distributed samples from RNGs can be observed as multiple slopes in the cumulative distribution. A simple way to check this is to use the Ramer-Douglas-Peucker (RDP) algorithm [3]. Given the distance threshold ε, the algorithm determines the lines that can explain all plots within the distance ε. The algorithm operates as follows:

1. Approximate all plots with a single line.
2. Calculate the distance of each plot from the line.

3. Compare the distance between each plot and the line with ϵ.

4. If all plots are within the distance ϵ from the line, terminate. Otherwise, find the plot that is furthest away from the line.

5. Approximate all plots with two lines, one connecting the first plot and the plot furthest away from the initial single line, and the other connecting this plot and the final plot.

6. Repeat the process by increasing the number of lines until all plots fall within the distance ϵ from the respective lines. The algorithm is shown in Fig. 3.18.

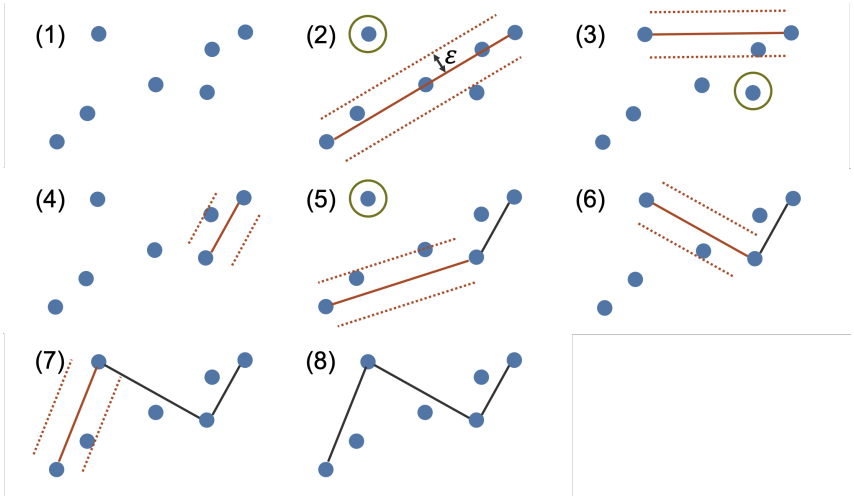


Fig. 3.18. A conceptual flow of the Ramer-Douglas-Peucker (RDP) algorithm [3].

We apply this algorithm to the cumulative distribution of the random number sequence output by each qubit of a real quantum computing device to detect whether the output is identically distributed. That is, if the cumulative distribution of the output random number sequence can be approximated by a single line using the RDP algorithm, it follows that the sequence is identically distributed.

3.3.5 Results from a Real Device

A quantum computer is a device that leverages the laws of quantum mechanics to perform computations [10]. It is crucial that a device is capable of being stable, that is, capable of consistently generating the same probability distribution over time and across qubits [39, 45–53]. One way to analyze the stability of a quantum computing device is by analyzing the distribution of the output across repeated runs of a single program. The simplest and inherently quantum mechanical task to be used to analyze the stability of quantum computing devices is the program for quantum random number generation [2, 44, 140]. We applied the proposed scheme to the qubits of a real quantum computing device to analyze their temporal stability. Another crucial trait expected of the output sequence of quantum random number generation is that its output is independent. This trait arises from the Born rule in quantum mechanics, which states that the measurement results of a quantum state are probabilistic. The independence assumption can be analyzed using the temporal correlation test [140], but only for stable samples.

To perform quantum random number generation, we employed a simple algorithm described in Fig. 3.19.

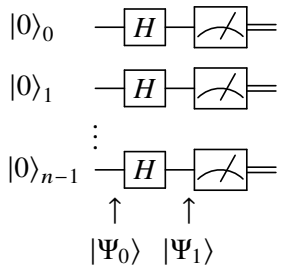


Fig. 3.19. A circuit for quantum random number generation on a quantum computing device. All n qubits are prepared at the balanced superposition state by the Hadamard gate, and are measured in the computational basis.

Each qubit is prepared in the state $|\Psi\rangle_0 = |00\cdots0\rangle$ and the Hadamard gate is applied, resulting in the balanced superposition state $[(|0\rangle + |1\rangle)/\sqrt{2}]^{\otimes n}$. All qubits are measured simultaneously along the computational basis $|0\rangle$ and $|1\rangle$, so that each qubit reads 0 or 1.

The following results were obtained from a 7-qubit device known as IBM 7Q Casablanca. We obtained 1000 samples, each containing 8192 bits from all 7 qubits of the device. For each sample, we checked the identically distributed assumption with the RDP algorithm. In this analysis, we set the threshold $\varepsilon = 65.65$. Simulation using PRNGs detected 200,474 out of 20,000,000 samples as unstable under this setting. This indicates that even with an identically distributed generator, we expect 1.00 % of samples to be detected as unstable. For 1000 samples, the 3σ range is given by $[0.056\ \%, 1.94\ \%]$. Provided in Table 3.7 is a summary of the timestamps for our experiment.

Table 3.7. The timestamps at the times the job programs were completed by IBM 7Q Casablanca.

Job complete time (UTC)	Sample number	Job ID
2020-11-26 11:37	0 – 99	5fbf6d0f779faf0013580db5
2020-11-26 11:48	100 – 199	5fbf9393030c3a0014e416fd
2020-11-26 11:57	200 – 299	5fbf9624d2c29e0013a6572e
2020-11-26 12:06	300 – 399	5fbf9836a4d96b0013783615
2020-11-26 12:15	400 – 499	5fbf9a4880afe10013a641f1
2020-11-26 12:24	500 – 599	5fbf9c59810def0012adc1f6
2020-11-26 12:33	600 – 699	5fbf9e96bb3b2b001354bb7c
2020-11-26 12:41	700 – 799	5fbfa0a2c8f281001240c230
2020-11-26 12:50	800 – 899	5fbfa28880afe10013a6426b
2020-11-26 12:58	900 – 999	5fbfa490c8f281001240c271

The results of the respective qubits are shown in Fig. 3.20~3.26. We focus on two properties of the output: bias and stability. The bias is checked by the proportion of 1s in the output. The proportion of 1s in an unbiased sample should fall within the 99 % confidence interval of $[0.486, 0.514]$. We also examine whether the proportion of samples falling outside the 99 % confidence interval is within the 3σ range given by $[0.056\ \%, 1.94\ \%]$. Stability is checked for each sample by the RDP algorithm. If the percentage of unstable samples falls outside the 3σ range of $[0.056\ \%, 1.94\ \%]$, the qubit is labeled as unstable.

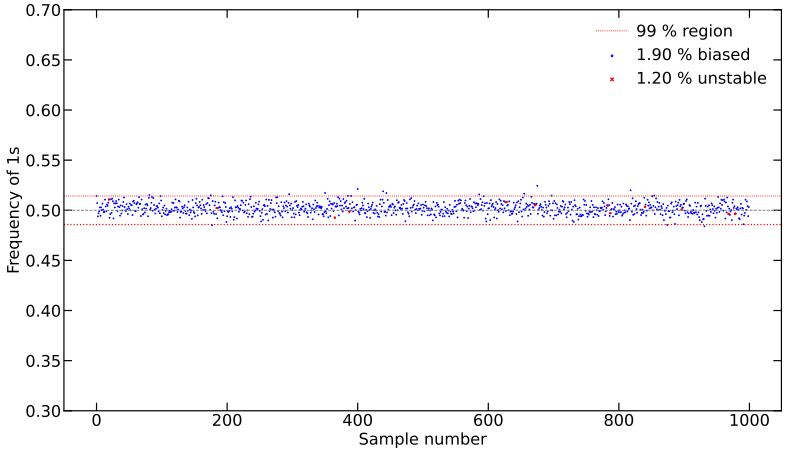


Fig. 3.20. The result of stability analysis of IBM 7Q Casablanca, qubit #0. The dotted lines represent the 99 % confidence interval under the unbiased assumption. The red plots are the unstable samples detected by the RDP algorithm.

Shown in Fig 3.20 are the results of stability analysis of qubit #0. The proportion of biased samples under the 99 % confidence interval is 1.90 %, which falls outside the 3σ range. It follows that the qubit is biased. The unstable sample rate 1.20 % is an acceptable rate and the qubit is not labeled as unstable.

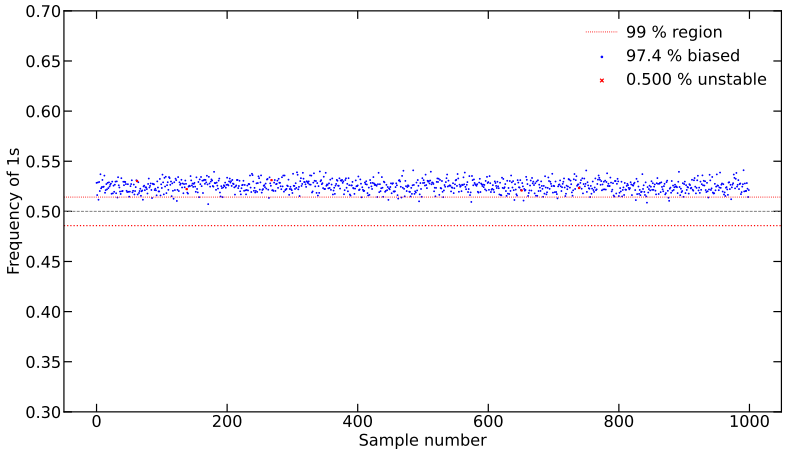


Fig. 3.21. The result of stability analysis of IBM 7Q Casablanca, qubit #1. The dotted lines represent the 99 % confidence interval under the unbiased assumption. The red plots are the unstable samples detected by the RDP algorithm.

Presented in Fig 3.21 are the results of stability analysis of qubit #1. The qubit is biased, because 97.4 % of the samples fall outside the 99 % interval. The unstable rate is 0.500 %, which does not indicate instability.

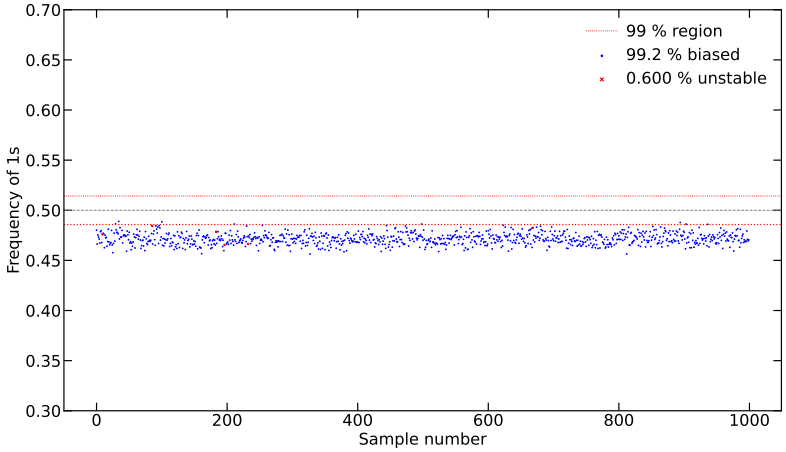


Fig. 3.22. The result of stability analysis of IBM 7Q Casablanca, qubit #2. The dotted lines represent the 99 % confidence interval under the unbiased assumption. The red plots are the unstable samples detected by the RDP algorithm.

In Fig 3.22, we present the results of stability analysis of qubit #2. The qubit is biased because it produced 99.2 % biased samples. However, the unstable sample rate does not imply instability.

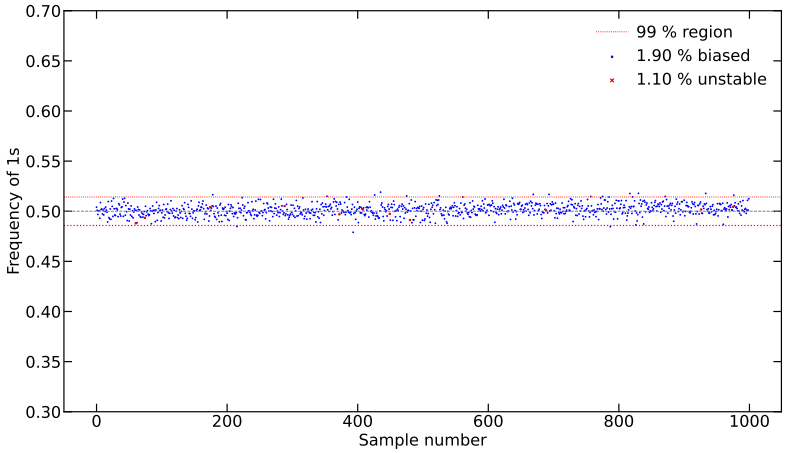


Fig. 3.23. The result of stability analysis of IBM 7Q Casablanca, qubit #3. The dotted lines represent the 99 % confidence interval under the unbiased assumption. The red plots are the unstable samples detected by the RDP algorithm.

The results of stability analysis of qubit #3 are shown in Fig 3.23. The percentage of biased samples is within the acceptable range. The unstable rate 1.10 % is also within range. The qubit is therefore unbiased and stable.

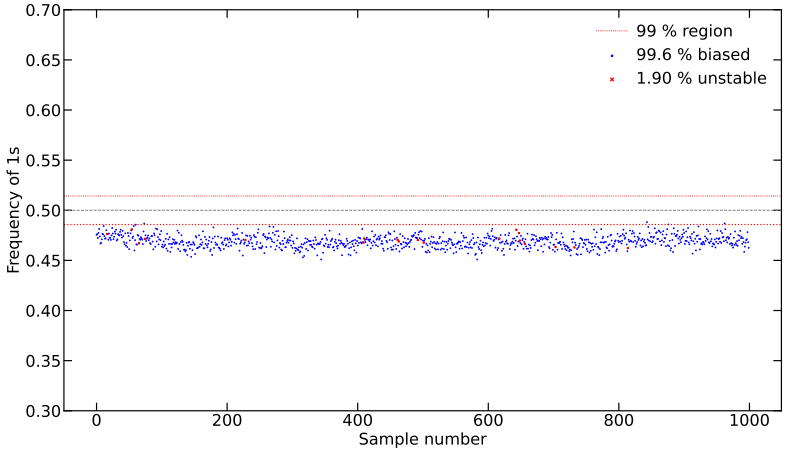


Fig. 3.24. The result of stability analysis of IBM 7Q Casablanca, qubit #4. The dotted lines represent the 99 % confidence interval under the unbiased assumption. The red plots are the unstable samples detected by the RDP algorithm.

The results of stability analysis of qubit #4 are represented in Fig 3.24. The qubit produced 99.6 % biased samples and is therefore biased. The unstable rate of 1.90 % falls outside the acceptable range, and the qubit is unstable.

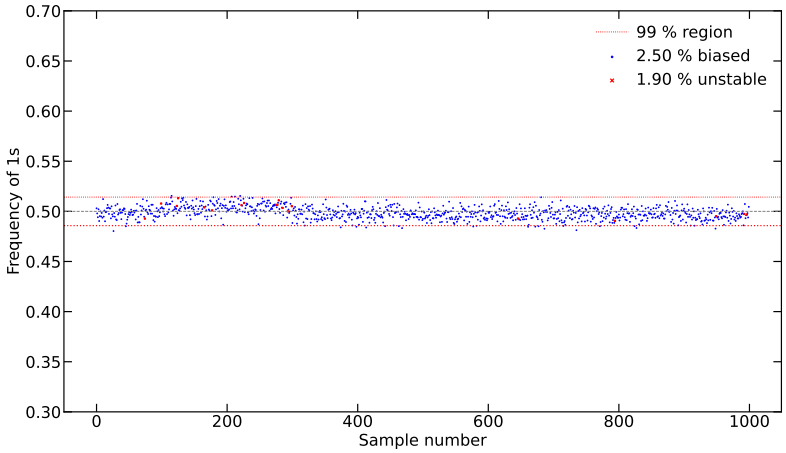


Fig. 3.25. The result of stability analysis of IBM 7Q Casablanca, qubit #5. The dotted lines represent the 99 % confidence interval under the unbiased assumption. The red plots are the unstable samples detected by the RDP algorithm.

Summarized in Fig 3.25 are the results of stability analysis of qubit #5. The qubit is biased, with a biased sample rate of 2.50 %. The unstable sample rate is 1.90 %, which suggests that the qubit is unstable. The unstable samples are clustered around sample #100 $\sim$ #300, during which time the proportion of 1s is also higher than that of the rest of the samples.

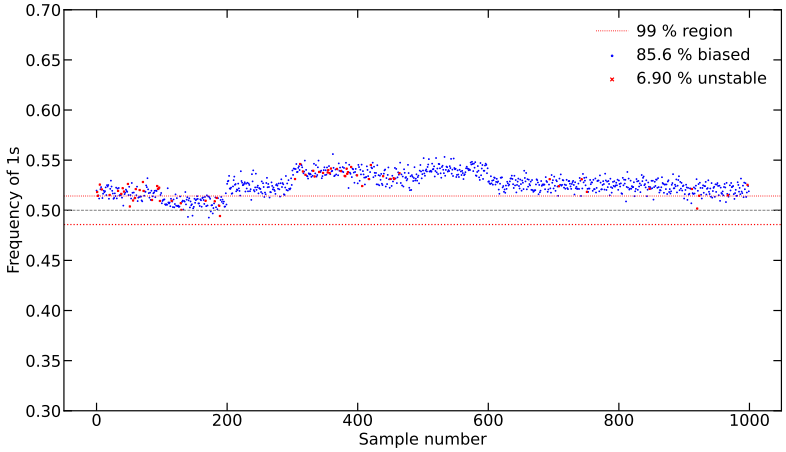


Fig. 3.26. The result of stability analysis of IBM 7Q Casablanca, qubit #6. The dotted lines represent the 99 % confidence interval under the unbiased assumption. The red plots are the unstable samples detected by the RDP algorithm.

The results of stability analysis of qubit #6 are presented in Fig 3.26. The qubit is biased and unstable. Fluctuations in the proportion of 1s can also be observed, as well as unevenly distributed unstable samples. We summarize the results in Table 3.8.

Table 3.8. The benchmarking results of IBM 7Q Casablanca.

Qubit Number	Bias Rate	Unstable Rate	Result
0	1.9 %	1.2 %	unbiased, stable
1	97.4 %	0.5 %	biased, stable
2	99.2 %	0.6 %	biased, stable
3	1.9 %	1.1 %	unbiased, stable
4	99.6 %	1.9 %	biased, stable
5	2.5 %	1.9 %	biased, stable
6	85.6 %	6.9 %	biased, unstable

Chapter 4

Noise-Robustness of Combinatorial Optimization using QRAC

Combinatorial optimization is one of the promising applications of a quantum computer. A notable quantum algorithm for combinatorial optimization is QAOA. It is an approximate algorithm for solving combinatorial optimization problems that can converge to the optimal solution in the limit of infinite circuit depth. However, real quantum computers have restricted resources. The limited number of qubits constrains the problem-size that can be solved on the device, and the effect of noise restricts the achievable circuit depth. QRAO was proposed to address the former problem. QRAO is a quantum optimization algorithm to obtain approximate solutions to combinatorial problems with fewer qubits utilizing quantum random access code (QRAC) [27]. This algorithm can reduce the number of qubits by up to one-third compared to QAOA. However, the effect of noise on the algorithm is unknown. In the present chapter, we simulate the effect of noise on QRAO, with the aim of showing how the restriction of noise affects the algorithm. We also provide a possible explanation for the impact of noise on QRAO. To do so, we first provide background on a general form of combinatorial optimization called QUBO. We then introduce quantum combinatorial optimization algorithms such as QAOA, VQE, and QRAO. Finally, we present the simulation results and analysis of QRAO under noise.

4.1 Quadratic Unconstrained Binary Optimization (QUBO)

QUBO is a form of combinatorial optimization that encompasses a wide range of important problems encountered in industry and science [141, 142]. It is the problem of maximizing or minimizing the function

$$y = m^t Q m, \tag{4.1}$$

where m is a binary vector representing the variables of the problem, and Q is a square matrix representing the constants of the problem [141]. QUBO problems are classified as NP-hard problems [143], whose practical meaning is that exact solvers designed to obtain optimal solutions are unlikely to succeed, except for very small problem instances [141, 144].

4.1.1 MaxCut Problem

The MaxCut problem is a specific type of QUBO problem that aims to partition the nodes of a graph into two subsets, maximizing the total weight of the edges that connect the nodes in different subsets [143].

The optimal combination of subset assignments $m^* \in \{0,1\}^{|V|}$ is determined as

$$m^* = \arg\max_{\{0,1\}^{|V|}}(-2m_i m_j + m_i + m_j) \tag{4.2}$$

$$= \arg\max_{\{0,1\}^{|V|}}(-2m_i m_j + m_i^2 + m_j^2), \tag{4.3}$$

where $|V|$ represents the number of nodes of the graph, and m_i indicates the subset to which the i-th node belongs, also called the node's coloring. The Q matrix for the MaxCut problem has the following properties:

- The diagonal elements $Q_{i,i}$ are set to 1.
- The off-diagonal elements $Q_{i,j}$ correspond to the weight of the edge between node i and j.

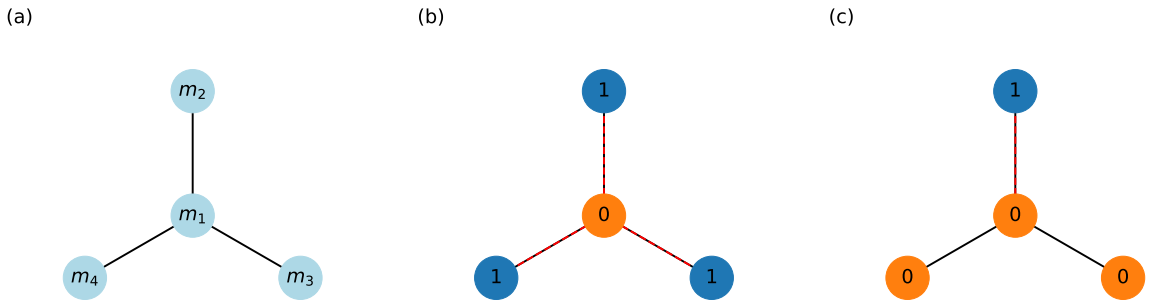


Fig. 4.1. (a) An example of a four-node graph. (b) A partitioning of the nodes of the graph that results in the maximum cut. The nodes with the classical bit 0 and 1 assigned are shown in green and red, respectively. The red edges are cut. (c) A partitioning of the nodes of the graph that does not result in the maximum cut.

For example, the graph in Fig. 4.1 (a) has the QUBO representation

$$y = [m_0, m_1, m_2, m_3] \begin{bmatrix} 1 & -2 & -2 & -2 \\ -2 & 1 & 0 & 0 \\ -2 & 0 & 1 & 0 \\ -2 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} m_0 \\ m_1 \\ m_2 \\ m_3 \end{bmatrix}. \tag{4.4}$$

The MaxCut problem has diverse applications in areas such as statistical physics, design of communications networks, and image processing [145]. Because the problem is classified as NP-hard [146], finding exact solutions for large instances is computationally infeasible [144]. Therefore, the MaxCut problem is approached by approximate algorithms, and heuristic algorithms [147]. Approximate algorithms have guarantees on how close the solution is to optimal, while heuristic algorithms have no guarantee but find a solution in a more restricted time [148]. The MaxCut value of a graph is given by

$$\text{cut}(m^*) = \frac{1}{2} \sum_{e_{i,j} \in E} w_{i,j} \left(1 - (-1)^{m_i + m_j}\right), \tag{4.5}$$

where $e_{i,j}$ represents the edge between nodes i and j in the edge set E, and $w_{i,j}$ indicates the weight of the edge $e_{i,j}$. The quality of the approximate solution is characterized by the approximation ratio, which is given by

$$\gamma = \frac{\text{cut}(m)}{\text{cut}(m^*)} \tag{4.6}$$

The approximate solution takes values ranging from 0 to 1, where 1 is optimal. The MaxCut problem is known to have the lower bound referred to as the Edwards-Erdos bound [149]. It is a lower bound on the unweighted MaxCut value of a connected graph, which are graphs in which there is a path between every pair of nodes. The Edwards-Erdos bound is given by

$$\text{cut}(m^*) \geq \frac{|E|}{2} + \frac{|V| - 1}{4}, \tag{4.7}$$

for a connected graph with $|E|$ edges and $|V|$ nodes. The Edwards-Erdos bound was then extended to general weighted graphs [150].

The simplest algorithm for the MaxCut problem is to partition the nodes into either subset uniformly at random. This achieves an expected cut of $|E|/2$. The best known polynomial-time approximate algorithm is the Goemans-Williamson algorithm [151], which has a lower bound on the expected cut value of 0.878. The Goemans-Williamson algorithm uses a technique called Semi-Definite Programming (SDP) relaxation. SDP relaxation replaces the discrete binary variables of a combinatorial problem with continuous vector-valued variables, allowing the relaxed problem to be solved in polynomial time with the desired precision [152]. The SDP relaxation of the MaxCut problem is formulated as the task of finding the vectors v_i that maximizes

$$\frac{1}{2} \sum_{e_{i,j} \in E} w_{i,j} \left(1 - v_i \cdot v_j\right), \tag{4.8}$$

where v_i belongs to the k-dimensional unit sphere S_k, where $k \geq |V|$. By solving the SDP relaxation of the problem, we obtain the vectors v_i corresponding to each node.

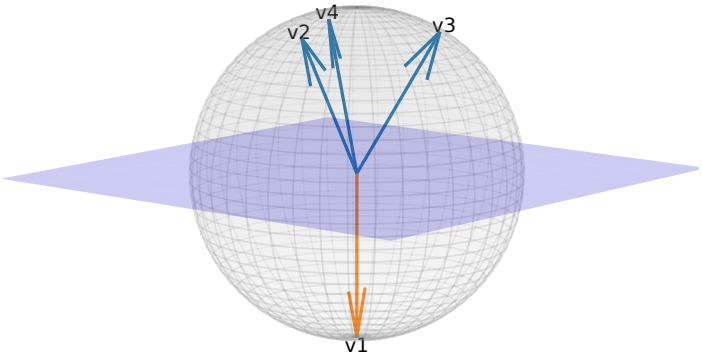


Fig. 4.2. Schematic of the Goemans-Williamson algorithm for the 4-node graph in Fig 4.1(a), where the nodes $1 \sim 4$ are mapped to unit vectors $v_i \in S_3$. A hyperplane partitions the vectors into two subsets.

The algorithm proceeds to take the inner product between a vector $r \in S_k$ generated uniformly at random and each vector v_i. The nodes are partitioned into subsets based on the sign of the corresponding inner product, as illustrated in Fig 4.2. Let us denote the approximation ratio obtained by this algorithm as α. Then, the expected approximation ratio is given by

$$\alpha = \min_{0 \leq \theta \leq \pi} \frac{2}{\pi} \cdot \frac{\theta}{1 - \cos\theta} > 0.878, \tag{4.9}$$

where $\theta \in [0, \pi]$ is the range of possible angles between the vectors v_i and v_j.

4.2 Quantum Approximate Optimization Algorithm (QAOA)

QAOA is a quantum algorithm for finding approximate solutions for combinatorial optimization problems with the combined effort of quantum computers and classical optimizers. In QAOA, we consider a problem defined on binary strings $z = \{z_1, z_2, \ldots, z_N\} \in \{-1, +1\}^N$. The objective is to maximize the cost function $C(z) = \sum_{i,j} C_{i,j}(z)$, where $C_{i,j}(z) = z_i z_j$, to maximize the approximation ratio

$$\gamma = \frac{C(z)}{C(z^*)}. \tag{4.10}$$

Here, $C(z^*)$ is the cost of the optimal solution z^*. Given such a problem, the first step of QAOA is to prepare the following two unitary gates

$$U_C = e^{-i\alpha_i H_C}, \tag{4.11}$$

$$U_X = e^{-i\beta_i H_X}, \tag{4.12}$$

where H_C and H_X are defined as

$$H_C = -\sum_{i,j} \frac{1}{2} C_{i,j}(1 - Z_i Z_j), \tag{4.13}$$

$$H_X = \sum_{j=1}^{n} X_j. \tag{4.14}$$

We then construct the QAOA circuit to obtain the state $|\alpha, \beta\rangle$. $|\alpha, \beta\rangle$ is obtained by applying the gates U_C and U_X alternately on the state $|+\rangle^{\otimes n}$ as

$$|\alpha, \beta\rangle = U_X(\beta_l) U_C(\alpha_l) \cdots U_X(\beta_1) U_C(\alpha_1) |+\rangle^{\otimes n}. \tag{4.15}$$

The measurement results of the state $|\alpha, \beta\rangle$ provide us with an estimate of the expectation value $\langle H_C \rangle$. Based on the estimate of $\langle H_C \rangle$, the parameters $\{\alpha_1, \alpha_2, \ldots, \alpha_l\}$ and $\{\beta_1, \beta_2, \ldots, \beta_l\}$ are updated using a classical optimizer to maximize $\langle H_C \rangle$. The process of QAOA is illustrated in Fig. 4.3.

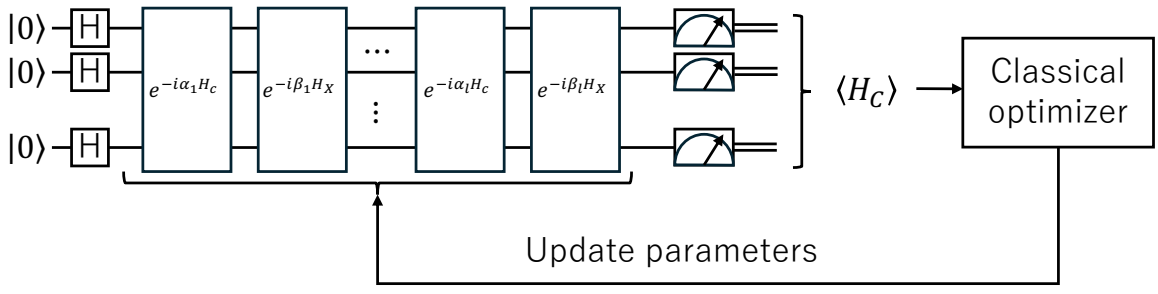


Fig. 4.3. Schematic of QAOA. The circuit prepares the n-qubit state $|+\rangle^{\otimes n}$ and alternately applies $e^{-i\alpha_i H_C}$ and $e^{-i\beta_i H_X}$ alternately. The parameters α_i and β_i are updated by a classical optimizer so that $\langle H_C \rangle$ is maximized.

When $l = 1$, the QAOA has a guaranteed lower bound on the expected approximation ratio [63, 64]. It has also been proven that as $l \to \infty$, QAOA theoretically converges to the optimal solution [63, 64]. In practice, however, increasing the value of l leads to an increase in circuit depth and circuit parameters, which are major hurdles in QAOA [153].

QAOA can be applied to the MaxCut problem by associating Z_i to node i and constructing the cost Hamiltonian H_C by setting $C_{i,j}$ as the weight of the edge between nodes i and j. For 2-regular graphs, which are graphs where every node is connected to exactly two other nodes, a general lower bound on the approximation ratio is given by [154, 155]

$$\frac{2l+1}{2l+2}. \tag{4.16}$$

For 3-regular graphs, the theoretical bounds for $l = 1, 2, 3$ are 0.6924, 0.7559, and 0.7924, respectively [63, 156, 157]. The effect of noise on QAOA has also been investigated. For example, a study [158] randomly set the parameters for a QAOA circuit and compared the state produced by the circuit in the presence and absence of noise. As a result of comparing the effect of depolarizing noise, bit-flip noise, and dephasing noise, they show that as the number of quantum gates in the QAOA circuit increases, the fidelity between the states produced by the noiseless circuit and the noisy circuit decreases exponentially with the noise parameter. Another study reports that in terms of absolute runtime, QAOA is estimated to outperform classical solvers on a scale of several hundred to a few thousand qubits for 3-regular graphs [159].

4.3 Variational Quantum Eigensolver (VQE)

The objective of VQE is to obtain an approximate solution of the ground state and the upper bound of the associated energy of a Hamiltonian [160, 161]. The method has been proposed as an alternative to algorithms such as QPE, which are difficult to implement on NISQ devices [162]. A Hamiltonian is an operator that represents the total energy of a system. A system can only be found in one of the eigenstates of the Hamiltonian, with the corresponding eigenvalue as its energy. Let us consider the characteristic equation

$$H\left|\psi_e\right\rangle = e\left|\psi_e\right\rangle, \tag{4.17}$$

where H is the Hamiltonian, e the eigenvalue, and ψ the eigenstate corresponding to the eigenvalue. The ground state corresponds to the minimum eigenvalue of the Hamiltonian. The energy expectation value of a state $|\psi\rangle$ is given by

$$E(|\psi\rangle) = \langle\psi| H |\psi\rangle. \tag{4.18}$$

According to the variational principle, an arbitrary state $|\psi\rangle$ has the relation

$$e_0 \leq E(|\psi\rangle), \tag{4.19}$$

where e_0 is the minimum eigenvalue. This indicates that regardless of the state $|\psi\rangle$, the associated energy is no smaller than the ground energy of the Hamiltonian. It also indicates that the energy associated with the state $|\psi\rangle$ provides an upper bound of the ground state of the Hamiltonian. In VQE, the arbitrary state $|\psi\rangle$ is referred to as the trial state and is updated systematically to arrive at an upper bound of the ground state that is as close to the ground state as possible.

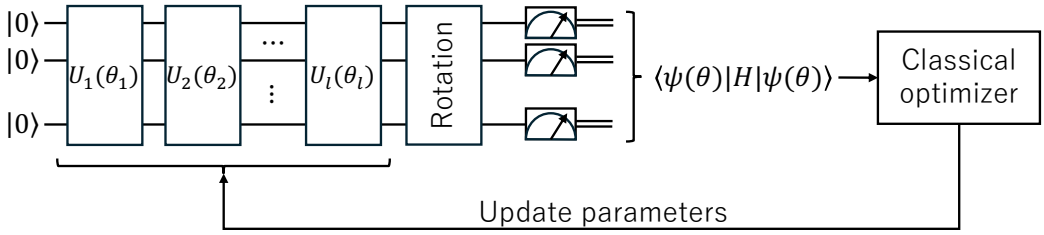


Fig. 4.4. Schematic of VQE. The ansatz $U_i(\theta_i)$ is repeated l times to create an l-layered parameterized quantum circuit. The circuit is measured after applying the appropriate rotations to the trial state, so that the energy expectation value $\langle\psi(\theta)|H|\psi(\theta)\rangle$ of the trial state is obtained. Based on the energy expectation value, a classical optimizer is employed to update the parameters of the circuit to maximize $\langle\psi(\theta)|H|\psi(\theta)\rangle$.

VQE can be summarized by the following steps:

1. Represent the Hamiltonian as a linear combination of Pauli matrices.
2. Prepare the parameterized quantum circuit.
3. Optimize the parameters of the quantum circuit using classical optimizers.

These steps are illustrated in Fig. 4.4. We explain each step in the following subsections.

4.3.1 Hamiltonian Representation

In VQE, the Hamiltonian is represented by a linear combination of Pauli matrices. This is because such a representation is convenient for quantum computers to estimate the energy expectation value. We want the Hamiltonian of the form

$$H = \sum_i a_i P_i, \tag{4.20}$$

where P_i represents the tensor product of Pauli matrices $\{I, X, Y, Z\}$ acting on different qubits, and a_i are real coefficients. The expectation value with respect to the trial state $|\psi(\theta)\rangle$ can then be decomposed as

$$\langle H\rangle = \sum_i a_i \langle\psi(\theta)|P_i|\psi(\theta)\rangle . \tag{4.21}$$

This decomposition is convenient for quantum computers, because each term $\langle\psi(\theta)|P_i|\psi(\theta)\rangle$ can be estimated by applying the appropriate rotations to the trial state and measuring in the computational basis. Of course, the Hamiltonian, whose ground state we are interested in, does not necessarily take the form in Eq. (4.20). To express a Hamiltonian in the desired form, transformations such as the Jordan-Wigner transformation [163] or the Bravyi-Kitaev transformation [164] are used.

4.3.2 Ansatz

To search for a trial state with the lowest energy on a quantum computer, a parameterized circuit called the ansatz is used. The ansatz is a series of parameterized unitary operations that can be denoted $U(\theta)$, where θ is a set of parameters that takes values in $(-\pi, \pi]$. The initial state $|\psi_0\rangle$ is subjected to the ansatz to obtain the trial state as

$$U(\theta)\,|\psi_0\rangle = |\psi(\theta)\rangle . \tag{4.22}$$

The trial state is typically $|0\rangle^{\otimes n} = |\mathbf{0}\rangle$ for an n-qubit Hamiltonian. The energy of a trial state is then obtained by

$$e = \langle \mathbf{0} | U^{\dagger}(\theta) H U(\theta) | \mathbf{0} \rangle . \tag{4.23}$$

The parameters θ are updated to minimize e. The hardware-efficient [165] ansatz is a type of ansatz that reduces the impact of noise by using native gates and connections [166], thus making it compatible with the capabilities of NISQ devices [162]. The ansatz is defined by

$$U(\theta) = \prod_{i=1}^{d} U_{\text{rotation}}(\theta_i) \times U_{\text{entanglement}}, \tag{4.24}$$

where $U_{\text{rotation}}(\theta_i)$ refers to single-qubit rotation gates and $U_{\text{entanglement}}$ to controlled-NOT gates [160].

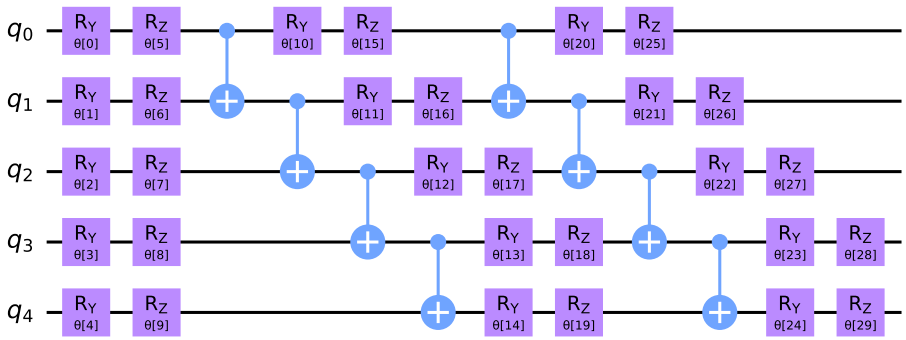


Fig. 4.5. The hardware-efficient ansatz with 5 qubits, 2 layers, and a linear entanglement style.

Fig. 4.5 is an example of a construction of the hardware-efficient ansatz [165] consisting of single rotation gates R_Y, R_Z, and linearly entangled controlled-NOT gates.

4.3.3 Optimizer

The purpose of the optimizer is to find parameters θ that achieves as low an energy expectation value as possible. It has been proven that finding the parameters that produce the ground state is NP-hard [167]. We therefore aim to approximate the best possible state by utilizing a classical optimizer. Classical optimizers can be classified into two groups: gradient-based optimizers and gradient-free optimizers.

A prime example of a gradient-based optimizer is gradient descent. In gradient descent, the expected energy value of the trial state $\langle \psi(\theta) | H | \psi(\theta) \rangle$ is updated by the rule

$$\theta_{i+1} = \theta_i - \eta \nabla_{\theta_i} \langle \psi(\theta) | H | \psi(\theta) \rangle , \tag{4.25}$$

where η is the learning rate, and ∇_θ the gradient with respect to the parameters θ. According to the parameter shift rule [168–171], the gradient is given by

$$\nabla_{\theta_i} = \frac{\langle \psi(\theta + \frac{\pi}{2}) | H | \psi(\theta + \frac{\pi}{2}) \rangle - \langle \psi(\theta - \frac{\pi}{2}) | H | \psi(\theta - \frac{\pi}{2}) \rangle}{2}. \tag{4.26}$$

The parameter shift rule [168–171] allows the estimation of the gradient by running the circuit twice per parameter. Gradient-based methods are subject to the barren-plateau phenomenon [172, 173], where the gradient of the cost function vanishes exponentially with the number of qubits [174].

An example of a gradient-free optimizer is the Nakanishi-Fuji-Todo (NFT) algorithm [175]. The NFT algorithm is recognized for its remarkable tolerance to statistical measurement noise [176]. The optimizer updates each parameter in the circuit sequentially or randomly. The algorithm updates each parameter as follows [175]:

1. Estimate $\langle\psi(\theta_0)|H|\psi(\theta_0)\rangle$ for the initial parameters θ_0.
2. Choose an index j either randomly or sequentially.
3. For the chosen parameter, estimate $\langle\psi(\theta_j+\pi/2)|H|\psi(\theta_j+\pi/2)\rangle$ and $\langle\psi(\theta_j-\pi/2)|H|\psi(\theta_j-\pi/2)\rangle$, respectively.
4. Find the value for θ_j that minimizes $\langle\psi(\theta_j)|H|\psi(\theta_j)\rangle$, utilizing the fact that $\langle\psi(\theta_j)|H|\psi(\theta_j)\rangle$ is a cosine function of θ_j.
5. Repeat steps 1 to 4 until $\langle\psi(\theta_j)|H|\psi(\theta_j)\rangle$ converges.

The insight of the NFT algorithm is the fact that the energy expectation value $\langle\psi(\theta_j)|H|\psi(\theta_j)\rangle$ for a single parameter θ_j follows the cosine curve described as

$$a_{1j}\cos(\theta_j - a_{2j}) + a_{3j}, \tag{4.27}$$

where a_{1j}, a_{2j}, a_{3j} are coefficients to be determined by evaluating the energy at the three different points.

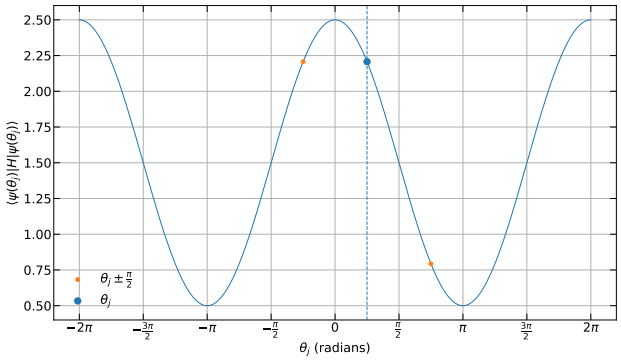


Fig. 4.6. Fitting $\langle\psi(\theta_j)|H|\psi(\theta_j)\rangle$ with the cosine curve.

Typically, the points $\theta_j - \pi/2$, θ_j, and $\theta_j + \pi/2$ are chosen to fit the cosine curve as displayed in Fig. 4.6.

4.4 Quantum Random Access Optimization (QRAO)

QRAO reduces the number of qubits required to approximately solve a combinatorial optimization problem by utilizing quantum random access code (QRAC) [27]. The algorithm is a quantum relaxation of the MaxCut problem. The MaxCut problem is relaxed as the problem of maximizing

$$\text{cut}(m) = \text{Tr}(F(m)H), \tag{4.28}$$

where $F(m)$ is a quantum state in the form of a density matrix, and H is an n-qubit Hamiltonian. Here, $F(m)$ is a quantum state associated with the binary string m of length $|V|$, hence this is a quantum relaxation of the MaxCut problem. Such a pair of F and H can be constructed with a time complexity of $O(|V|)$. QRAO constructs F and H such that F is the product state of the QRAC states of a single qubit.

Different constructions of $F(m)$ using QRAC have been proposed [177]. QRAO uses methods such as VQE to approximately obtain $F(m)$, which is then mapped to the classical solution by a measurement process referred to as quantum state rounding. In the following subsections, we outline each step involved in the QRAO process as illustrated in Fig 4.7.

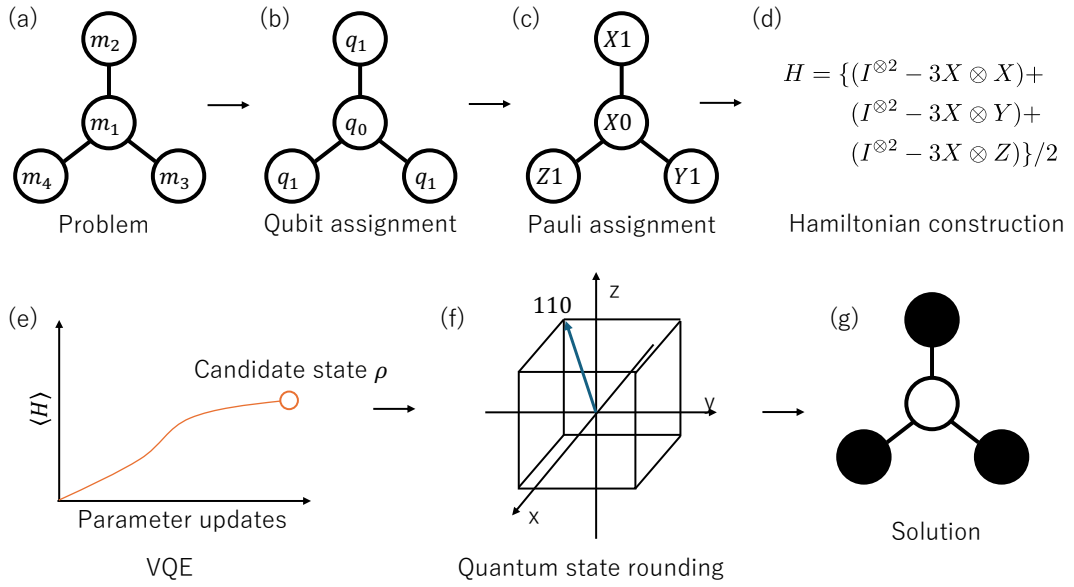


Fig. 4.7. Schematic of QRAO. (a) The classical MaxCut problem. (b) The qubit assignment of the nodes of the graph. (c) The Pauli matrix assignment of the nodes. (d) The (3, 1)-QRAC Hamiltonian construction. (e) The process of obtaining the candidate state ρ via VQE. (f) The measurement process referred to as quantum state rounding to obtain the classical solution. (g) The resulting classical solution.

In QAOA and VQE, there is no conclusive evidence that the existence of entanglement in the ansatz contributes to a higher approximation ratio in combinatorial optimization [178, 179], which suggests that these algorithms may not have reason to be run on quantum computers [177]. However, with QRAO, certain instances have been shown to yield better results with entanglement [180].

4.4.1 Quantum Random Access Code (QRAC)

The Holevo bound [58] is a fundamental limit on the amount of classical information that can be encoded and extracted from a quantum system. The theorem states that at least n qubits are required to encode and recover n classical bits with certainty. However, QRAC allows for the encoding of multiple classical bits per qubit by permitting the probabilistic recovery of the initial bits. QRAC was first proposed in the context of quantum finite automata [55, 56, 59] and quantum communication complexity [60–62]. The effect of noise in the context of communication has been investigated for a generalized version of QRAC, where 2 classical bits are encoded into a d-level quantum system referred to as a quantum digit (qudit) [181]. They simulated the effect of noise on the average success probability of QRAC and compared it to that of the classical random access code (RAC). In classical RAC, the best strategy is to encode only the first bit and leave the recovery of the second bit to chance. Their results show that the effect of noise, such as depolarizing noise, dephasing noise, and amplitude damping noise, can be reduced by optimizing QRAC states and measurement operators to maximize the average success probability [181].

The encoding of k classical bits to n quantum bits with the probability of recovering the classical bits with at least a probability of $\tilde{p}$ is denoted $(k, n, \tilde{p})$-QRAC. We introduce $(2, 1, 0.85)$-QRAC and $(3, 1, 0.78)$-QRAC.

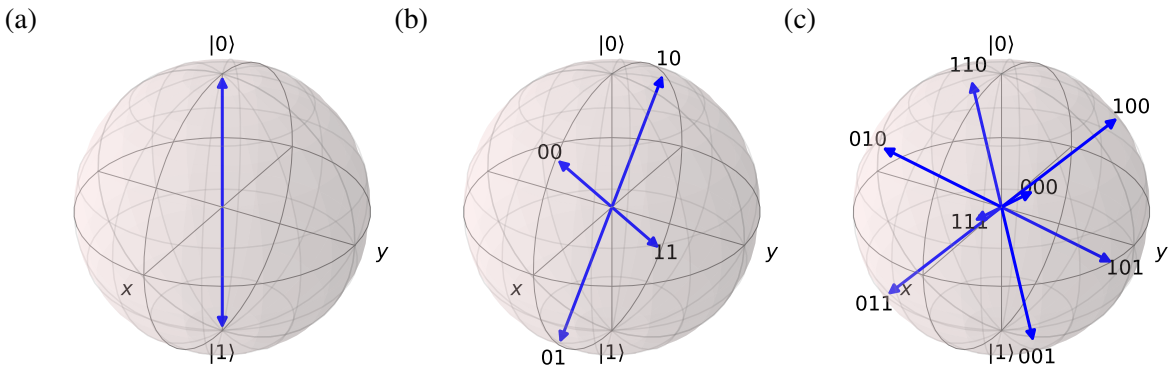


Fig. 4.8. (a) A Bloch sphere representation of the Ising encoding. (b) A Bloch sphere representation of the $(2, 1, 0.85)$-QRAC encoding. (c) A Bloch sphere representation of the $(3, 1, 0.78)$-QRAC encoding.

4.4.1.1 $(2, 1, 0.85)$-QRAC(2, 1, 0.85)-QRAC

In $(2, 1, 0.85)$-QRAC, two classical bits are encoded per qubit. The encoding rule is as follows:

$$00 \rightarrow |\psi_{00}\rangle = \cos\left(\frac{\pi}{4}\right)|0\rangle + \sin\left(\frac{\pi}{4}\right)|1\rangle, \tag{4.29}$$

$$01 \rightarrow |\psi_{01}\rangle = \cos\left(\frac{\pi}{4}\right)|0\rangle + e^{i\frac{\pi}{2}}\sin\left(\frac{\pi}{4}\right)|1\rangle, \tag{4.30}$$

$$10 \rightarrow |\psi_{10}\rangle = \cos\left(\frac{\pi}{4}\right)|0\rangle + e^{i\pi}\sin\left(\frac{\pi}{4}\right)|1\rangle, \tag{4.31}$$

$$11 \rightarrow |\psi_{11}\rangle = \cos\left(\frac{\pi}{4}\right)|0\rangle + e^{i\frac{3\pi}{2}}\sin\left(\frac{\pi}{4}\right)|1\rangle. \tag{4.32}$$

The states are represented as points evenly distributed along the circumference of the x-y plane of the Bloch sphere, as illustrated in Fig. 4.8 (b). The density matrix representation of the encoding is given by

$$f(m) = \frac{1}{2}\left(I + \frac{1}{\sqrt{2}}\left((-1)^{m_1}X + (-1)^{m_2}Z\right), \right) \tag{4.33}$$

where m_1 and m_2 are the encoded bits. To recover the bits, the state is measured in the following basis:

- m_1: measure in the X computational basis $\{|+\rangle, |-\rangle\}$;
- m_2: measure in the computational basis $\{|0\rangle, |1\rangle\}$.

The probability of successful recovery of the bit is given by the angle between the respective states and the measurement basis, which is $\pi/8$. Therefore, the success probability is given by

$$\tilde{p} = \frac{1}{2}\left(\cos\left(\frac{\pi}{8}\right) + 1\right) \approx 0.853553 > 0.85. \tag{4.34}$$

In the following, $(2, 1, 0.85)$-QRAC will be abbreviated as $(2, 1)$-QRAC.

4.4.1.2 $(3, 1, 0.78)$-QRAC(3, 1, 0.78)-QRAC

In $(3, 1, 0.85)$-QRAC, three classical bits are encoded per qubit. Three is the maximum number of classical bits that can be encoded per qubit and recovered with a probability greater than $1/2$ [182]. The intuitive reason behind this is that to construct $(4, 1, \tilde{p} > 0.5)$-QRAC requires partitioning the Bloch

sphere into 16 regions using 4 great circles, which is impossible [57]. The encoding rule is as follows:

$$000 \rightarrow |\psi_{000}\rangle = \cos\left(\frac{\pi}{6}\right)|0\rangle + \sin\left(\frac{\pi}{6}\right)|1\rangle, \tag{4.35}$$

$$001 \rightarrow |\psi_{001}\rangle = \cos\left(\frac{\pi}{6}\right)|0\rangle + e^{i\frac{\pi}{3}}\sin\left(\frac{\pi}{6}\right)|1\rangle, \tag{4.36}$$

$$010 \rightarrow |\psi_{010}\rangle = \cos\left(\frac{\pi}{6}\right)|0\rangle + e^{i\frac{2\pi}{3}}\sin\left(\frac{\pi}{6}\right)|1\rangle, \tag{4.37}$$

$$011 \rightarrow |\psi_{011}\rangle = \cos\left(\frac{\pi}{6}\right)|0\rangle + e^{i\pi}\sin\left(\frac{\pi}{6}\right)|1\rangle, \tag{4.38}$$

$$100 \rightarrow |\psi_{100}\rangle = \cos\left(\frac{\pi}{6}\right)|0\rangle + e^{i\frac{4\pi}{3}}\sin\left(\frac{\pi}{6}\right)|1\rangle, \tag{4.39}$$

$$101 \rightarrow |\psi_{101}\rangle = \cos\left(\frac{\pi}{6}\right)|0\rangle + e^{i\frac{5\pi}{3}}\sin\left(\frac{\pi}{6}\right)|1\rangle, \tag{4.40}$$

$$110 \rightarrow |\psi_{110}\rangle = \cos\left(\frac{\pi}{6}\right)|0\rangle + e^{i2\pi}\sin\left(\frac{\pi}{6}\right)|1\rangle, \tag{4.41}$$

$$111 \rightarrow |\psi_{111}\rangle = \cos\left(\frac{\pi}{6}\right)|0\rangle + e^{i\frac{7\pi}{3}}\sin\left(\frac{\pi}{6}\right)|1\rangle. \tag{4.42}$$

The states correspond to the eight vertices of the cube inscribed in the Bloch sphere, as depicted in Fig. 4.8 (c). The density matrix representation of the encoding is given by

$$f(m) = \frac{1}{2}\left(I + \frac{1}{\sqrt{3}}\left((-1)^{m_1}X + (-1)^{m_2}Y + (-1)^{m_3}Z\right)\right). \tag{4.43}$$

The recovery is done by measuring the QRAC state along the following basis:

- m_1: measure in the X basis $\{|+\rangle, |-\rangle\}$,
- m_2: measure in the Y basis $\{|+'\rangle, |-'\rangle\}$, where $|+'\rangle = \frac{1}{\sqrt{2}}(|0\rangle + i|1\rangle)$ and $|-'\rangle = \frac{1}{\sqrt{2}}(|0\rangle - i|1\rangle)$,
- m_3: measure in the Z basis $\{|0\rangle, |1\rangle\}$.

The success probability of decoding a bit is provided by

$$\tilde{p} = \frac{1}{2} + \frac{1}{2\sqrt{3}} \approx 0.788675 > 0.78. \tag{4.44}$$

In the following, $(3, 1, 0.78)$-QRAC is denoted $(3, 1)$-QRAC.

4.4.2 Encoding and Hamiltonian Construction

In this subsection, we focus on how the classical variables of the MaxCut problem can be assigned to qubits and Pauli matrices, and based on these assignments, how the Ising and (3, 1)-QRAC Hamiltonian can be constructed.

4.4.2.1 Constructing the Ising Hamiltonian

The initial step of QRAO is to assign qubits to the classical variables in the QUBO problem.

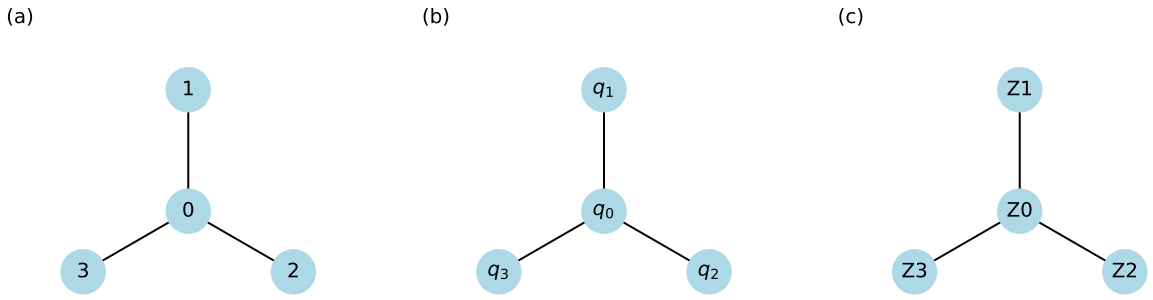


Fig. 4.9. An example of how a four-node graph is encoded for the construction of the Ising Hamiltonian. (a) An example of a four-node graph. (b) An example of qubit assignment for the construction of the Ising Hamiltonian. (c) An example of Pauli matrix assignment for the construction of the Ising Hamiltonian.

For the MaxCut problem, these classical variables correspond to the nodes of the graph. The maximum number of classical variables that can be assigned per qubit depends on the type of encoding employed. For the construction of the Ising Hamiltonian, each node is assigned to a different qubit. As a result, the same number of qubits is required as the number of nodes in the graph. For example, a four-node graph depicted in Fig. 4.9(a) can be encoded in the Ising Hamiltonian, as seen in Fig. 4.9(b). It can be observed that four qubits are required to encode a four-node graph. For each qubit, the Pauli matrix Z is then assigned. The resulting Pauli matrix assignment is illustrated in Fig. 4.9(c). Finally, the Ising Hamiltonian H_{Ising} is formulated as

$$H_{\text{Ising}} = \frac{1}{2} \sum_{e_{i,j} \in E} (I - Z_i Z_j), \tag{4.45}$$

where $e_{i,j}$ refers to the edge between i-th and j-th node, and Z_i to the Pauli matrix Z supported by i-th qubit. The resulting Ising Hamiltonian is defined with the same number of qubits as nodes and includes as many terms as there are edges in the graph.

4.4.2.2 Constructing the (3, 1)-QRAC Hamiltonian

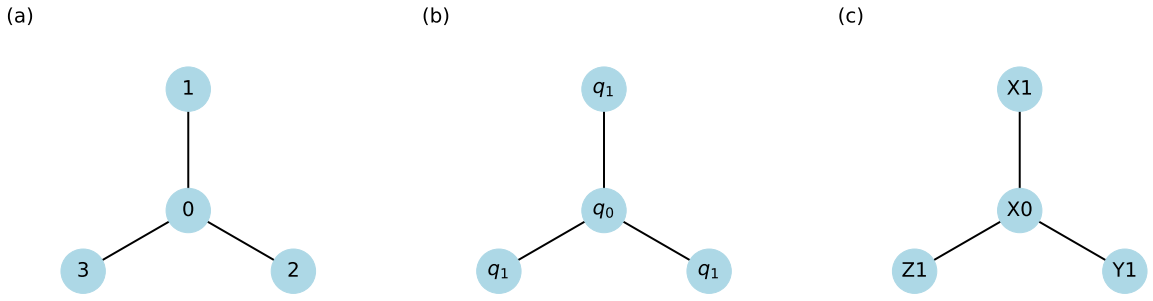


Fig. 4.10. An example of how a four-node graph is encoded for the (3, 1)-QRAC Hamiltonian. (a) An example of a four-node graph. (b) An example of qubit assignment for the (3, 1)-QRAC Hamiltonian. (c) An example of Pauli matrix assignment for the (3, 1)-QRAC Hamiltonian.

In the (3, 1)-QRAC Hamiltonian, up to three nodes can be assigned to each qubit. The number of qubits employed depends on the connectivity of the graph, because of the constraint that adjacent nodes must be assigned to different qubits. Due to this constraint, a complete graph requires as many qubits as there are nodes to encode. An example of a qubit assignment for the (3, 1)-QRAC Hamiltonian can be

seen in Fig. 4.10(b), where a four-node graph is assigned to two qubits. Each qubit is assigned one of the Pauli matricesX, Y, Z, and each matrix is assigned only once per qubit. An example of this assignment is presented in Fig. 4.10(c). Finally, the (3, 1)-QRAC Hamiltonian is formulated as

$$H = \frac{1}{2} \sum_{e_{i,j} \in E} (I - 3P_i P_j), \tag{4.46}$$

where P_i denotes the Pauli matrix assigned to the i-th node.

4.4.3 Optimization

The optimization process aims to maximize the expectation value of the Hamiltonian by altering the quantum state using variational techniques, such as the Variational Quantum Eigensolver [18] (VQE) or the Quantum Approximate Optimization Algorithm [64] (QAOA). The resulting state is then subjected to a series of measurements referred to as quantum state rounding, to obtain the classical solution. In the present study, we focus on VQE, because the QAOA approach is not commonly used in the context of QRAO. In the case of QRAO, the sign of the Hamiltonian is reversed in VQE ensuring that the upper bound of the ground energy corresponds to the lower bound of the maximum energy of the Hamiltonian.

4.4.4 Quantum State Rounding

Quantum state rounding is a measurement process for QRAO to obtain the classical bit from the quantum state (candidate state). There are two methods of rounding: Pauli rounding and magic state rounding. Although Pauli rounding has been reported to yield higher approximation ratios than magic state rounding [27], it lacks a theoretical guarantee on the expected approximation ratio, whereas magic state rounding provides such a guarantee.

Pauli rounding In Pauli rounding, the classical bit is obtained by estimating the expectation value of the Pauli matrix assigned to each node of the graph. To obtain the classical bit assigned to the i-th node from the candidate state ρ, the sign of $\mathrm{Tr}(P_i\rho)$ is estimated. If the sign is positive, the classical bit is set to 1, if the sign is negative, the classical bit is set to 0, and if the sign is 0, the classical bit is assigned uniformly at random.

Magic state rounding In magic state rounding, one of the four bases $\{\mu_1, \mu_2, \mu_3, \mu_4\}$ is chosen for each qubit, and the qubit is measured once along the selected basis. The four bases correspond to the four space diagonals of the cube in the Bloch sphere representation of the (3, 1)-QRAC encoding. The four bases are:

$$\begin{aligned}
\mu_1^{\pm} &= \frac{1}{2}\left(I \pm \frac{1}{\sqrt{3}}(X + Y + Z)\right), \\
\mu_2^{\pm} &= \frac{1}{2}\left(I \pm \frac{1}{\sqrt{3}}(X - Y - Z)\right), \\
\mu_3^{\pm} &= \frac{1}{2}\left(I \pm \frac{1}{\sqrt{3}}(-X + Y - Z)\right), \\
\mu_4^{\pm} &= \frac{1}{2}\left(I \pm \frac{1}{\sqrt{3}}(-X - Y + Z)\right),
\end{aligned}$$

as illustrated in Fig. 4.11.

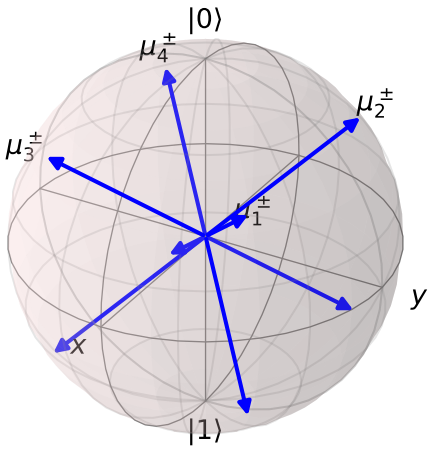


Fig. 4.11. The relationship between the four bases of magic state rounding and the four space diagonals of the cube in the Bloch sphere representation of the (3, 1)-QRAC encoding.

One of the advantages of magic state rounding is that it has a lower bound on the expected approximation ratio as

$$\mathbb{E}(\gamma) = \mathbb{E}\left[\frac{\text{cut}(m)}{\text{cut}(m^*)}\right] = \frac{\mathbb{E}\left[\text{Tr}\left(\mathcal{M}^{\otimes n}(\rho)H\right)\right]}{\text{Tr}\left(F(m^*)H\right)} \geq \frac{5}{9}. \tag{4.47}$$

Here, ρ represents the candidate state subjected to rounding, $\text{cut}(m^*)$ the maximum cut value, m^* the optimal configuration, $F(m^*)$ the map that satisfies $\text{Tr}(F(m^*)H) = \text{cut}(m^*)$. Note that each iteration of magic state rounding entails measuring every qubit on a basis randomly chosen from the set $\{\mu_i^+, \mu_i^-\}$ for $i \in [4]$. The expected cut value of magic state rounding is denoted $\mathbb{E}\left[\text{Tr}\left(\mathcal{M}^{\otimes n}(\rho)H\right)\right]$. This lower bound is derived under the assumption that the candidate state ρ has an expected energy at least as high as the maximum cut $\text{cut}(m^*)$. A subsequent study [183] found that the expected approximation ratio of magic state rounding is

$$\text{Tr}(H\rho) = 9\mathbb{E}\left(\text{Tr}\left(\mathcal{M}^{\otimes n}(\rho)H\right)\right) - 4|E|. \tag{4.48}$$

4.5 Results of Noise-Robustness from Simulation

In this section, we present the simulation results of QRAO. We solve the MaxCut problem for 50 random 3-regular graphs for the numbers of nodes 8, 12, 16, 20, 24, and 28 using the noiseless simulator and noisy simulator. We compare the approximation ratios obtained by the Ising Hamiltonian and the (3, 1)-QRAC Hamiltonian to see the effect of qubit reduction and rounding on the classical solution. We also analyze the noise-robustness of QRAO under depolarizing noise and show the number of measurements required to estimate the trace values corresponding to the nodes.

4.5.1 Simulation Setup and Parameters

The simulations were conducted with the following simulators:

- Noiseless simulator (Statevector simulator),
- Fake noise simulator (Statevector simulator with fake noise),
- Depolarizing noise simulator (Statevector simulator with depolarizing noise, whose error probability is 1% on all the controlled-NOT gates).

To estimate the energy expectation value in VQE, we performed 1024 measurements on each Pauli term. Fake noise is a noise model provided by IBM that is designed to replicate the behavior of a real noisy device. The noise model combines the single-qubit depolarizing error, single-qubit thermal relaxation error, two-qubit depolarizing error, and single-qubit readout error, with parameters tuned based on real system snapshots [4]. The candidate states of the Ising and (3, 1)-QRAC Hamiltonian were obtained by VQE, where 3 layered hardware efficient ansatz was employed. As for the classical optimizer in VQE, we chose the NFT algorithm, performing 2 parameter updates for each parameter in the VQE circuit. We used the linear entanglement style, where the controlled-NOT gate depth is $n-1$ for n qubits per layer. For the Ising Hamiltonian, noisy simulations for graphs with 20 or more nodes could not be completed within a reasonable time frame because the noisy statevector simulator uses extensive sampling to simulate the effect of noise. An alternative to the noisy statevector simulator is the noisy density matrix simulator, which does not require sampling. However, it uses $2^n \times 2^n$ matrices for n-qubit simulations, making it resource consuming.

We computed the 95 % confidence intervals for the mean approximation ratios of each Hamiltonian to compare the effect of noise on the classical solution. We defined the mean approximation ratio $\overline{\gamma}_{|V|}$ for 50 $|V|$-node graphs for the (3, 1)-QRAC Hamiltonian (magic state rounding) and the Ising Hamiltonian by

$$\overline{\gamma}_{|V|} = \frac{1}{25000} \sum_{j=1}^{50} \sum_{k=1}^{500} (\gamma_{|V|_j})_k. \tag{4.49}$$

Here, $|V|_j$ represents the number of nodes in the j-th graph, and k denotes the k-th measurement result. Note that $25,000$ is obtained by multiplying the number of graphs for each number of nodes 50 and the number of measurements for each graph 500. The mean approximation ratio for 50 $|V|$-node graphs of the QRAC Hamiltonian (Pauli rounding) is given by

$$\overline{\gamma}_{|V|} = \frac{1}{50} \sum_{j=1}^{50} \gamma_{|V|_j}. \tag{4.50}$$

We define the 95% confidence intervals of the mean approximation ratios by

$$\left(\overline{\gamma}_{|V|} - 1.96 \frac{\sigma_{\overline{\gamma}_{|V|}}}{\sqrt{25000}}, \overline{\gamma}_{|V|} + 1.96 \frac{\sigma_{\overline{\gamma}_{|V|}}}{\sqrt{25000}} \right) \tag{4.51}$$

for the Ising Hamiltonian and QRAC Hamiltonian via magic state rounding, where

$$\sigma_{\overline{\gamma}_{|V|}} = \sqrt{\frac{1}{25000} \sum_{j=1}^{50} \sum_{k=1}^{500} \left((\gamma_{|V|_j})_k - \overline{\gamma}_{|V|} \right)^2}, \tag{4.52}$$

and

$$\left(\overline{\gamma}_{|V|} - 1.96 \frac{\sigma_{\overline{\gamma}_{|V|}}}{\sqrt{50}}, \overline{\gamma}_{|V|} + 1.96 \frac{\sigma_{\overline{\gamma}_{|V|}}}{\sqrt{50}} \right) \tag{4.53}$$

for the QRAC Hamiltonian via Pauli rounding, where

$$\sigma_{\overline{\gamma}_{|V|}} = \sqrt{\frac{1}{50} \sum_{j=1}^{50} \left(\gamma_{|V|_j} - \overline{\gamma}_{|V|} \right)^2}. \tag{4.54}$$

The VQE energy ratio is the ratio between the energy expectation value of the candidate state and the maximum eigenvalue of the Hamiltonian. The VQE energy ratio for 50 $|V|$-node 3-regular graphs is defined by

$$\overline{E}(|V|) = \frac{1}{50}\sum_{j=1}^{50}\tilde{E}_j, \tag{4.55}$$

where

$$\tilde{E}_j := \frac{E_j - E_{j,\max}}{E_{j,\max} - E_{j,\min}}. \tag{4.56}$$

The 95% confidence intervals are defined by

$$\left(\overline{E}(|V|) - \frac{1.96}{\sqrt{50}}\sqrt{\sum_{j=1}^{50}\left(\tilde{E}_j - \overline{E}(|V|)\right)^2}, \overline{E}(|V|) + \frac{1.96}{\sqrt{50}}\sqrt{\sum_{j=1}^{50}\left(\tilde{E}_j - \overline{E}(|V|)\right)^2}\right), \tag{4.57}$$

where E_j is the VQE energy expectation value for the j-th graph, and $(E_{j,\min}, E_{j,\max})$ are the minimum and maximum eigenvalues corresponding to the j-th graph. The VQE energy E_j for the QRAC Hamiltonian is rescaled as $(E_j - E_{j,\min})/(E_{j,\max} - E_{j,\min})$.

4.5.2 Simulation Results

The result of the noiseless simulation is shown in Fig. 4.12 (a). We first compare the Pauli rounding results for the (3, 1)-QRAC Hamiltonian plotted in blue with those of the Ising Hamiltonian plotted in black in Fig. 4.12 (a). We find that the mean approximation ratio of the Ising Hamiltonian decreases as the number of nodes increases. Taking into account the error bars representing the 95 % confidence intervals into account, the (3, 1)-QRAC Hamiltonian shows a higher mean approximation ratio for graphs with 20 nodes or more. For the result of the magic state rounding of the (3, 1)-QRAC Hamiltonian plotted in orange, the mean approximation ratio decreases as the number of nodes increases, but the ratio remains higher than the lower bound of $5/9 \approx 0.56$.

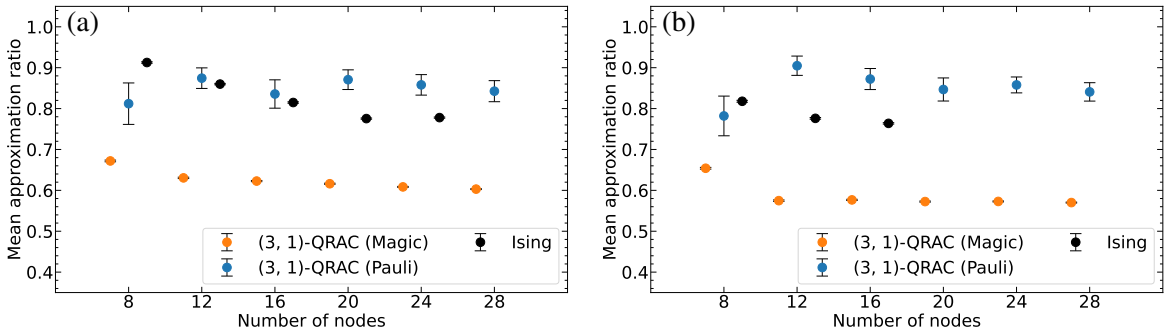


Fig. 4.12. The mean approximation ratio and 95 % confidence interval of the MaxCut problem, where the results from the Ising Hamiltonian are shown in black, and the results from the (3, 1)-QRAC Hamiltonian are shown in blue for Pauli rounding and orange for magic state rounding. Each number of nodes on the x-axis represents three plots located on the left, center, and right. For each number of nodes, 50 random 3-regular graphs were solved. (a) Results from the statevector simulator. (b) Results from the fake backend FakeMumbaiV2 [4] (Fig. 4 from [5] by Tamura is licensed under https://creativecommons.org/licenses/by/4.0/CC BY 4.0).

Next, we compare the noiseless result in Fig. 4.12 (a) to the result under fake noise in Fig. 4.12 (b). By comparing the mean approximation ratio of the (3, 1)-QRAC Hamiltonian via Pauli rounding between

the two figures, we observe that for the same number of nodes, the fluctuations remain within the range of the 95 % confidence intervals. This indicates that the mean approximation ratio of the (3, 1)-QRAC Hamiltonian is robust against noise. The mean approximation ratio of the Ising Hamiltonian, on the other hand, shows a decline for the same number of nodes outside the range of the 95 % confidence intervals.

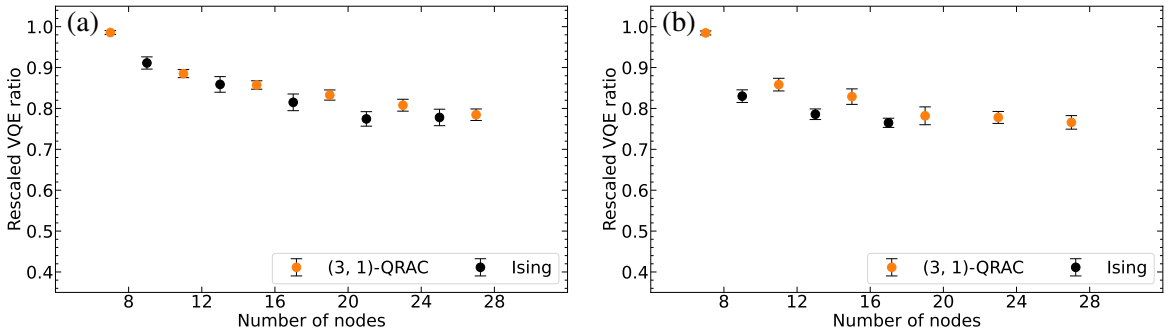


Fig. 4.13. The rescaled VQE energy ratio of the candidate states used to obtain Fig. 4.12, where the results from the Ising Hamiltonian are shown in black, and the results from the (3, 1)-QRAC Hamiltonian are shown in orange. Each number of nodes on the x-axis represents two plots located on the left and right. For each number of nodes, 50 random 3-regular graphs were solved. (a) Results from the statevector simulator. (b) Results from the fake backend FakeMumbaiV2 [4] (Fig. 4 from [5] by Tamura is licensed under https://creativecommons.org/licenses/by/4.0/CC BY 4.0).

The rescaled VQE energy ratios of the candidate states used for quantum state rounding are shown in Fig. 4.13 (a) for the noiseless simulation and Fig. 4.13 (b) for the fake noise simulation. In both Fig. 4.13 (a) and (b), we observe that the VQE energy ratio declines as the number of nodes increases. This can be explained by the fact that larger graphs require more qubits, resulting in an increase in circuit depth. An increase in circuit depth leads to an increase in the amount of accumulated noise, resulting in a decline in the VQE energy ratio. Upon comparing (a) and (b), we observe that with noise, both Hamiltonian achieve a lower VQE ratio with the same problem-size.

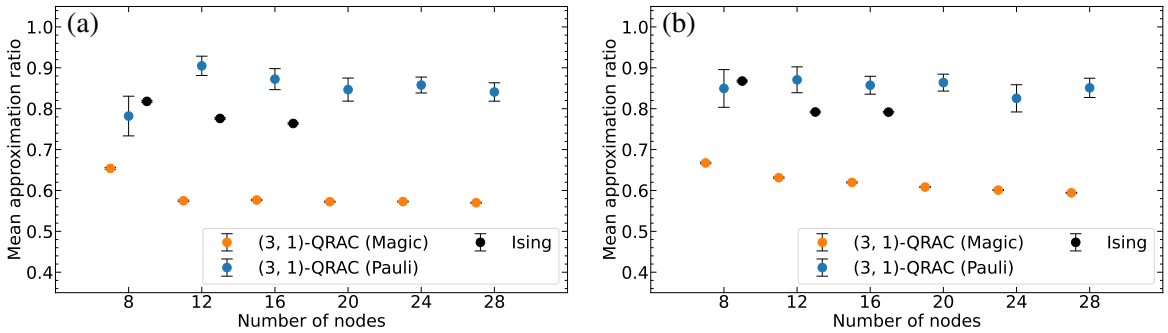


Fig. 4.14. The mean approximation ratio and 95 % confidence interval of the MaxCut problem, where the results from the Ising Hamiltonian are shown in black, and the results from the (3, 1)-QRAC Hamiltonian are shown in blue for Pauli rounding and orange for magic state rounding. Each number of nodes on the x-axis represents three plots located on the left, center, and right. For each number of nodes, 50 random 3-regular graphs were solved. (a) Results from the fake backend FakeMumbaiV2 [4]. (b) Results from the statevector simulator where depolarizing noise is applied on the controlled-NOT gates with an error parameter of 1 % (Fig. 4 from [5] by Tamura is licensed under https://creativecommons.org/licenses/by/4.0/CC BY 4.0).

In Fig. 4.14, we compare (a) the results under fake noise with (b) the results under depolarizing noise. We observe that the mean approximation ratio of the (3, 1)-QRAC Hamiltonian via Pauli rounding shows robustness under depolarizing noise as well.

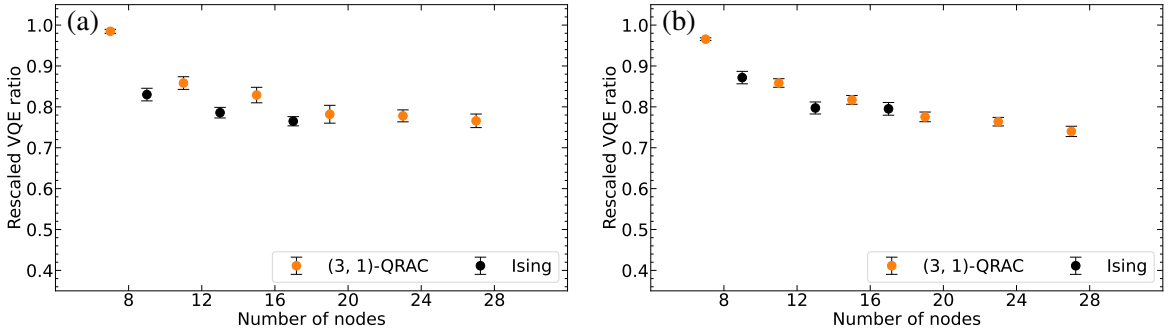


Fig. 4.15. The rescaled VQE energy ratio of the candidate states used to obtain Fig. 4.12, where the results from the Ising Hamiltonian are shown in black, and the results from the (3, 1)-QRAC Hamiltonian are shown in orange. Each number of nodes on the x-axis represents two plots located on the left and right. For each number of nodes, 50 random 3-regular graphs were solved. (a) Results from the fake backend FakeMumbaiV2 [4]. (b) Results from the statevector simulator where depolarizing noise is applied on the controlled-NOT gates with an error parameter of 1 % (Fig. 4 from [5] by Tamura is licensed under https://creativecommons.org/licenses/by/4.0/CC BY 4.0).

In Fig. 4.15, we display the rescaled VQE energy ratios under (a) fake noise and (b) depolarizing noise. We observe how the effect of depolarizing noise on the rescaled VQE energy ratios is consistent with the effect of fake noise. In the next subsection, we discuss the noise-robustness of the (3, 1)-QRAC Hamiltonian under depolarizing noise.

4.5.3 Effect of Noise on QRAO

The robustness of the (3, 1)-QRAC Hamiltonian via Pauli rounding can be explained by the following relation:

$$\begin{aligned}\mathrm{Tr}(P_j \mathcal{D}_p^N(\rho)) &= \mathrm{Tr}\left[P_j\left(p^N\rho + (1-p^N)\frac{I}{2^n}\right)\right] \\ &= p^N \mathrm{Tr}[P_j\rho] + (1-p^N)\mathrm{Tr}\left[P_j\frac{I}{2^n}\right] \\ &= p^N \mathrm{Tr}[P_j\rho], \end{aligned} \tag{4.58}$$

where $(1-p)$ represents the error probability of depolarizing noise, P_j the Pauli matrix corresponding to the j-th node, and N the number of applications of depolarizing noise on the n-qubit candidate state ρ. The relation indicates that the repeated application of depolarizing noise causes the absolute trace value corresponding to each node of the graph to decrease but does not affect the sign of the trace value. Because the sign of the trace value corresponding to each node determines the classical solution in Pauli rounding, this relationship indicates that depolarizing noise does not influence the classical solution of Pauli rounding, but rather affects only the number of shots needed to estimate the classical solution. the minimum number of shots S required to estimate the sign of the trace value assigned to each node with error probability at most δ is given by

$$S \geq \frac{\ln(1/\delta)}{2\varepsilon^2}, \tag{4.59}$$

and the order of shots is

$$O\left(\frac{\ln(|V|)}{\varepsilon^2}\right). \tag{4.60}$$

Here, V is the number of nodes in the graph, and $\varepsilon > 0$ is defined by $\Pr(X_{ij} = 1) = 1/2 + \varepsilon$. X_{ij} is the result of measurement of the i-th measurement and the Pauli matrix P_j assigned to the j-th node. To

estimate the sign of the trace values of all $|V|$ nodes, the order is

$$O\left(\frac{|V|\ln(|V|)}{\varepsilon^2}\right). \tag{4.61}$$

Note that $\varepsilon = -p^N \mathrm{Tr}(P_j\rho)/2$ under depolarizing noise. The minimum number of shots given by

$$S \geq \frac{2\ln(1/\delta)}{p^{2N}\mathrm{Tr}(P_j\rho)^2} \tag{4.62}$$

scales inversely with the depolarizing noise success parameter p and exponentially with the number of noise applications N. The inequality also applies to the Ising Hamiltonian. Assuming that the (3,1)-QRAC Hamiltonian uses exactly $\frac{1}{3}$ of the qubits required by the Ising Hamiltonian, and the number of noise applications for the Ising Hamiltonian is $N = |V|l$, where l denotes the number of ansatz layers, the ratio of the required shots for the (3,1)-QRAC Hamiltonian to those for the Ising Hamiltonian is given by $p^{\frac{4}{3}l|V|}$. This indicates that under the influence of noise, the (3, 1)-QRAC Hamiltonian requires fewer shots to achieve the same precision of estimating the classical solution.

For the (3, 1)-QRAC Hamiltonian via magic state rounding, there is a guaranteed lower bound of 5/9 for the expected approximation ratio given that the energy expectation value of the candidate state is no lower than the maximum cut value. For the Ising Hamiltonian, the energy expectation value of the candidate state directly represents the expected approximation ratio. Let us denote the number of noise applications for the (3, 1)-QRAC Hamiltonian N_3 and for the Ising Hamiltonian N_1. We consider a scenario where the maximum eigenstate repeatedly undergoes depolarizing noise with error probability $(1-p)$. The resulting state is then subjected to magic state rounding, which corresponds to measurement in the computational basis for the Ising Hamiltonian. The expected approximation ratio under noise is given by

$$\mathbb{E}(\gamma_3') \geq \frac{5}{9}p^{N_3} + (1-p^{N_3})\frac{|E|}{2\mathrm{cut}(m^*)}. \tag{4.63}$$

for the (3, 1)-QRAC Hamiltonian and

$$\mathbb{E}(\gamma_1') = p^{N_1} + (1-p^{N_1})\frac{|E|}{2\mathrm{cut}(m^*)}, \tag{4.64}$$

for the Ising Hamiltonian. Assuming that $p \in (0,1)$ and $N_1 > N_3$, which implies $p^{N_1} > p^{N_3}$, the condition for inequality $\mathbb{E}(\gamma_3') \geq \mathbb{E}(\gamma_1')$ to hold is $\mathrm{cut}(m^*)/|E| \geq 9/10$. Note that $\mathbb{E}(\gamma_3')$ is the lower bound of the approximation ratio, while $\mathbb{E}(\gamma_1')$ is the expectation value.

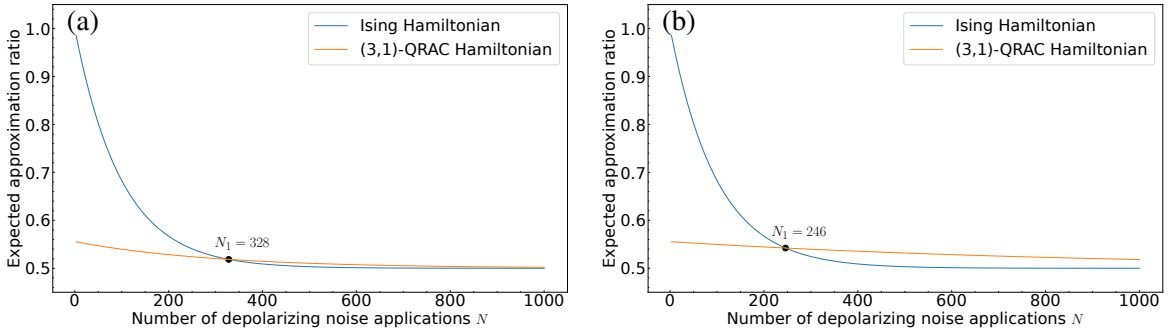


Fig. 4.16. Analytic curves of the expected approximation ratio of the Ising Hamiltonian and the lower bound on the expected approximation ratio of the (3, 1)-QRAC Hamiltonian via magic state rounding. We assumed $\mathrm{cut}(m^*)/|E| = 1$ and $p = 0.99$. (a) Linear entanglement style, where $N_3 = N_1/3$. (b) Full entanglement style, where $N_3 = N_1/9$.

The analytic curves of the expected approximation ratio of the Ising Hamiltonian and the lower bound of the expected approximation ratio of the (3, 1)-QRAC Hamiltonian are presented in Fig. 4.16. We find that the intersection point of the two curves changes to a smaller N_1 as the noise parameter $(1 - p)$ increases or the entanglement style changes from (a) linear to (b) full. Note that in this analysis, the difficulty of obtaining the maximum eigenstates for the respective Hamiltonian is not taken into consideration.

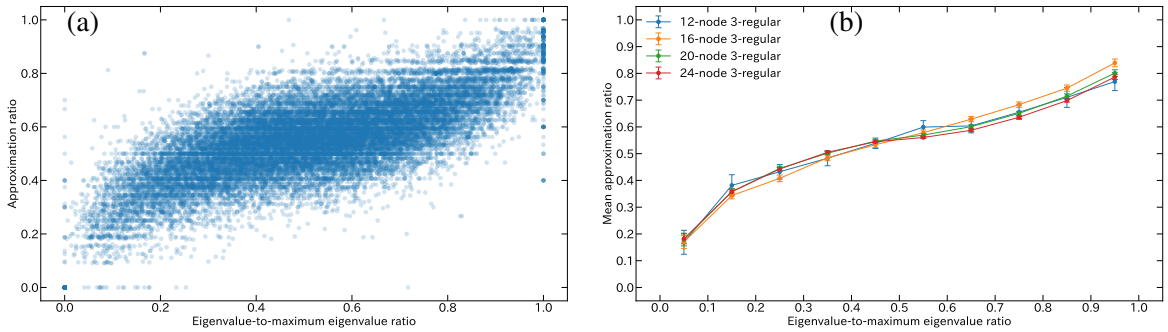


Fig. 4.17. The relationship between the approximation ratio obtained via Pauli rounding and the eigenvalue-to-maximum eigenvalue ratio of the (3, 1)-QRAC Hamiltonian for 50 random 3-regular graphs with with 12, 16, 20, and 24 nodes. (a) Scatter plot of the approximation ratios associated with the respective ratios. (b) Histogram of the mean approximation ratio and the 95% confidence intervals for the respective ratios.

The simulated relationship between the energy expectation value achieved and the approximation ratio of the (3, 1)-QRAC Hamiltonian is shown in Fig. 4.17. We observe from Fig. 4.17 (b) that on average, Pauli rounding yields a higher approximation ratio as the energy expectation value increases. However, the scatter plot shown in Fig. 4.17 indicates that even with the maximum eigenstate, there are cases where the approximation ratio is lower than 0.5. These cases occur when the trace value of the Pauli matrices assigned to the respective nodes equals zero, resulting in a randomly assigned classical solution. For example, consider the graph in Fig. 4.18. The trace values of the nodes with regard to the maximum eigenstate of the (3, 1)-QRAC Hamiltonian are zero.

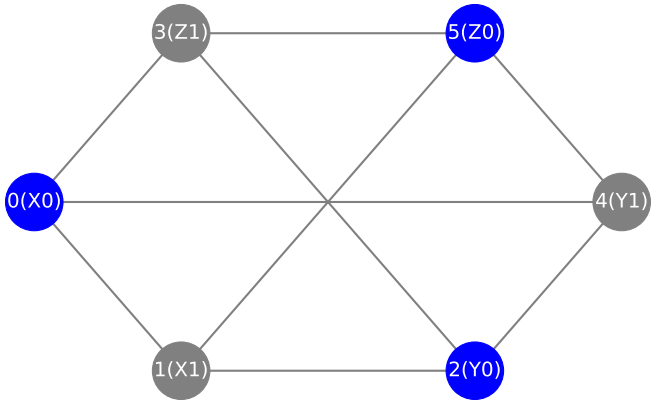


Fig. 4.18. An example of a graph and Pauli matrix assignment that yields a trace value of zero for all nodes.

The (3, 1)-QRAC Hamiltonian in this case is degenerate and the matrix has two maximum eigenvectors. Because the linear combination of the two maximum eigenvectors is also a maximum eigenstate, the choice of coefficients in the linear combination may affect the trace values. We observe from Fig. 4.19 that the phenomenon of trace values becoming zero can be resolved depending on the coefficients of the linear combination.

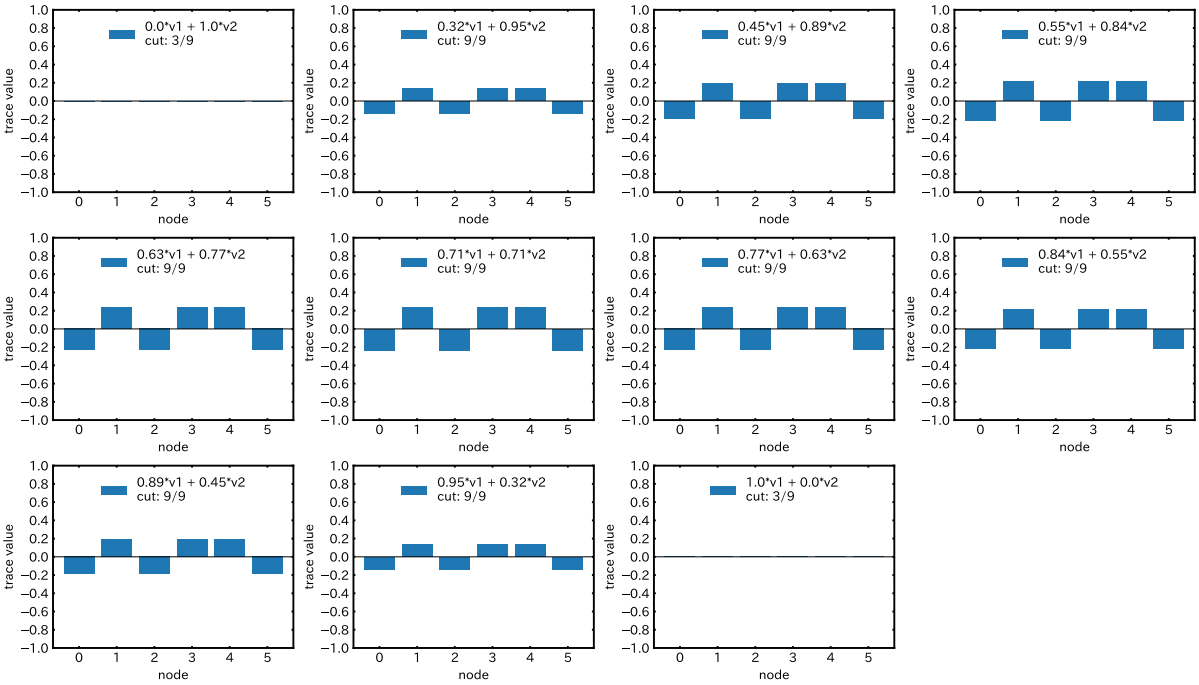


Fig. 4.19. The trace values corresponding to the nodes of the graph in Fig. 4.18 depending on the coefficients of the linear combination $\sqrt{k}v_1 + \sqrt{1-k}v_2$ of the two maximum eigenvectors v_1 and v_2.

Chapter 5

Conclusion

5.1 Summary of Contributions

A quantum computer is a computing device that leverages the laws of quantum mechanics to potentially surpass classical computers in certain applications. However, some of the most promising quantum algorithms remain beyond the practical reach of current devices, due to their limited number of qubits and the influence of noise. While quantum computers have become widely accessible via the cloud, we face the challenge of making informed decisions about which algorithms to execute on which qubits and devices. In this thesis, we addressed these issues by proposing a simple, hardware agnostic, but inherently quantum mechanical method of benchmarking the stability of qubits, and by analyzing the noise-robustness of a combinatorial optimization algorithm that reduces the use of qubits. The content of each chapter is summarized as follows.

In Chapter 2, we laid the groundwork for the thesis by illustrating how a quantum computer applies quantum gates to its qubits to perform computations. We also provided basic tools for expressing noisy quantum operations, which is essential for describing the behavior of quantum operations in current devices.

In Chapter 3, we proposed our method for benchmarking the qubits of a quantum computing device via quantum random number generation, along with benchmarking results from real quantum computing devices. As an introduction to the method, we reviewed conventional methods for benchmarking quantum computing devices and their qubits. We also outlined crucial concepts in random number generation including definitions of randomness, randomness extractors, types of random number generators, and conventional statistical methods of testing generators.

In Chapter 4, we discussed the noise-robustness of combinatorial optimization on a quantum computer when encoding the problem using QRAC. We found that using the (3, 1)-QRAC encoding leads to a more noise-robust approximation ratio compared to the Ising-type encoding. We further investigated the reason behind this noise-robustness by assuming depolarizing noise, and found that the (3,1)-QRAC encoding requires fewer measurements than the Ising-type encoding under the influence of noise.

5.2 Future Work

Here, we present some open problems related to Chapters 3 and 4. Some open problems in the benchmarking of current quantum computing devices include the following.

- Given the lack of a practical application that clearly demonstrates the unique value of a quantum computer, what performance should a benchmark be based on?
- With an unstable device with fluctuating performance and lack of reproducibility, how can we obtain lasting information from a benchmark?

- How can the noise of a quantum computing device be characterized, when access to the hardware parameters is limited?

We also identify open problems in random number generation in the following.

- How can the statistical randomness (the independence of bits) be tested when the output is non-identically distributed?
- How can random number generators be compared in terms of their capability to produce random bits?
- Are there other methods to confirm the existence of truly random bits without relying on the Bell inequality?

In combinatorial optimization using QRAC, the following problems are identified.

- While Pauli rounding yields on average a higher approximation than magic state rounding, are there instances where magic state rounding outperforms Pauli rounding? If so, can those instances be identified in advance?
- The relationship between the energy expectation value achieved by the candidate state and the expected approximation of Pauli rounding is unknown. Can such a relationship be established by restricting the types of problem?
- There are QRACs that encode multiple classical bits across several qubits. How do these encodings perform in combinatorial optimization?
- Are there types of noise that increase the mean approximation ratio of QRAO via Pauli rounding by resolving cases where the trace values drop to zero?

Solving the problems above will lead to significant progress in benchmarking and utilizing quantum computers. I hope this thesis supports users in making informed decisions about the devices, qubits, and algorithms suited for their specific applications.

Bibliography

[1] M. Ishida and H. Ikeda. Random number generator. *Annals Institute Statistical Mathematics*, 8:119–126, 1956.

[2] K. Tamura and Y. Shikano. Quantum random numbers generated by a cloud superconducting quantum computer. In T. Takagi, M. Wakayama, K. Tanaka, N. Kunihiro, K. Kimoto, and Y. Ikematsu, editors, *International Symposium on Mathematics, Quantum Theory, and Cryptography*, pages 17–37, Singapore, 2021. Springer Singapore.

[3] D. H. Douglas and T. K. Peucker. Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. *Cartogr. Geogr. Inf. Sci.*, 10:112–122, 1973.

[4] G. Aleksandrowicz, T. Alexander, P. Barkoutsos, L. Bello, Y. Ben-Haim, D. Bucher, F. J. Cabrera-Hernández, J. Carballo-Franquis, A. Chen, C.-F. Chen, J. M. Chow, A. D. Córcoles-Gonzales, A. J. Cross, A. Cross, J. Cruz-Benito, C. Culver, S. D. L. P. González, E. D. L. Torre, D. Ding, E. Dumitrescu, I. Duran, P. Eendebak, M. Everitt, I. F. Sertage, A. Frisch, A. Fuhrer, J. Gambetta, B. G. Gago, J. Gomez-Mosquera, D. Greenberg, I. Hamamura, V. Havlicek, J. Hellmers, Lukasz Herok, H. Horii, S. Hu, T. Imamichi, T. Itoko, A. Javadi-Abhari, N. Kanazawa, A. Karazeev, K. Krsulich, P. Liu, Y. Luh, Y. Maeng, M. Marques, F. J. Martín-Fernández, D. T. McClure, D. McKay, S. Meesala, A. Mezzacapo, N. Moll, D. M. Rodríguez, G. Nannicini, P. Nation, P. Ollitrault, L. J. O'Riordan, H. Paik, J. Pérez, A. Phan, M. Pistoia, V. Prutyanov, M. Reuter, J. Rice, A. R. Davila, R. H. P. Rudy, M. Ryu, N. Sathaye, C. Schnabel, E. Schoute, K. Setia, Y. Shi, A. Silva, Y. Siraichi, S. Sivarajah, J. A. Smolin, M. Soeken, H. Takahashi, I. Tavernelli, C. Taylor, P. Taylour, K. Trabing, M. Treinish, W. Turner, D. Vogt-Lee, C. Vuillot, J. A. Wildstrom, J. Wilson, E. Winston, C. Wood, S. Wood, S. Wörner, I. Y. Akhalwaya, and C. Zoufal. Qiskit: An Open-source Framework for Quantum Computing, 2019.

[5] K. Tamura, Y. Suzuki, R. Raymond, H. C. Watanabe, Y. Sato, R. Kondo, M. Sugawara, and N. Yamamoto. Noise robustness of quantum relaxation for combinatorial optimization. *IEEE Trans. Quantum Eng.*, 5:1–9, 2024.

[6] A. Rukhin, J. Soto, J. Nechvatal, M. Smid, E. Barker, S. Leigh, M. Levenson, M. Vangel, D. Banks, A. Heckert, J. Dray, and S. Vo. A statistical test suite for random and pseudorandom number generators for cryptographic applications. *NIST Spec. Publ.*, 800-22(Rev. 1a), 2010.

[7] P. Benioff. The computer as a physical system: A microscopic quantum mechanical hamiltonian model of computers as represented by turing machines. *J. Stat. Phys.*, 22(5):563–591, 1980.

[8] Y. I. Manin. *Mathematics as metaphor: Selected essays of Yuri I. Manin*, volume 20. American Mathematical Soc., 2007.

[9] R. P. Feynman. Simulating physics with computers. *Int. J. Theor. Phys.*, 21(6/7):467–488, 1982.

[10] M. A. Nielsen and I. L. Chuang. *Quantum Computation and Quantum Information: 10th Anniversary Edition*. Cambridge University Press, USA, 10th edition, 2011.

[11] P. W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM J. Comput.*, 26(5):1484–1509, 1997.

[12] L. K. Grover. A fast quantum mechanical algorithm for database search. In *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing - STOC '96*, New York, New York, USA, 1996. ACM Press.

[13] J. Preskill. Quantum computing in the NISQ era and beyond. *Quantum*, 2:79, 2018.

[14] C. Gidney and M. Ekerå. How to factor 2048 bit RSA integers in 8 hours using 20 million noisy qubits. *Quantum*, 5(433):433, 2021.

[15] S. Chen, J. Cotler, H.-Y. Huang, and J. Li. The complexity of NISQ. *Nat. Commun.*, 14(1):6001, 2023.

[16] H.-L. Huang, X.-Y. Xu, C. Guo, G. Tian, S.-J. Wei, X. Sun, W.-S. Bao, and G.-L. Long. Near-term quantum computing techniques: Variational quantum algorithms, error mitigation, circuit compilation, benchmarking and classical simulation. *Sci. China Phys. Mech. Astron.*, 66(5), 2023.

[17] A. Callison and N. Chancellor. Hybrid quantum-classical algorithms in the noisy intermediate-scale quantum era and beyond. *Phys. Rev. A*, 106:010101, 2022.

[18] A. Peruzzo, J. McClean, P. Shadbolt, M.-H. Yung, X.-Q. Zhou, P. J. Love, A. Aspuru-Guzik, and J. L. O'brien. A variational eigenvalue solver on a photonic quantum processor. *Nat. Commun.*, 5(1):4213, 2014.

[19] J. R. McClean, J. Romero, R. Babbush, and A. Aspuru-Guzik. The theory of variational hybrid quantum-classical algorithms. *New J. Phys.*, 18(2):023023, 2016.

[20] J. Romero, R. Babbush, J. R. McClean, C. Hempel, P. J. Love, and A. Aspuru-Guzik. Strategies for quantum computing molecular energies using the unitary coupled cluster ansatz. *Quantum Sci. Technol.*, 4(1):014008, 2018.

[21] A. Y. Kitaev. Quantum measurements and the abelian stabilizer problem. *arXiv preprint arXiv:9511026*, 1995.

[22] D. S. Abrams and S. Lloyd. Simulation of many-body fermi systems on a universal quantum computer. *Phys. Rev. Lett.*, 79(13):2586–2589, 1997.

[23] D. S. Abrams and S. Lloyd. Quantum algorithm providing exponential speed increase for finding eigenvalues and eigenvectors. *Phys. Rev. Lett.*, 83(24):5162–5165, 1999.

[24] P. Deglmann, A. Schäfer, and C. Lennartz. Application of quantum calculations in the chemical industry-an overview. *Int. J. Quantum Chem.*, 115(3):107–136, 2015.

[25] B. J. Williams-Noonan, E. Yuriev, and D. K. Chalmers. Free energy methods in drug design: Prospects of "alchemical perturbation" in medicinal chemistry. *J. Med. Chem.*, 61(3):638–649, 2018.

[26] A. Van der Ven, Z. Deng, S. Banerjee, and S. P. Ong. Rechargeable alkali-ion battery materials: Theory and computation. *Chem. Rev.*, 120(14):6977–7019, 2020.

[27] B. Fuller, C. Hadfield, J. R. Glick, T. Imamichi, T. Itoko, R. J. Thompson, Y. Jiao, M. M. Kagele, A. W. Blom-Schieber, R. Raymond, and A. Mezzacapo. Approximate solutions of combinatorial problems via quantum relaxations. *IEEE Trans. Quantum Eng.*, 5:1–15, 2024.

[28] X. Liu, A. Angone, R. Shaydulin, I. Safro, Y. Alexeev, and L. Cincio. Layer VQE: A variational approach for combinatorial optimization on noisy quantum computers. *IEEE Trans. Quantum Eng.*, 3:1–20, 2022.

[29] U. Azad, B. K. Behera, E. A. Ahmed, P. K. Panigrahi, and A. Farouk. Solving vehicle routing problem using quantum approximate optimization algorithm. *IEEE Trans. Intell. Transp. Syst.*, 2022.

[30] K. Kurowski, T. Pecyna, M. Slysz, R. Różycki, G. Waligóra, and J. Wceglarz. Application of quantum approximate optimization algorithm to job shop scheduling problem. *Eur. J. Oper. Res.*, 310(2):518–528, 2023.

[31] D. J. Egger, C. Gambella, J. Marecek, S. McFaddin, M. Mevissen, R. Raymond, A. Simonetto, S. Woerner, and E. Yndurain. Quantum computing for finance: State-of-the-art and future prospects. *IEEE Trans. Quantum Eng.*, 1:1–24, 2020.

[32] S. Marsh and J. Wang. Combinatorial optimization via highly efficient quantum walks. *Phys. Rev. Research*, 2:023302, 2020.

[33] J. Eisert, D. Hangleiter, N. Walk, I. Roth, D. Markham, R. Parekh, U. Chabaud, and E. Kashefi. Quantum certification and benchmarking. *Nat. Rev. Phys.*, 2(7):382–390, 2020.

[34] E. Magesan, J. M. Gambetta, B. R. Johnson, C. A. Ryan, J. M. Chow, S. T. Merkel, M. P. da Silva, G. A. Keefe, M. B. Rothwell, T. A. Ohki, M. B. Ketchen, and M. Steffen. Efficient measurement of quantum gate error by interleaved randomized benchmarking. *Phys. Rev. Lett.*, 109(8):080505, 2012.

[35] D. S. França and A. K. Hashagen. Approximate randomized benchmarking for finite groups. *J. Phys. A Math. Theor.*, 51(39):395302, 2018.

[36] S. G. Stanchev and N. V. Vitanov. Multipass quantum process tomography. *Sci. Rep.*, 14(1):18185, 2024.

[37] S. Resch and U. R. Karpuzcu. Benchmarking quantum computers and the impact of quantum noise. *ACM Comput. Surv.*, 54(7):1–35, 2022.

[38] C. H. Baldwin, K. Mayer, N. C. Brown, C. Ryan-Anderson, and D. Hayes. Re-examining the quantum volume test: Ideal distributions, compiler optimizations, confidence intervals, and scalable resource estimations. *Quantum*, 6(707):707, 2022.

[39] S. Dasgupta and T. Humble. Impact of unreliable devices on stability of quantum computations. *ACM T. Quantum Comput.*, 5(4), Oct. 2024.

[40] R. Blume-Kohout and K. C. Young. A volumetric framework for quantum computer benchmarks. *Quantum*, 4(362):362, 2020.

[41] K. Wright, K. M. Beck, S. Debnath, J. M. Amini, Y. Nam, N. Grzesiak, J.-S. Chen, N. C. Pisenti, M. Chmielewski, C. Collins, K. M. Hudek, J. Mizrahi, J. D. Wong-Campos, S. Allen, J. Apisdorf, P. Solomon, M. Williams, A. M. Ducore, A. Blinov, S. M. Kreikemeier, V. Chaplin, M. Keesan, C. Monroe, and J. Kim. Benchmarking an 11-qubit quantum computer. *Nat. Commun.*, 10(1):5464, 2019.

[42] M. Herrero-Collantes and J. C. Garcia-Escartin. Quantum random number generators. *Rev. Mod. Phys.*, 89:015004, 2017.

[43] Y. Li, Y. Fei, W. Wang, X. Meng, H. Wang, Q. Duan, and Z. Ma. Quantum random number generator using a cloud superconducting quantum computer based on source-independent protocol. *Sci. Rep.*, 11(1):23873, 2021.

[44] K. Tamura and Y. Shikano. Quantum random number generation with the superconducting quantum computer ibm 20q tokyo. In M. Hirvensalo and A. Yakaryilmaz, editors, *Proceedings of Workshop on Quantum Computing and Quantum Information*, volume 30 of *TUCS Lecture Notes*, pages 13–25, 2019.

[45] S. Dasgupta and T. S. Humble. Characterizing the stability of NISQ devices. In *2020 IEEE International Conference on Quantum Computing and Engineering (QCE)*. IEEE, 2020.

[46] S. Dasgupta and T. S. Humble. Stability of noisy quantum computing devices. *arXiv preprint arXiv:2105.09472*, 2021.

[47] S. Dasgupta and T. S. Humble. Characterizing the reproducibility of noisy quantum circuits. *Entropy*, 24(2), 2022.

[48] S. Dasgupta and T. S. Humble. Adaptive stabilization of quantum circuits executed on unstable devices. In *2022 IEEE International Conference on Quantum Computing and Engineering (QCE)*, pages 736–740, 2022.

[49] S. Dasgupta and T. Humble. Assessing the stability of noisy quantum computation. In *Optical Engineering + Applications*, 2022.

[50] S. Dasgupta, T. S. Humble, and A. Danageozian. Adaptive mitigation of time-varying quantum noise . In *2023 IEEE International Conference on Quantum Computing and Engineering (QCE)*, pages 99–110, Los Alamitos, CA, USA, Sept. 2023. IEEE Computer Society.

[51] S. Dasgupta and T. S. Humble. Reliable devices yield stable quantum computations. In *2023 IEEE International Conference on Quantum Computing and Engineering (QCE)*, volume 02, pages 223–226, 2023.

[52] S. Dasgupta and T. S. Humble. Improving probabilistic error cancellation in the presence of nonstationary noise. *IEEE Trans. Quantum Eng.*, 5:1–14, 2024.

[53] S. Dasgupta. Stability of quantum computers. *arXiv preprint arXiv:2404.19082*, 2024.

[54] S. Wiesner. Conjugate coding. *SIGACT News*, 15(1):78–88, 1983.

[55] A. Ambainis, A. Nayak, A. Ta-Shma, and U. Vazirani. Dense quantum coding and a lower bound for 1-way quantum automata. In *Proceedings of the thirty-first annual ACM symposium on Theory of Computing*, STOC '99, pages 376–383, New York, NY, USA, 1999. Association for Computing Machinery.

[56] A. Ambainis, A. Nayak, A. Ta-Shma, and U. Vazirani. Dense quantum coding and quantum finite automata. *J. ACM*, 49(4):496–511, 2002.

[57] A. Ambainis, D. Leung, L. Mancinska, and M. Ozols. Quantum random access codes with shared randomness. *arXiv preprint arXiv:0810.2937*, 2008.

[58] A. S. Holevo. On capacity of a quantum communications channel. *Probl. Peredachi Inf.*, 15(4):3–11, 1979.

[59] A. Nayak. Optimal lower bounds for quantum automata and random access codes. In *40th Annual Symposium on Foundations of Computer Science (Cat. No.99CB37039)*. IEEE Comput. Soc, 2003.

[60] H. Klauck. Lower bounds for quantum communication complexity. In *Proceedings 42nd IEEE Symposium on Foundations of Computer Science*. IEEE, 2001.

[61] D. Gavinsky, J. Kempe, O. Regev, and R. de Wolf. Bounded-error quantum state identification and exponential separations in communication complexity. *SIAM J. Comput.*, 39(1):1–24, 2009.

[62] S. Aaronson. Limitations of quantum advice and one-way communication. In *Proceedings. 19th IEEE Annual Conference on Computational Complexity, 2004*. IEEE, 2004.

[63] E. Farhi, J. Goldstone, and S. Gutmann. A quantum approximate optimization algorithm applied to a bounded occurrence constraint problem. *arXiv [quant-ph]*, 2014.

[64] E. Farhi, J. Goldstone, and S. Gutmann. A quantum approximate optimization algorithm. *arXiv preprint arXiv:1411.4028*, 2014.

[65] N. P. Landsman. Born rule and its interpretation. In *Compendium of Quantum Physics*, pages 64–70. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.

[66] Y. Shimada and IPSJ Publishing Committee. *Quantum Computing: From Fundamental Algorithms to Quantum Machine Learning*. Ohmsha Ltd., 2020.

[67] Y. Li, Y. Fei, W. Wang, X. Meng, H. Wang, Q. Duan, and Z. Ma. Quantum random number generator using a cloud superconducting quantum computer based on source-independent protocol. *Sci. Rep.*, 11(1):23873, 2021.

[68] M. A. Aziz, B. P. Gond, S. Nandi, S. Ray, D. Bhoumik, and R. Majumdar. Thermal relaxation error on QKD: Effect and a probable bypass. *arXiv preprint arXiv:2207.01159*, 2022.

[69] K. Kraus, A. Böhm, J. D. Dollard, and W. H. Wootters, editors. *States, effects, and operations fundamental notions of quantum theory*. Springer Berlin Heidelberg, Berlin, Heidelberg, 1983.

[70] B. Nachman, M. Urbanek, W. A. de Jong, and C. W. Bauer. Unfolding quantum computer readout noise. *Npj Quantum Inf.*, 6(1), 2020.

[71] K. Bharti, A. Cervera-Lierta, T. H. Kyaw, T. Haug, S. Alperin-Lea, A. Anand, M. Degroote, H. Heimonen, J. S. Kottmann, T. Menke, W.-K. Mok, S. Sim, L.-C. Kwek, and A. Aspuru-Guzik. Noisy intermediate-scale quantum algorithms. *Rev. Mod. Phys.*, 94(1), 2022.

[72] T. Proctor, K. Young, A. D. Baczewski, and R. Blume-Kohout. Benchmarking quantum computers. *arXiv preprint arXiv:2407.08828*, 2024.

[73] J. Emerson, R. Alicki, and K. Życzkowski. Scalable noise estimation with random unitary operators. *J. Opt. B Quantum Semiclassical Opt.*, 7(10):S347–S352, 2005.

[74] J. Emerson, M. Silva, O. Moussa, C. Ryan, M. Laforest, J. Baugh, D. G. Cory, and R. Laflamme. Symmetrized characterization of noisy quantum processes. *Science*, 317(5846):1893–1896, 2007.

[75] E. Knill, D. Leibfried, R. Reichle, J. Britton, R. B. Blakestad, J. D. Jost, C. Langer, R. Ozeri, S. Seidelin, and D. J. Wineland. Randomized benchmarking of quantum gates. *Phys. Rev. A*, 77(1), 2008.

[76] S. Gulde, M. Riebe, G. P. T. Lancaster, C. Becher, J. Eschner, H. Häffner, F. Schmidt-Kaler, I. L. Chuang, and R. Blatt. Implementation of the deutsch-jozsa algorithm on an ion-trap quantum computer. *Nature*, 421(6918):48–50, 2003.

[77] C. Negrevergne, T. S. Mahesh, C. A. Ryan, M. Ditty, F. Cyr-Racine, W. Power, N. Boulant, T. Havel, D. G. Cory, and R. Laflamme. Benchmarking quantum control methods on a 12-qubit system. *Phys. Rev. Lett.*, 96(17):170501, 2006.

[78] B. P. Lanyon, T. J. Weinhold, N. K. Langford, M. Barbieri, D. F. V. James, A. Gilchrist, and A. G. White. Experimental demonstration of a compiled version of shor's algorithm with quantum entanglement. *Phys. Rev. Lett.*, 99(25):250505, 2007.

[79] L. DiCarlo, J. M. Chow, J. M. Gambetta, L. S. Bishop, B. R. Johnson, D. I. Schuster, J. Majer, A. Blais, L. Frunzio, S. M. Girvin, and R. J. Schoelkopf. Demonstration of two-qubit algorithms with a superconducting quantum processor. *Nature*, 460(7252):240–244, 2009.

[80] E. Kawakami, T. Jullien, P. Scarlino, D. R. Ward, D. E. Savage, M. G. Lagally, V. V. Dobrovitski, M. Friesen, S. N. Coppersmith, M. A. Eriksson, and L. M. K. Vandersypen. Gate fidelity and coherence of an electron spin in an si/SiGe quantum dot with micromagnet. *Proc. Natl. Acad. Sci. U. S. A.*, 113(42):11738–11743, 2016.

[81] D. Greenbaum. Introduction to quantum gate set tomography. *arXiv preprint arXiv:1509.02921*, 2015.

[82] R. Bhandari and N. A. Peters. On the general constraints in single qubit quantum process tomography. *Sci. Rep.*, 6(1):26004, 2016.

[83] A. Anshu and S. Arunachalam. A survey on the complexity of learning quantum states. *arXiv preprint arXiv:2305.20069*, 2023.

[84] J. Helsen, I. Roth, E. Onorati, A. H. Werner, and J. Eisert. General framework for randomized benchmarking. *PRX quantum*, 3(2), 2022.

[85] D. Gottesman. Theory of fault-tolerant quantum computation. *Phys. Rev. A*, 57(1):127–137, 1998.

[86] A. W. Cross, L. S. Bishop, S. Sheldon, P. D. Nation, and J. M. Gambetta. Validating quantum computers using randomized model circuits. *Phys. Rev. A*, 100(3), 2019.

[87] M. Born. Is classical mechanics in fact deterministic? In *Heidelberg Science Library*, pages 78–83. Springer Berlin Heidelberg, Berlin, Heidelberg, 1969.

[88] A. Eagle. Randomness is unpredictability. *Br. J. Philos. Sci.*, 56(4):749–790, 2005.

[89] R. von Mises. Probability, statistics and truth. *J. Philos.*, 36(25):696, 1939.

[90] A. Church. On the concept of a random sequence. *Bull. New Ser. Am. Math. Soc.*, 46(2):130–135, 1940.

[91] P. Martin-Löf. The definition of random sequences. *Inf. Contr.*, 9(6):602–619, 1966.

[92] P. Martin-Löf. The literature on von mises' kollektivs revisited. *Theoria*, 35(1):12–37, 1969.

[93] P. Martin-Löf. On the notion of randomness. In *Intuitionism and Proof Theory: Proceedings of the Summer Conference at Buffalo N.Y. 1968*, Studies in logic and the foundations of mathematics, pages 73–78. Elsevier, 1970.

[94] P. M. B. Vitanyi. Randomness. *arXiv preprint arXiv:0110086*, 2001.

[95] A. N. Kolmogorov and V. A. Uspenskii. Algorithms and randomness. *Theory Probab. Appl.*, 32(3):389–412, 1988.

[96] G. J. Chaitin. Randomness and mathematical proof. *Sci. Am.*, 232(5):47–52, 1975.

[97] R. J. Solomonoff. Algorithmic probability: Theory and applications. In *Information Theory and Statistical Learning*, pages 1–23. Springer US, Boston, MA, 2008.

[98] H. Zenil. *Randomness through computation: Some answers, more questions*. World Scientific Publishing, Singapore, Singapore, 2011.

[99] J. von Neumann. Various techniques used in connection with random digits. *Natl. Bur. Stand. Appl. Math. Ser.*, 12:36–38, 1951.

[100] R. Shaltiel. An introduction to randomness extractors. In *Automata, Languages and Programming*, Lecture notes in computer science, pages 21–41. Springer Berlin Heidelberg, Berlin, Heidelberg, 2011.

[101] M. Santha and U. V. Vazirani. Generating quasi-random sequences from semi-random sources. *J. Comput. Syst. Sci.*, 33(1):75–87, 1986.

[102] M. S. Turan, E. Barker, J. Kelsey, K. A. McKay, M. L. Baish, and M. Boyle. Recommendation for the entropy sources used for random bit generation. *NIST Spec. Publ.*, 800-90B, 2018.

[103] N. Nisan and D. Zuckerman. Randomness is linear in space. *J. Comput. Syst. Sci.*, 52(1):43–52, 1996.

[104] J. Radhakrishnan and A. Ta-Shma. Bounds for dispersers, extractors, and depth-two superconcentrators. *SIAM J. Discrete Math.*, 13(1):2–24, 2000.

[105] Y. Mansour, N. Nisan, and P. Tiwari. The computational complexity of universal hashing. In *Proceedings Fifth Annual Structure in Complexity Theory Conference*. IEEE Comput. Soc. Press, 2002.

[106] X. Zhang, Y.-Q. Nie, H. Liang, and J. Zhang. FPGA implementation of toeplitz hashing extractor for real time post-processing of raw random numbers. *2016 IEEE-NPSS Real Time Conf. (RT)*, pages 1–5, 2016.

[107] Q. Li, X. Sun, X. Zhang, and H. Zhou. Improved real-time post-processing for quantum random number generators. *Adv. Quantum Technol.*, 7(7), 2024.

[108] L. Trevisan. Extractors and pseudorandom generators. *J. ACM*, 48(4):860–879, 2001.

[109] A. De, C. Portmann, T. Vidick, and R. Renner. Trevisan's extractor in the presence of quantum side information. *SIAM J. Comput.*, 41(4):915–940, 2012.

[110] S. Vadhan. The unified theory of pseudorandomness: guest column. *SIGACT News*, 38(3):39–54, 2007.

[111] W. Mauerer, C. Portmann, and V. B. Scholz. A modular framework for randomness extraction based on trevisan's construction. *arXiv preprint arXiv:1212.0520*, 2012.

[112] D. E. Knuth. *The Art of Computer Programming*, volume 2. Addison-Wesley, third edition, 1997.

[113] P. L'Ecuyer. Uniform random number generators. In *1998 Winter Simulation Conference. Proceedings (Cat. No.98CH36274)*. IEEE, 2002.

[114] W. E. Thomson. A modified congruence method of generating pseudo-random numbers. *Comput. J.*, 1(2):83–83, 1958.

[115] P. L'Ecuyer. Tables of linear congruential generators of different sizes and good lattice structure. *Math. Comput.*, 68(225):249–261, 1999.

[116] S. K. Park and K. W. Miller. Random number generators: good ones are hard to find. *Commun. ACM*, 31(10):1192–1201, 1988.

[117] G. L. Steele, Jr and S. Vigna. Computationally easy, spectrally good multipliers for congruential pseudorandom number generators. *Softw. Pract. Exp.*, 52(2):443–458, 2022.

[118] G. Marsaglia. Random numbers fall mainly in the planes. *Proc. Natl. Acad. Sci. U. S. A.*, 61(1):25–28, 1968.

[119] J. Boyar. Inferring sequences produced by a linear congruential generator missing low-order bits. *J. Cryptology*, 1(3):177–184, 1989.

[120] T. E. Hull and A. R. Dobell. Random number generators. *SIAM Rev. Soc. Ind. Appl. Math.*, 4(3):230–254, 1962.

[121] M. Matsumoto and T. Nishimura. Mersenne twister: A 623-dimensionally equidistributed uniform pseudorandom number generator. *ACM Trans. Model. Comput. Simul.*, 8(1):3–30, 1998.

[122] M. Matsumoto. To the users of pseudorandom number generators—from a developer of mersenne twister. *Jpn. Stat. Soc.*, 35(2):165–180, 2006.

[123] S. Harase. Conversion of mersenne twister to double-precision floating-point numbers. *Math. Comput. Simul.*, 161:76–83, 2019.

[124] C. Ryan, M. Kshirsagar, G. Vaidya, A. Cunningham, and R. Sivaraman. Design of a cryptographically secure pseudo random number generator with grammatical evolution. *Sci. Rep.*, 12(1):8602, 2022.

[125] L. Blum, M. Blum, and M. Shub. A simple unpredictable pseudo-random number generator. *SIAM J. COMPUT.*, 15(2):364–383, 1986.

[126] L. Blum, M. Blum, and M. Shub. Comparison of two pseudo-random number generators. In *Advances in Cryptology: Proceedings of CRYPTO '82*, pages 61–78. Plenum, 1982.

[127] M. Geisler, M. Krøigård, and A. Danielsen. About random bits, 2004.

[128] E. B. Guedes, F. M. de Assis, and B. Lula, Jr. Quantum attacks on pseudorandom generators. *Math. Struct. Comput. Sci.*, 23(3):608–634, 2013.

[129] S. Pironio, A. Acín, S. Massar, A. B. de la Giroday, D. N. Matsukevich, P. Maunz, S. Olmschenk, D. Hayes, L. Luo, T. A. Manning, and C. Monroe. Random numbers certified by bell's theorem. *Nature*, 464(7291):1021–1024, 2010.

[130] M. Avesani, D. G. Marangon, G. Vallone, and P. Villoresi. Source-device-independent heterodyne-based quantum random number generator at 17 gbps. *Nat. Commun.*, 9(1):5365, 2018.

[131] V. Mannalatha, S. Mishra, and A. Pathak. A comprehensive review of quantum random number generators: concepts, classification and the origin of randomness. *Quantum Inf. Process.*, 22(12), 2023.

[132] Y. Yoshizawa, H. Kimura, H. Inoue, K. Fujita, M. Toyama, and O. Miyatake. Physical random numbers generated by radioactivity. *J. Japanese Soc. Comput. Statist.*, 12(1):67–81, 1999.

[133] J. G. Rarity, P. C. M. Owens, and P. R. Tapster. Quantum random-number generation and key sharing. *J. Mod. Opt.*, 41(12):2435–2444, 1994.

[134] T. Jennewein, C. Simon, G. Weihs, H. Weinfurter, and A. Zeilinger. Quantum cryptography with entangled photons. *Phys. Rev. Lett.*, 84(20):4729–4732, 2000.

[135] A. Acín and L. Masanes. Certified randomness in quantum physics. *Nature*, 540(7632):213–219, 2016.

[136] J. F. Clauser, M. A. Horne, A. Shimony, and R. A. Holt. Proposed experiment to test local hidden-variable theories. *Phys. Rev. Lett.*, 23(15):880–884, 1969.

[137] S. Storz, J. Schär, A. Kulikov, P. Magnard, P. Kurpiers, J. Lütolf, T. Walter, A. Copetudo, K. Reuer, A. Akin, J.-C. Besse, M. Gabureac, G. J. Norris, A. Rosario, F. Martin, J. Martinez, W. Amaya, M. W. Mitchell, C. Abellan, J.-D. Bancal, N. Sangouard, B. Royer, A. Blais, and A. Wallraff. Loophole-free bell inequality violation with superconducting circuits. *Nature*, 617(7960):265–270, 2023.

[138] BIG Bell Test Collaboration. Challenging local realism with human choices. *Nature*, 557(7704):212–216, 2018.

[139] P. L'Ecuyer and R. Simard. TestU01: A C library for empirical testing of random number generators. *ACM Trans. Math. Softw.*, 33:22, 2007.

[140] Y. Shikano, K. Tamura, and R. Raymond. Detecting temporal correlation via quantum random number generation. *EPTCS*, 315:18–25, 2020.

[141] F. Glover, G. Kochenberger, R. Hennig, and Y. Du. Quantum bridge analytics I: a tutorial on formulating and using QUBO models. *Ann. Oper. Res.*, 314(1):141–183, 2022.

[142] M. Grotschel and L. Lovász. Combinatorial optimization. *Handb. Comb.*, 2(1541-1597):4, 1995.

[143] R. M. Karp. Reducibility among combinatorial problems. In R. E. Miller, J. W. Thatcher, and J. D. Bohlinger, editors, *Complexity of Computer Computations: Proceedings of a symposium on the Complexity of Computer Computations, held March 20–22, 1972, at the IBM Thomas J. Watson Research Center, Yorktown Heights, New York, and sponsored by the Office of Naval Research, Mathematics Program, IBM World Trade Corporation, and the IBM Research Mathematical Sciences Department*, pages 85–103. Springer US, Boston, MA, 1972.

[144] Z. A. Mann. The top eight misconceptions about np-hardness. *Computer*, 50(5):72–79, 2017.

[145] F. Barahona, M. Grötschel, M. Jünger, and G. Reinelt. An application of combinatorial optimization to statistical physics and circuit layout design. *Oper. Res.*, 36(3):493–513, 1988.

[146] S. Gu and Y. Yang. A deep learning algorithm for the max-cut problem based on pointer network structure with supervised learning and reinforcement learning strategies. *Mathematics*, 8(2):298, 2020.

[147] R.-S. Wang and L.-M. Wang. Maximum cut in fuzzy nature: Models and algorithms. *J. Comput. Appl. Math.*, 234(1):240–252, 2010.

[148] H. Fakhravar. Combining heuristics and exact algorithms: A review. *arXiv preprint arXiv:2202.02799*, 2022.

[149] S. Bylka, A. Idzik, and Z. Tuza. Maximum cuts: Improvements and local algorithmic analogues of the edwards-erdos inequality. *Discrete Math.*, 194(1-3):39–58, 1999.

[150] G. Gutin and A. Yeo. Lower bounds for maximum weighted cut. *SIAM J. Discrete Math.*, 37(2):1142–1161, 2023.

[151] M. X. Goemans and D. P. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *J. ACM*, 42(6):1115–1145, 1995.

[152] S. Burer, R. D. C. Monteiro, and Y. Zhang. Rank-two relaxation heuristics for MAX-CUT and other binary quadratic programs. *SIAM J. Optim.*, 12(2):503–521, 2002.

[153] L. Zhou, S.-T. Wang, S. Choi, H. Pichler, and M. D. Lukin. Quantum approximate optimization algorithm: Performance, mechanism, and implementation on near-term devices. *Phys. Rev. X.*, 10(2), 2020.

[154] G. B. Mbeng, R. Fazio, and G. Santoro. Quantum annealing: a journey through digitalization, control, and hybrid quantum variational schemes. *arXiv preprint arXiv:1906.08948*, 2019.

[155] Z. Wang, S. Hadfield, Z. Jiang, and E. G. Rieffel. Quantum approximate optimization algorithm for MaxCut: A fermionic view. *Phys. Rev. A*, 97(2), 2018.

[156] K. Blekos, D. Brand, A. Ceschini, C.-H. Chou, R.-H. Li, K. Pandya, and A. Summer. A review on quantum approximate optimization algorithm and its variants. *Phys. Rep.*, 1068:1–66, 2024.

[157] J. Wurtz and P. Love. MaxCut quantum approximate optimization algorithm performance guarantees for $p > 1$. *Phys. Rev. A*, 103(4), 2021.

[158] J. Marshall, F. Wudarski, S. Hadfield, and T. Hogg. Characterizing local noise in QAOA circuits. *IOP SciNotes*, 1(2):025208, 2020.

[159] G. G. Guerreschi and A. Y. Matsuura. QAOA for max-cut requires hundreds of qubits for quantum speed-up. *Sci. Rep.*, 9(1):6903, 2019.

[160] J. Tilly, H. Chen, S. Cao, D. Picozzi, K. Setia, Y. Li, E. Grant, L. Wossnig, I. Rungger, G. H. Booth, and J. Tennyson. The variational quantum eigensolver: A review of methods and best practices. *Phys. Rep.*, 986:1–128, 2022.

[161] M. Stechly. Introduction to Variational Quantum Algorithms. *arXiv preprint arXiv:2402.15879*, 2024.

[162] D. A. Fedorov, B. Peng, N. Govind, and Y. Alexeev. VQE method: a short survey and recent developments. *Mater. Theory*, 6(1):1–21, 2022.

[163] P. Jordan and E. P. Wigner. Über das paulische Äquivalenzverbot. In *The Collected Works of Eugene Paul Wigner*, pages 109–129. Springer Berlin Heidelberg, Berlin, Heidelberg, 1993.

[164] S. B. Bravyi and A. Y. Kitaev. Fermionic quantum computation. *Ann. Phys. (N. Y.)*, 298(1):210–226, 2002.

[165] A. Kandala, A. Mezzacapo, K. Temme, M. Takita, M. Brink, J. M. Chow, and J. M. Gambetta. Hardware-efficient variational quantum eigensolver for small molecules and quantum magnets. *Nature*, 549(7671):242–246, 2017.

[166] L. Leone, S. F. E. Oliviero, L. Cincio, and M. Cerezo. On the practical usefulness of the hardware efficient ansatz. *Quantum*, 2022.

[167] L. Bittel and M. Kliesch. Training variational quantum algorithms is NP-hard. *Phys. Rev. Lett.*, 127(12):120502, 2021.

[168] L. Banchi and G. E. Crooks. Measuring analytic gradients of general quantum evolution with the stochastic parameter shift rule. *Quantum*, 5(386):386, 2021.

[169] M. Schuld, V. Bergholm, C. Gogolin, J. Izaac, and N. Killoran. Evaluating analytic gradients on quantum hardware. *Phys. Rev. A*, 99(3), 2019.

[170] K. Mitarai, M. Negoro, M. Kitagawa, and K. Fujii. Quantum circuit learning. *Phys. Rev. A*, 98:032309, 2018.

[171] J. Li, X. Yang, X. Peng, and C.-P. Sun. Hybrid quantum-classical approach to quantum optimal control. *Phys. Rev. Lett.*, 118(15), 2017.

[172] M. Cerezo, A. Sone, T. Volkoff, L. Cincio, and P. J. Coles. Cost function dependent barren plateaus in shallow parametrized quantum circuits. *Nat. Commun.*, 12(1):1791, 2021.

[173] J. R. McClean, S. Boixo, V. N. Smelyanskiy, R. Babbush, and H. Neven. Barren plateaus in quantum neural network training landscapes. *Nat. Commun.*, 9(1):4812, 2018.

[174] A. Arrasmith, M. Cerezo, P. Czarnik, L. Cincio, and P. J. Coles. Effect of barren plateaus on gradient-free optimization. *Quantum*, 5(558):558, 2021.

[175] K. M. Nakanishi, K. Fujii, and S. Todo. Sequential minimal optimization for quantum-classical hybrid algorithms. *Phys. Rev. Research*, 2(4):043158, 2020.

[176] W. Li, Y. Ge, S.-X. Zhang, Y.-Q. Chen, and S. Zhang. Efficient and robust parameter optimization of the unitary coupled-cluster ansatz. *J. Chem. Theory Comput.*, 20(9):3683–3696, 2024.

[177] K. Teramoto, R. Raymond, E. Wakakuwa, and H. Imai. Quantum-relaxation based optimization algorithms: Theoretical extensions. *arXiv preprint arXiv:2302. 09481*, 2023.

[178] G. Nannicini. Performance of hybrid quantum-classical variational heuristics for combinatorial optimization. *Phys. Rev. E.*, 99(1-1):013304, 2019.

[179] P. Díez-Valle, D. Porras, and J. J. García-Ripoll. Quantum variational optimization: The role of entanglement and problem hardness. *Phys. Rev. A*, 104(6), 2021.

[180] K. Teramoto, R. Raymond, and H. Imai. The role of entanglement in quantum-relaxation based optimization algorithms. In *2023 IEEE International Conference on Quantum Computing and Engineering (QCE)*. IEEE, 2023.

[181] R. A. da Silva and B. Marques. Semidefinite-programming-based optimization of quantum random access codes over noisy channels. *Phys. Rev. A*, 107:042433, 2023.

[182] M. Hayashi, K. Iwama, H. Nishimura, R. Raymond, and S. Yamashita. (4, 1)-quantum random access coding does not exist—one qubit is not enough to recover one of four bits. *New J. Phys.*, 8(8):129, 2006.

[183] R. Kondo, Y. Sato, R. Raymond, and N. Yamamoto. Recursive quantum relaxation for combinatorial optimization problems. *Quantum*, 9:1594, 2025.

Appendix A

Probability Distributions in Tests for RNGs

As an introduction to randomness tests, the present section introduces the definitions and relationships among three probability distributions: the binomial distribution, the normal distribution, and the Chi-squared distribution.

Binomial Distribution The binomial distribution is the ideal distribution of the number of heads in n independent tosses of a coin with probability of heads p. Consider tossing a coin 100 times with the probabilities of heads $p = 0.1, 0.3, 0.5, 0.7$, and 0.9, respectively. The probabilities of the number of heads follow the distributions shown in Fig. A1.

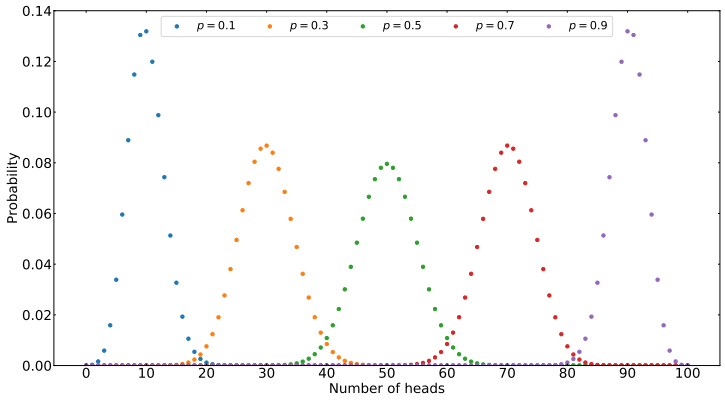


Fig. A1. The probabilities of the number of heads in 100 coin tosses with the probabilities of heads $p = 0.1, 0.3, 0.5, 0.7$, and 0.9.

As can be observed in the figure above, the mean μ and standard deviation σ of the binomial distribution are given by $\mu = np$ and $\sigma = \sqrt{np(1-p)}$. The binomial distribution is often denoted as $B(n, p)$, and is defined by the following equation:

$$B(n, p, k) \equiv \binom{n}{k} p^k (1-p)^{n-k}, \tag{A.1}$$

where k is the number of heads in n coin tosses. Aside from coin tosses, the output of random number generators also follows the binomial distribution. More specifically, the sum of 1s in an n-bit sequence generated by a random number generator whose probability of generating 1 is p follows $B(n, p)$. The frequency test in NIST SP 800-22[6] is based on this feature. Some basic characteristics of the binomial distribution are summarized in Table A.1.

Table A.1. Basic information of the binomial distribution $B(n, p)$.

Mean μ	np.
Standard deviation σ	$\sqrt{np(1-p)}$.
Example	The number of heads k in n coin tosses where the probability of heads is p.

Because computing the combination is time consuming, the binomial distribution is often approximated by the normal distribution. One notable feature of the binomial distribution is its reproductive property.

Reproductive property of the binomial distribution

For independent random variables $X \sim B(n, p)$ and $Y \sim B(m, p_2)$, $X + Y \sim B(n + m, p)$, where "$A \sim B$" stands for "A follows the distribution B".

When n is large and p is small, the binomial distribution is approximated by the Poisson distribution:

Approximation of the binomial distribution by the Poisson distribution

If n is large and p is small for $X \sim B(n, p)$, then $X \sim \frac{(np)^k e^{-np}}{k!}$.

Normal Distribution The normal distribution with mean μ and standard deviation σ is given by the equation

$$N(\mu, \sigma) \equiv \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right).$$

The sum of heads in n coin tosses whose probability of heads is p is approximated by the normal distribution when n is large. The figure below presents the probabilities of the number of heads in 100 coin tosses with the probabilities of heads $p = 0.1, 0.3, 0.5, 0.7$, and 0.9. Shown in scatter plots are the binomial distributions and the continuous plots the approximated normal distributions.

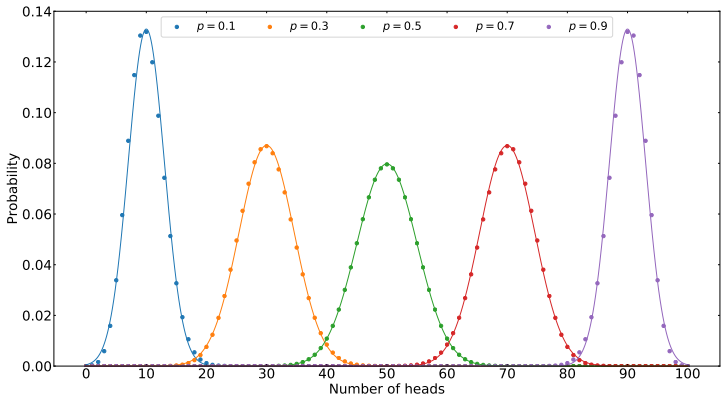


Fig. A2. The probabilities of the number of heads in 100 coin tosses with the probabilities of heads $p = 0.1, 0.3, 0.5, 0.7$, and 0.9 approximated by the normal distribution. The scatter plots represent the binomial distribution and the continuous plots the probabilities approximated by the normal distribution.

Comparing the binomial distributions and the normal distributions in Fig. A2, it is clear that at $n = 100$, the binomial distributions for $p = 0.1, 0.3, 0.5, 0.7$, and 0.9 are well approximated by the normal distributions $N(np, \sqrt{np(1-p)})$. To demonstrate how the accuracy of approximation increases in proportion to

the size of n, the mean absolute difference between the binomial and normal distribution number for trials $n = 10, 20, \ldots, 100$ is plotted in the following figure. Note that the accuracy is defined by $\sum_{k=1}^{n} |B(n,p,k) - N\left(np, \sqrt{np(1-p)}\right)|/n$.

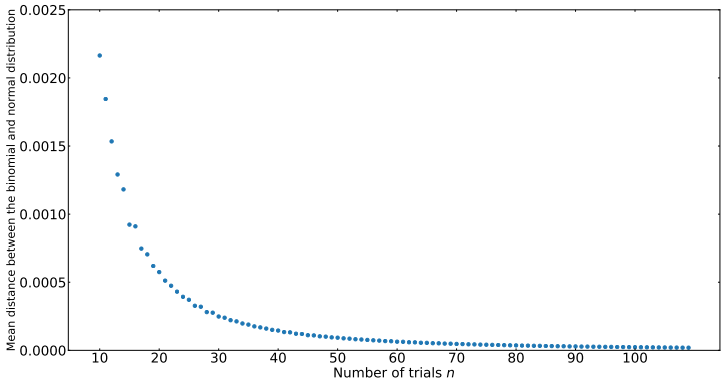


Fig. A3. The mean absolute difference between the binomial distribution $B(n,p)$ and the normal distribution $N(np, \sqrt{np(1-p)})$ for $n = 10, 20, \ldots, 100$.

Observe how the difference converges to 0 as the number of trials n increases. This phenomenon occurs with the sum of any independent and identically distributed variables $X_1, X_2, \ldots, X_n$ regardless of their probability distributions due to the central limit theorem.

Central Limit Theorem

When $X_1, X_2, \ldots, X_n$ are independent and identically distributed variables whose mean and standard deviation are μ and σ, it follows that $\sum_{i=1}^{n} X_i \sim N(n\mu, \sqrt{n}\sigma)$.

It is often convenient to convert the normal distribution $N(\mu, \sigma)$ to the standard normal distribution $N(0,1)$ using the following transformation.

Standardization

If $X \sim N(\mu, \sigma)$, then $\frac{X-\mu}{\sigma} \sim N(0,1)$.

From the central limit theorem and the standardization theorem is derived the following theorem.

Approximation theorem

If $X \sim B(n,p)$, then $\frac{X-np}{\sqrt{np(1-p)}} \sim N(0,1)$.

As was true with the binomial distribution, the normal distribution has a reproductive property.

Reproductive property of the normal distribution

If $X \sim N(\mu_1, \sigma)$ and $Y \sim N(\mu_2, \sigma)$, then $X + Y \sim N(\mu_1 + \mu_2, \sigma)$.

The cumulative distribution function of the normal distribution, denoted by Φ, is given by

$$\Phi(x) \equiv \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} e^{\frac{-t^2}{2}} dt \equiv \frac{1}{2}\left[1 + \operatorname{erf}\left(\frac{x-\mu}{\sigma\sqrt{2}}\right)\right],$$

where

$$\operatorname{erf}(x) \equiv \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} dt.$$

The relationship between the normal distribution and its cumulative distribution function is as follows:

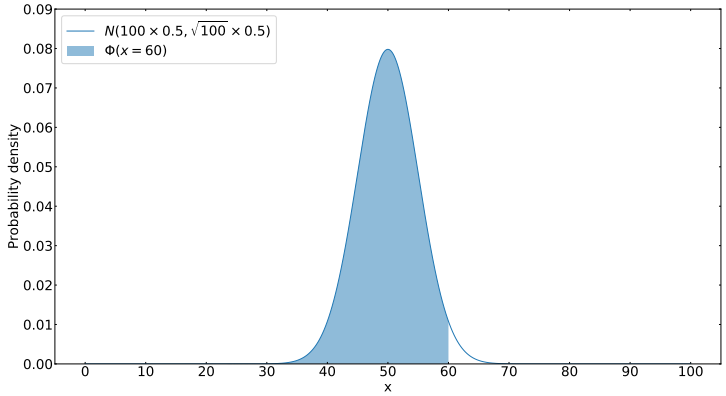


Fig. A4. The relationship between the normal distribution $N(100 \times 0.5, \sqrt{100} \times 0.5)$ and its cumulative distribution function $\Phi(x)$. The area shaded in blue corresponds to the value of the cumulative distribution function $\Phi(x)$ for $x = 60$.

The cumulative distribution function $\Phi(x)$ corresponds to the probability that the value of X is no greater than x.

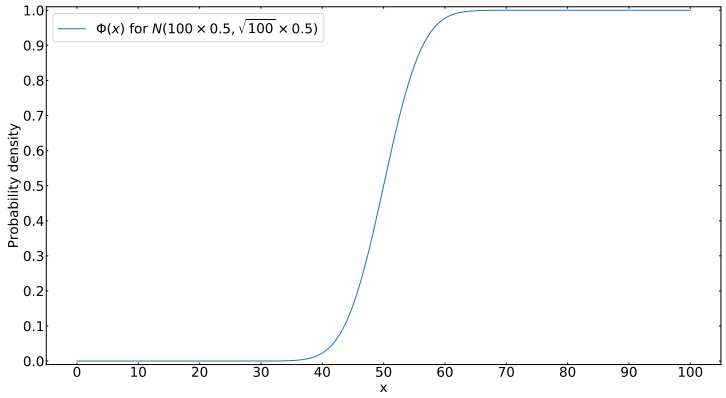


Fig. A5. The cumulative distribution function $\Phi(x)$ corresponding to $N(100 \times 0.5, \sqrt{100} \times 0.5)$.

Because $\Phi(x) \equiv \Pr[X \leq x]$, $\Phi(x)$ takes values from 0 to 1 as in Fig. A5. For the standard normal distribution $N(0, 1)$, the cumulative distribution function is given by

$$\Phi(x) \equiv \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} e^{\frac{-t^2}{2}} dt \equiv \frac{1}{2} \left[1 + \operatorname{erf}\left(\frac{x}{\sqrt{2}} \right) \right].$$

While the cumulative distribution function describes the probability that a normal random variable takes values no greater than x, it is also useful to know the probability that a normal random variable takes values further away from the mean than x. This is called the P-value and is given by the complementary error function

$$\operatorname{erfc}(x) \equiv 1 - \operatorname{erf}(\mathrm{x})$$

as

$$\text{P-value} \equiv \text{erfc}\left(\frac{|x|}{\sqrt{2}}\right).$$

Note that here, the assumption is $x \sim N(0,1)$. Since any random variable following $N(\mu, \sigma)$, the equation above can be applied to any normal distribution. The visual meaning of the P-value is as follows:

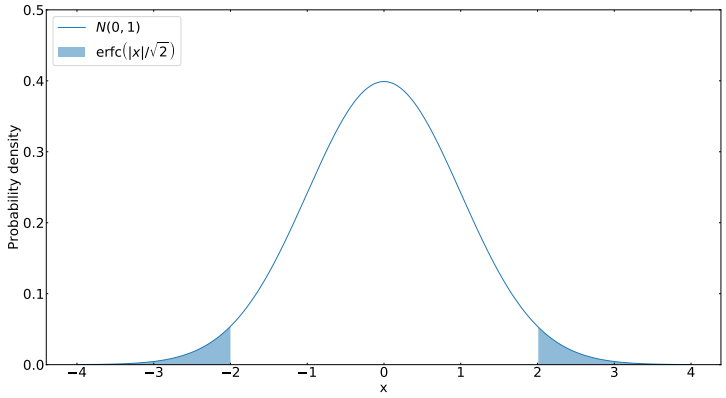


Fig. A6. The visual representation of the two-sided P-value of $x \sim N(0,1)$ for $x = 2$.

The P-value indicates how far a value of x from the mean has been obtained. It takes values ranging from 0 to 1, and a value closer to 1 is more typical because it is closer to the mean.

Chi Squared Distribution Let us consider random variables $X_1, X_2, \ldots, X_k$ that follow the standard normal distribution $N(0,1)$. Then, the distribution of $Z = \sum_{i=1}^{k} X_i$ is called the χ^2 (chi squared) distribution with k degrees of freedom. The probability density function of the χ^2 distribution with k degrees of freedom is

$$f(x,k) \equiv \begin{cases} \frac{1}{2^{k/2}\Gamma(k/2)} x^{k/2-1} e^{-x/2} & \text{for} \quad x > 0, \\ 0 \quad \text{for} \quad x \leq 0. \end{cases}$$

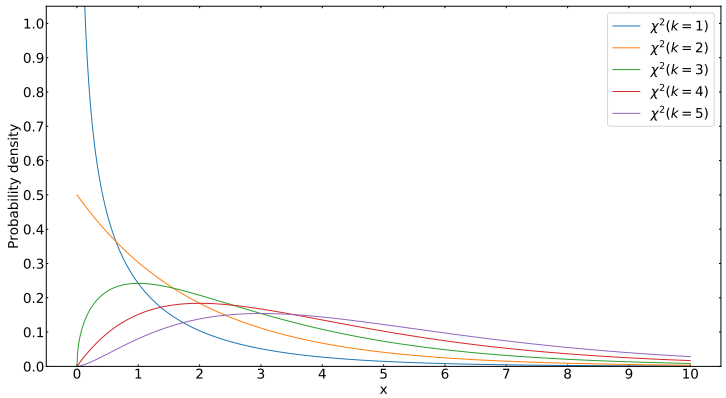


Fig. A7. The χ^2 distribution of $k = 1, 2, 3, 4$, and 5 degrees of freedom.

One of the major applications of the χ^2 distribution is the χ^2 test, which is a statistical test of goodness of fit. This is done by comparing the observed number of data O_i and the expected number of data E_i in each class i. We provide an example of data for the χ^2 test in Table A.2.

Table A.2. Example data for the χ^2 test for 10 classes.

Class i	1	2	3	4	5	6	7	8	9	10
Expected number of data E_i	10	10	10	10	10	10	10	10	10	10
Observed number of data O_i	8	6	13	12	10	5	13	10	13	10

The test statistic is

$$\chi^2(obs) = \sum_{i=1}^{11} \frac{O_i - E_i}{E_i} \sim \chi^2(k = 10 - 1).$$

For the example in Table A.2, the test static is computed as $\chi^2(obs) \approx 7.60$. Now, the probability that a χ^2 test statistic value is obtained under the hypothesis that the observed data has the same distribution as the expected data can be derived as

$$\text{igamc}\left(\frac{10-1}{2}, \frac{\chi^2(obs)}{2}\right),$$

where $\text{igamc}(a, x)$ is an upper incomplete gamma function, defined as

$$\text{igamc}(a, x) \equiv \int_x^\infty t^{a-1} e^{-t} dt. \tag{A.2}$$

The value obtained is called the P-value, which is approximately 0.5749 in the present example. The relationship between the P-value and the χ^2 distribution is represented by the following figure.

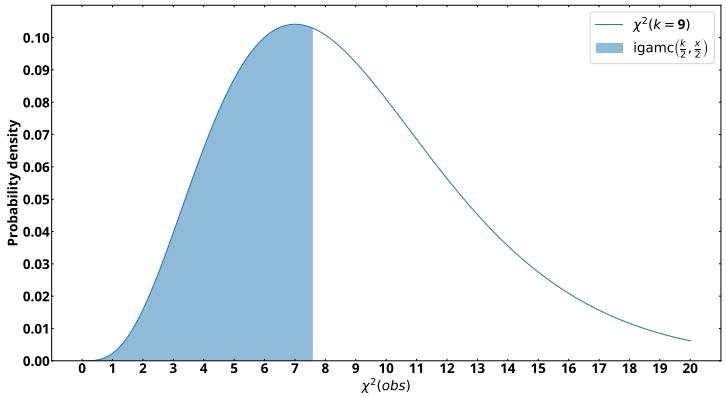


Fig. A8. The meaning of the P-value of the χ^2 test, where the number of classes is 10 and the observed χ^2 test statistic value is 7.60.

The blue area corresponds to the P-value of the χ^2 test, which is equivalent to the cumulative distribution of the χ^2 distribution. The P-value of 0.5749 means that there is a 57.49 % chance that the samples were obtained from the expected distribution.

Appendix B

Stability Analysis of IBM Kawasaki

We present the stability benchmarking result of the IBM Kawasaki device.

Table B.1. The timestamps at the times the job programs were completed by IBM Kawasaki.

Job complete time	Sample number	Job ID
2024-11-05 08:58	0 - 99	cwmwzan31we00088h0fg
2024-11-05 09:05	100 - 199	cwmwzcn40e000089dj0g
2024-11-05 09:11	200 - 299	cwmwze59r49g0089gfs0
2024-11-05 09:21	300 - 399	cwmwzfnmptp000865rb0
2024-11-05 09:25	400 - 499	cwmwzh6mptp000865rd0
2024-11-05 09:33	500 - 599	cwmwzjp0r6b0008q6ry0
2024-11-05 09:38	600 - 699	cwmwzky543p00086xx8g
2024-11-05 09:44	700 - 799	cwmwzn60r6b0008q6ryg
2024-11-05 09:49	800 - 899	cwmwzpp31we00088h0k0
2024-11-05 09:54	900 - 999	cwmwzqy543p00086xx9g

Table B.2. The benchmarking results of IBM Kawasaki.

Qubit Number	Bias Rate	Unstable Rate	Result
0	100.0 %	1.4 %	biased, stable
1	2.1 %	1.0 %	biased, stable
2	1.0 %	1.2 %	unbiased, stable
3	3.2 %	0.4 %	biased, stable
4	9.5 %	1.3 %	biased, stable
5	5.6 %	1.0 %	biased, stable
6	2.2 %	0.6 %	biased, stable
7	5.3 %	0.5 %	biased, stable
8	2.1 %	0.8 %	biased, stable
9	15.9 %	1.2 %	biased, stable
10	6.1 %	0.8 %	biased, stable
11	0.8 %	1.1 %	unbiased, stable
12	100.0 %	0.9 %	biased, stable
13	2.2 %	0.9 %	biased, stable
14	1.5 %	0.5 %	unbiased, stable
15	0.7 %	1.1 %	unbiased, stable

Qubit Number	Bias Rate	Unstable Rate	Result
16	28.3 %	1.1 %	biased, stable
17	4.3 %	0.9 %	biased, stable
18	17.8 %	1.3 %	biased, stable
19	35.6 %	7.2 %	biased, unstable
20	100.0 %	0.9 %	biased, stable
21	8.4 %	1.3 %	biased, stable
22	1.6 %	0.9 %	unbiased, stable
23	1.5 %	0.2 %	unbiased, stable
24	1.5 %	1.7 %	unbiased, stable
25	12.2 %	0.8 %	biased, stable
26	4.5 %	1.5 %	biased, stable
27	28.3 %	0.8 %	biased, stable
28	17.1 %	1.4 %	biased, stable
29	100.0 %	0.5 %	biased, stable
30	2.5 %	1.7 %	biased, stable
31	6.2 %	0.8 %	biased, stable
32	12.5 %	1.1 %	biased, stable
33	19.6 %	0.5 %	biased, stable
34	11.0 %	0.8 %	biased, stable
35	5.1 %	1.2 %	biased, stable
36	61.5 %	1.6 %	biased, stable
37	21.4 %	0.8 %	biased, stable
38	2.0 %	1.3 %	biased, stable
39	23.9 %	1.1 %	biased, stable
40	24.8 %	0.8 %	biased, stable
41	19.2 %	1.1 %	biased, stable
42	16.3 %	1.0 %	biased, stable
43	1.3 %	0.9 %	unbiased, stable
44	7.6 %	1.2 %	biased, stable
45	28.4 %	1.2 %	biased, stable
46	5.0 %	1.2 %	biased, stable
47	17.0 %	0.5 %	biased, stable
48	3.7 %	0.6 %	biased, stable
49	37.6 %	0.9 %	biased, stable
50	2.2 %	0.5 %	biased, stable
51	1.0 %	0.9 %	unbiased, stable
52	2.8 %	2.6 %	biased, unstable
53	5.0 %	0.8 %	biased, stable
54	9.4 %	1.0 %	biased, stable
55	0.7 %	1.1 %	unbiased, stable
56	1.1 %	1.5 %	unbiased, stable
57	1.0 %	1.4 %	unbiased, stable
58	3.7 %	0.8 %	biased, stable
59	24.9 %	0.8 %	biased, stable
60	8.7 %	0.6 %	biased, stable
61	2.1 %	0.7 %	biased, stable
62	2.8 %	0.8 %	biased, stable

Qubit Number	Bias Rate	Unstable Rate	Result
63	6.6 %	1.2 %	biased, stable
64	17.0 %	1.1 %	biased, stable
65	1.4 %	1.6 %	unbiased, stable
66	13.0 %	1.2 %	biased, stable
67	6.8 %	0.6 %	biased, stable
68	2.4 %	0.6 %	biased, stable
69	7.5 %	1.3 %	biased, stable
70	13.2 %	0.4 %	biased, stable
71	1.5 %	1.0 %	unbiased, stable
72	1.4 %	1.1 %	unbiased, stable
73	12.0 %	1.1 %	biased, stable
74	1.1 %	0.8 %	unbiased, stable
75	5.4 %	1.2 %	biased, stable
76	99.0 %	1.5 %	biased, stable
77	76.4 %	0.7 %	biased, stable
78	1.9 %	1.1 %	unbiased, stable
79	21.1 %	0.8 %	biased, stable
80	4.8 %	0.9 %	biased, stable
81	11.6 %	0.9 %	biased, stable
82	7.9 %	0.7 %	biased, stable
83	3.1 %	0.6 %	biased, stable
84	1.5 %	0.8 %	unbiased, stable
85	5.1 %	1.0 %	biased, stable
86	3.2 %	0.9 %	biased, stable
87	1.3 %	0.9 %	unbiased, stable
88	4.6 %	0.6 %	biased, stable
89	3.3 %	1.2 %	biased, stable
90	22.3 %	1.6 %	biased, stable
91	90.1 %	3.4 %	biased, unstable
92	1.8 %	1.1 %	unbiased, stable
93	100.0 %	1.6 %	biased, stable
94	9.4 %	1.1 %	biased, stable
95	3.6 %	2.1 %	biased, unstable
96	91.2 %	0.5 %	biased, stable
97	100.0 %	0.9 %	biased, stable
98	100.0 %	0.0 %	biased, stable
99	79.2 %	1.0 %	biased, stable
100	2.0 %	1.2 %	biased, stable
101	89.9 %	1.6 %	biased, stable
102	4.6 %	1.0 %	biased, stable
103	64.8 %	1.0 %	biased, stable
104	26.3 %	1.1 %	biased, stable
105	16.3 %	4.3 %	biased, unstable
106	30.2 %	1.2 %	biased, stable
107	100.0 %	0.7 %	biased, stable
108	100.0 %	0.9 %	biased, stable
109	6.4 %	0.7 %	biased, stable

Qubit Number	Bias Rate	Unstable Rate	Result
110	100.0 %	12.9 %	biased, unstable
111	3.4 %	1.4 %	biased, stable
112	100.0 %	1.2 %	biased, stable
113	97.2 %	1.5 %	biased, stable
114	2.3 %	1.2 %	biased, stable
115	1.5 %	0.9 %	unbiased, stable
116	97.7 %	0.9 %	biased, stable
117	12.1 %	0.9 %	biased, stable
118	1.7 %	1.2 %	unbiased, stable
119	1.6 %	0.4 %	unbiased, stable
120	3.2 %	1.2 %	biased, stable
121	1.1 %	0.4 %	unbiased, stable
122	3.7 %	1.2 %	biased, stable
123	4.0 %	1.4 %	biased, stable
124	0.9 %	1.0 %	unbiased, stable
125	1.4 %	0.9 %	unbiased, stable
126	100.0 %	0.9 %	biased, stable

Appendix C

Stability Analysis of IBM Fez

We present the stability benchmarking result of the IBM Fez device.

Table C.1. The timestamps at the times the job programs were completed by IBM Fez.

Job complete time	Sample number	Job ID
2024-11-07 15:15	0 - 99	cwp65edehebg008hpehg
2024-11-07 15:21	100 - 199	cwp65gptdtng0086kx00
2024-11-07 15:26	200 - 299	cwp65jp5v39g008gy2fg
2024-11-07 15:33	300 - 399	cwp65netdtng0086kx0g
2024-11-07 17:25	400 - 499	cwp65qp5v39g008gy2hg
2024-11-07 17:34	500 - 599	cwp65t7997wg008x5980
2024-11-08 08:39	600 - 699	cwp65wfehebg008hpen0
2024-11-08 09:14	700 - 799	cwp65yqtdtng0086kx2g
2024-11-08 09:26	800 - 899	cwp660r997wg008x599g
2024-11-09 05:59	900 - 999	cwp662r2ac5g008hm9b0

Table C.2. The benchmarking results of IBM Fez.

Qubit Number	Bias Rate	Unstable Rate	Result
0	28.9 %	1.0 %	biased, stable
1	14.0 %	0.8 %	biased, stable
2	13.7 %	0.7 %	biased, stable
3	71.4 %	0.8 %	biased, stable
4	14.5 %	1.2 %	biased, stable
5	3.2 %	2.1 %	biased, unstable
6	12.4 %	2.9 %	biased, unstable
7	3.8 %	0.9 %	biased, stable
8	100.0 %	0.7 %	biased, stable
9	5.9 %	0.9 %	biased, stable
10	7.7 %	0.5 %	biased, stable
11	26.7 %	0.5 %	biased, stable
12	27.9 %	1.2 %	biased, stable
13	18.1 %	1.2 %	biased, stable
14	9.9 %	0.7 %	biased, stable
15	27.7 %	1.3 %	biased, stable

Qubit Number	Bias Rate	Unstable Rate	Result
16	7.9 %	1.2 %	biased, stable
17	22.8 %	0.5 %	biased, stable
18	100.0 %	0.5 %	biased, stable
19	88.6 %	1.2 %	biased, stable
20	100.0 %	0.4 %	biased, stable
21	48.7 %	0.8 %	biased, stable
22	86.7 %	0.9 %	biased, stable
23	18.8 %	0.9 %	biased, stable
24	17.2 %	1.5 %	biased, stable
25	88.2 %	1.3 %	biased, stable
26	96.4 %	0.4 %	biased, stable
27	28.6 %	1.1 %	biased, stable
28	100.0 %	0.5 %	biased, stable
29	1.8 %	1.4 %	unbiased, stable
30	3.2 %	0.9 %	biased, stable
31	36.9 %	1.1 %	biased, stable
32	1.2 %	0.8 %	unbiased, stable
33	0.8 %	1.2 %	unbiased, stable
34	30.2 %	1.0 %	biased, stable
35	91.5 %	1.3 %	biased, stable
36	100.0 %	1.0 %	biased, stable
37	100.0 %	2.5 %	biased, unstable
38	75.7 %	0.9 %	biased, stable
39	98.4 %	1.0 %	biased, stable
40	2.0 %	0.8 %	biased, stable
41	90.0 %	15.9 %	biased, unstable
42	88.7 %	1.2 %	biased, stable
43	89.4 %	1.8 %	biased, stable
44	1.6 %	0.8 %	unbiased, stable
45	59.7 %	1.3 %	biased, stable
46	2.3 %	1.1 %	biased, stable
47	31.8 %	0.6 %	biased, stable
48	4.4 %	1.4 %	biased, stable
49	8.8 %	1.1 %	biased, stable
50	100.0 %	0.8 %	biased, stable
51	99.9 %	0.8 %	biased, stable
52	99.0 %	1.0 %	biased, stable
53	1.8 %	1.1 %	unbiased, stable
54	44.1 %	1.2 %	biased, stable
55	100.0 %	2.2 %	biased, unstable
56	5.0 %	1.3 %	biased, stable
57	27.7 %	1.2 %	biased, stable
58	52.9 %	0.7 %	biased, stable
59	100.0 %	0.8 %	biased, stable
60	100.0 %	0.9 %	biased, stable
61	55.8 %	1.1 %	biased, stable
62	6.1 %	0.8 %	biased, stable

Qubit Number	Bias Rate	Unstable Rate	Result
63	98.1 %	0.9 %	biased, stable
64	2.8 %	0.5 %	biased, stable
65	30.9 %	1.2 %	biased, stable
66	97.6 %	1.0 %	biased, stable
67	88.6 %	1.1 %	biased, stable
68	4.6 %	1.0 %	biased, stable
69	91.8 %	0.9 %	biased, stable
70	100.0 %	0.6 %	biased, stable
71	42.5 %	0.7 %	biased, stable
72	5.4 %	0.8 %	biased, stable
73	26.7 %	1.4 %	biased, stable
74	99.9 %	0.8 %	biased, stable
75	19.9 %	1.3 %	biased, stable
76	91.7 %	0.6 %	biased, stable
77	99.2 %	0.7 %	biased, stable
78	85.1 %	0.4 %	biased, stable
79	91.0 %	0.7 %	biased, stable
80	54.6 %	0.6 %	biased, stable
81	4.0 %	1.5 %	biased, stable
82	11.7 %	0.9 %	biased, stable
83	20.3 %	1.7 %	biased, stable
84	51.8 %	1.1 %	biased, stable
85	98.0 %	0.6 %	biased, stable
86	98.5 %	10.6 %	biased, unstable
87	47.6 %	1.0 %	biased, stable
88	99.9 %	1.3 %	biased, stable
89	99.7 %	1.3 %	biased, stable
90	1.9 %	0.9 %	unbiased, stable
91	12.9 %	0.7 %	biased, stable
92	1.5 %	0.8 %	unbiased, stable
93	100.0 %	1.9 %	biased, stable
94	69.0 %	0.9 %	biased, stable
95	20.4 %	1.0 %	biased, stable
96	92.8 %	1.4 %	biased, stable
97	76.6 %	0.9 %	biased, stable
98	46.8 %	1.1 %	biased, stable
99	96.4 %	0.9 %	biased, stable
100	37.4 %	1.3 %	biased, stable
101	64.4 %	1.3 %	biased, stable
102	9.2 %	1.7 %	biased, stable
103	1.6 %	0.8 %	unbiased, stable
104	99.5 %	0.4 %	biased, stable
105	1.0 %	1.0 %	unbiased, stable
106	100.0 %	0.9 %	biased, stable
107	1.2 %	0.9 %	unbiased, stable
108	100.0 %	0.5 %	biased, stable
109	67.5 %	1.3 %	biased, stable

Qubit Number	Bias Rate	Unstable Rate	Result
110	78.1 %	0.8 %	biased, stable
111	32.5 %	1.3 %	biased, stable
112	99.6 %	0.3 %	biased, stable
113	66.9 %	1.3 %	biased, stable
114	85.0 %	0.7 %	biased, stable
115	0.8 %	1.2 %	unbiased, stable
116	9.7 %	1.4 %	biased, stable
117	98.8 %	1.7 %	biased, stable
118	100.0 %	0.5 %	biased, stable
119	100.0 %	0.9 %	biased, stable
120	100.0 %	0.8 %	biased, stable
121	63.2 %	1.3 %	biased, stable
122	100.0 %	1.0 %	biased, stable
123	10.0 %	0.9 %	biased, stable
124	1.7 %	1.1 %	unbiased, stable
125	7.0 %	0.7 %	biased, stable
126	99.5 %	0.8 %	biased, stable
127	74.1 %	1.5 %	biased, stable
128	5.4 %	0.9 %	biased, stable
129	11.8 %	0.6 %	biased, stable
130	98.7 %	0.7 %	biased, stable
131	2.6 %	1.3 %	biased, stable
132	100.0 %	1.1 %	biased, stable
133	3.4 %	0.7 %	biased, stable
134	89.0 %	0.5 %	biased, stable
135	16.2 %	1.5 %	biased, stable
136	3.1 %	1.5 %	biased, stable
137	7.5 %	1.2 %	biased, stable
138	3.2 %	1.0 %	biased, stable
139	100.0 %	0.6 %	biased, stable
140	22.5 %	0.9 %	biased, stable
141	5.5 %	1.2 %	biased, stable
142	98.4 %	0.6 %	biased, stable
143	83.6 %	0.8 %	biased, stable
144	48.7 %	1.0 %	biased, stable
145	46.8 %	0.9 %	biased, stable
146	86.2 %	0.9 %	biased, stable
147	57.2 %	0.9 %	biased, stable
148	24.7 %	0.6 %	biased, stable
149	73.1 %	1.0 %	biased, stable
150	99.1 %	1.1 %	biased, stable
151	99.7 %	0.8 %	biased, stable
152	100.0 %	0.4 %	biased, stable
153	24.3 %	1.2 %	biased, stable
154	100.0 %	0.7 %	biased, stable
155	54.3 %	1.0 %	biased, stable

Appendix D

Derivation of The Order of Shots

Here, we outline the derivation of Eq. (4.59)~ (4.62). Pauli rounding is achieved by applying the appropriate rotations to the parameterized circuit to estimate the expectation value corresponding to each respective node. To estimate the expectation value with respect to the Pauli matrix X, the Hadamard gate is applied, followed by measurement in the computational basis. For the Pauli matrix Y, the Hadamard gate and the phase gate are applied before the measurement in the computational basis. For the Pauli matrix Z, no rotation is necessary.

Given a $|V|$-node graph with the Pauli matrix P_j assigned to the j-th node, we obtain a series of measurement results across S measurements for the Pauli matrices $P_1 \sim P_m$. For each node, the trace value is given by

$$\begin{aligned}\mathrm{Tr}(P_j\rho) &= 2\left(1 - \frac{\sum_{i=1}^{S} X_{ij}}{S}\right) - 1 \\ &= 1 - 2\frac{\sum_{i=1}^{S} X_{ij}}{S} \\ &= 1 - 2\frac{h_1^j}{S}.\end{aligned}$$

Now, we assume $\Pr[X_{i,j} = 1] = \frac{1}{2} + \varepsilon$, where $\Pr[X_{i,j} = 1]$ refers to the expectation value of the measurement result $X_{i,j}$, and $\varepsilon > 0$. The sign of the trace value is incorrectly estimated when $h_1^j = \sum_{i=1}^{S} X_{ij} \leq S/2$. We then utilize the Chernoff-Hoeffding bound which states that for independent random variables $X_{1j}, X_{2j}, \ldots, X_{Sj}$ in $\{0, 1\}$ with $\Pr[X_{i,j} = 1] = p_j$, $h_1^j = \sum_{i=1}^{S} X_{ij}$, and $\mu_j = E[h_1^j] = Sp_j$, the following holds for any $\lambda > 0$:

$$\Pr[h_1^j \leq \mu_j - \lambda] \leq \exp\left(-\frac{2\lambda^2}{S}\right).$$

Substituting μ with $S(1/2 + \varepsilon)$ and λ with εS results in the following:

$$\begin{aligned}\Pr\left[h_1^j \leq \mu_j - \lambda\right] &= \Pr\left[h_1^j \leq S\left(\frac{1}{2} + \varepsilon\right) - \varepsilon S\right] \\ &= \Pr\left[h_1^j \leq \frac{S}{2}\right] \leq \exp(-2S\varepsilon^2).\end{aligned}$$

The probability δ of estimating the sign of the trace value incorrectly is denoted

$$\Pr\left[h_1^j \leq \frac{S}{2}\right] \leq \exp(-2S\varepsilon^2) \leq \delta.$$

We therefore obtain

$$S \geq \frac{\ln(1/\delta)}{2\varepsilon^2}.$$

By using the fact that the probability of correctly estimating the sign of the trace values of all nodes $(1-\delta)^{|V|}$ is close to 1, we can apply Bernoulli's inequality to obtain

$$\alpha \leq (1-\delta)^{|V|} \leq \exp(-\delta|V|),$$

hence,

$$\delta \leq -\ln(\alpha)/|V|$$

Finally, we obtain

$$S = O\left(\frac{\ln(|V|)}{\varepsilon^2}\right)$$

for a single node, and

$$O\left(\frac{|V|\ln(|V|)}{\varepsilon^2}\right)$$

for m nodes. The trace value under the influence of depolarizing noise can be denoted $\mathrm{Tr}(P_j \mathcal{D}_p^N(\rho)) = p^N \mathrm{Tr}(P_j\rho)$. We therefore have $\varepsilon = -p^N \mathrm{Tr}(P_j\rho)/2$. Note that the trace value is assumed to be negative.