UNIVERSITÀ DEGLI STUDI DI PADOVA

Dipartimento di Fisica e Astronomia “Galileo Galilei”

Corso di Laurea in Fisica

Tesi di Laurea

# n-Qubit Repetition Code sull’Hardware Quantistico di IBM

# n-Qubit Repetition Code on IBM Quantum Hardware

Relatore

Dr. Ilaria Siloi

Correlatore

Prof. Simone Montangero

Laureando

Giovanni Bartolucci

Anno Accademico 2022/2023

# Abstract

The current implementation of qubits in noisy quantum hardware is imperfect, leading to non-unitary system evolutions due to interactions with the surrounding environment. Therefore, executing quantum algorithms on existing quantum platforms is prone to errors, potentially resulting in computation failures. The development of efficient protocols for error correction is essential to advance research toward universal fault-tolerant quantum computers. This field of study is known as Quantum Error Correction (QEC). In this thesis, we provide an introduction to the fundamentals of QEC and we test the Repetition code, a simple QEC algorithm, using IBM Quantum Hardware. The Repetition code introduces redundancy in the stored information by utilizing n-qubits to represent the state of a single qubit. By comparing results obtained from IBM quantum processors with simulations, we identify a range of qubit error rates where the analyzed codes significantly enhance the probability of a successful computation.

## Sommario

L'attuale implementazione dei qubit nell'hardware quantistico affetto da rumore è imperfetta, portando a evoluzioni non unitarie del sistema dovute alle interazioni con l'ambiente circostante. Pertanto, l'esecuzione di algoritmi quantistici sulle piattaforme quantistiche esistenti è soggetta a errori, con conseguenti potenziali fallimenti delle computazioni. Lo sviluppo di protocolli efficienti per la correzione degli errori è essenziale per far progredire la ricerca verso computer quantistici universali e "Fault Tolerant". Questo campo di studio è noto come Quantum Error Correction (QEC). In questa tesi, forniamo un'introduzione ai fondamenti di QEC e testiamo il Repetition code, un semplice algoritmo di QEC, utilizzando l'Hardware Quantistico di IBM. Il Repetition code introduce ridondanza nell'informazione salvata utilizzando n-qubit per rappresentare lo stato di un singolo qubit. Confrontando i risultati ottenuti dai processori quantistici IBM con le simulazioni, identifichiamo un range di tassi di errore sui qubit entro cui i codici analizzati aumentano significativamente la probabilità di successo della computazione.

# Index

# Introduction

Numerous recent scientific breakthroughs could not have been possible without the advent of (super)computers, capable of simulating our theories and processing vast amounts of data. However, from the computational complexity theory, several problems (non-deterministic polynomial) require resources in terms of computing time and memory that render their solutions intractable or inefficient on classical computers. Among these challenges lies the simulation of quantum systems. The concept of a quantum computer dates back over four decades to the visionary thinking of Richard Feynman, who proposed the use of quantum systems for simulating the fundamental constituents of reality [1].
Subsequently, this idea has inspired the development of native quantum algorithms that harness the exquisite properties of quantum mechanics, such as superposition and entanglement [2, 3], to accelerate computational processes [4, 5]. One well-known example is Peter Shor's 1994 algorithm which can factor prime numbers with an exponential gain over its best-discovered classical counterpart [6]. Other algorithms that offer a speed-up over their classical analogues are Lov Grover's 1996 algorithm [7], for the search of a given item in an unstructured database, and Quantum Eigensolver [8], capable of finding the eigenstates and eigenvalues of a given Hamiltonian.
However, the current hardware implementations, so-called NISQ (Noisy Intermediate-Scale Quantum) devices [9], are limited in terms of the number of qubits and error-prone. All types of quantum platforms exhibit a high sensitivity to various sources of noise, resulting in the degradation of their quantum properties. Therefore, it is imperative to establish an efficient way to mitigate and correct errors. One acknowledged strategy is correcting the errors during the running time by implementing specific quantum algorithms, the so-called Quantum Error Correction (QEC) codes. Some examples are Repetition, Steane's, 9-qubit Shor's and other stabilizer codes [10].
To pursue practical applications for NISQ devices and the advancement of research toward a fault-tolerant quantum computer [11], it is paramount to expand investigations within the realm of Quantum Error Correction (QEC). This thesis aligns itself with this goal, serving as an introductory exploration into the domain of QEC. Once defined a broad class of quantum error-correcting algorithms, the stabilizer codes, this work proceeds to examine selected codes and their implementations on real quantum hardware and simulation platforms. Our findings not only contribute to our understanding of code performance but also enable us to extrapolate a range of acceptable error rates for the successful execution of these codes on quantum hardware.
This thesis is organized as follows: the first chapter revisits fundamental concepts pertaining to Quantum Computing and Information Theory. In Chapter 2, we provide an introduction to Quantum Error Correction, elucidating key algorithms and the stabilizer formalism. In the final chapter, we detail the results we derived from testing Repetition and Steane stabilizer codes, executed on IBM's available quantum computing platforms as well as classical simulators.

# Chapter 1

# Fundamentals of Quantum Computing and Information Theory

Quantum computing is a new paradigm of computation that employs the rules of quantum mechanics to process information. The fundamental unit of classical computation is the bit, a binary variable. All classical computations can be decomposed in logical operations with bits. On the other hand, in quantum information theory [2] the fundamental unit of information is the qubit. From a physical point of view, the qubit can be any two-level quantum system: quantum mechanics permits such systems to be in one of the two levels or a superposition of the two [2, 3]. In the following work we describe the qubit as a general two-level quantum system independently from its physical implementation.
In this chapter, we introduce the basic concepts of quantum computation: the qubit, quantum states and their evolution, entanglement, projective measurements, quantum registers, gates, channels and quantum circuits.

## 1.1 The qubit

A qubit is defined in a 2-dimensional Hilbert space [2, 12]. Usually, the preferred basis for this space is the so-called "Computational Basis" or $\sigma_z$-basis:

$$\left\{ |0\rangle \equiv \begin{pmatrix} 1 \\ 0 \end{pmatrix} \ , \quad |1\rangle \equiv \begin{pmatrix} 0 \\ 1 \end{pmatrix} \right\}. \tag{1.1}$$

The general state of a qubit is a complex linear combination of the two basis states:

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle \qquad \alpha, \beta \in \mathbb{C}, \text{ where } \mathbb{C} \text{ is the the set of complex numbers.} \tag{1.2}$$

The superposition principle consists of the ability of the qubit to occupy distinct states at the same time (here $|0\rangle$ and $|1\rangle$) with a certain probability of being found in one or the other. The information encoded in the state, i.e. the $\alpha$, $\beta$ coefficients, cannot be extracted with a single measurement because doing so collapses the qubit in one of the eigenstates of the measurement operator. For example, measuring along $\sigma_z$ one can obtain only two results: "+1" (eigenvalue corresponding to $|0\rangle$) with probability $|\alpha|^2$ or "-1" (corresponding to $|1\rangle$) with probability $|\beta|^2$. So the squared modulus of each coefficient represent the probabilities, which should always be normalized to 1:

$$|\alpha|^2 + |\beta|^2 = 1.$$

This is in contrast with classical bits which are always certainly either 0 or 1.
It is important to note that the superposition is disrupted when observed (i.e. the qubit is measured)

[3], but it can be exploited during the computations in a way that is like exploring different paths at once. In addition to the normalization condition of the coefficients, a quantum state is required to be independent of global phases ($|\psi\rangle$ and $e^{i\phi}|\psi\rangle$ must represent the same state). To make these properties explicit it is useful to rewrite Equation (1.2) in the so-called geometrical representation:

$$|\psi\rangle = \cos\left(\frac{\theta}{2}\right)|0\rangle + \sin\left(\frac{\theta}{2}\right)e^{i\phi}|1\rangle \qquad 0 \leq \theta \leq \pi \quad 0 \leq \phi \leq 2\pi. \tag{1.3}$$

This way a qubit can be visualized as a vector pointing on the surface of a three-dimensional unitary sphere (see Fig.(1.1)), uniquely defined by the angles $\theta$ and $\phi$.

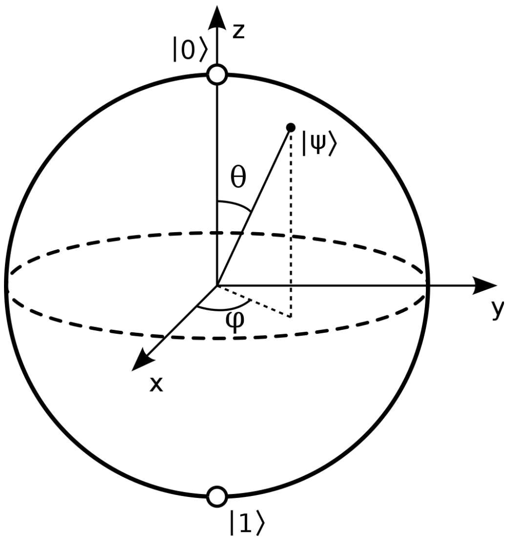

Figure 1.1: *A visual representation of the Bloch sphere,* with the generic qubit pure state $|\psi\rangle$, the basis states $|0\rangle \equiv +\hat{z}$ and $|1\rangle \equiv -\hat{z}$. Other notable states are $|+\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \equiv +\hat{x}$, $|-\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) \equiv -\hat{x}$; while the $\pm\hat{y}$ unitary vectors represent the states $\frac{1}{\sqrt{2}}(|0\rangle \pm i|1\rangle)$.

So, one qubit can encode the information of two real numbers (the angles $\theta$ and $\phi$) with arbitrary precision, which is a resource while executing the algorithms but can never be measured or directly examined. To retrieve it a process called Quantum Tomography is needed. It consists of a series of measurements on a set of identically prepared copies of the system to be probed, as many more as the result is wanted to be precise. Therefore, the amount of information that can be extracted from the system is proportional to the number of measurements made, which offers no advantage over saving the same numbers in memory with many bits. Unlike the classical realm though, here the superposition can be exploited to efficiently parallelize the computation and quantum entanglement opens the door to completely new operations [2].

## 1.2 Quantum states as density matrices

In quantum mechanics, a quantum state provides the mathematical description of a physical object. The mathematical object, called wave function, incorporates the knowledge of the corresponding quantum system. It specifies its creation, evolution, and possible measurement results with respective probabilities of any measurable quantity. A state that provides the maximum possible information of a system is called pure, otherwise, if some characteristic of the system is unknown, it is called mixed. A pure state is defined by a single wave function that for a qubit takes the general form in Eq.(1.2). A mixed state generalizes this concept to an ensemble of wave functions, each one having a (classical) probability of describing the system under consideration. To characterize pure and mixed states with a formalism that makes their respective properties explicit, it is useful to resort to the density matrix [13]. Density matrices are positive semidefinite operators acting on Hilbert spaces. The most general definition of a quantum state via density matrix is given by:

$$\rho = \sum_{i=1}^{n} p_i |\psi_i\rangle\langle\psi_i| \,, \tag{1.4}$$

with $p_i$ the probability that the system is in the state determined by the wave function $\psi_i$. Some of its properties are:

- $\rho$ is Hermitian.
- $\rho$ is a non-negative operator: $\langle\phi|\rho|\phi\rangle \geq 0 \quad \forall|\phi\rangle \in \mathcal{H}$.
- $Tr(\rho) \equiv \sum_{k=1}^{n} \langle k|\rho|k\rangle = 1,$ where $\{|k\rangle\}_{k=1...n}$ is a basis of the $n$-dimensional Hilbert space $\mathcal{H}$ and Tr() is the trace operation.
- The expectation value of any observable $\hat{O}$ in the state $\rho$ is given by:

$$\langle\hat{O}\rangle_\rho \equiv Tr(\rho\hat{O}) = \sum_{k=1}^{n} \langle k|\rho\hat{O}|k\rangle. \tag{1.5}$$

Density matrix formalism is useful for discerning pure states from mixed states:

- If $\rho$ is pure then: $\rho_p = |\psi\rangle\langle\psi|$ so there is only one $p_i = 1$ and every other $p_j = 0 \quad \forall j \neq i$. $\rho_p$ is a projector: $\rho_p^2 = \rho_p,$ so $Tr(\rho_p^2) = 1$.
- If $\rho$ is mixed then it has at least two different $p_i \neq 0$ and $Tr(\rho_m^2) < 1$.

## 1.3 Composite quantum systems and entanglement

After having introduced the description of a single qubit, we turn to systems composed of many distinguishable parts which are useful in quantum computing, for instance, in the definition of quantum registers. A very important emerging property of many-body quantum systems is entanglement, a phenomenon with no classical analogue that has noteworthy applications in this field. A generic $n$-body quantum system is defined in the tensor product space of every single Hilbert space in which each constituent lives [12]:

$$\mathcal{H} = \bigotimes_{i=1}^{n} \mathcal{H}_i = \mathcal{H}_1 \otimes \cdots \otimes \mathcal{H}_n. \tag{1.6}$$

$\mathcal{H}$ is itself a Hilbert space and, in it, the most general pure state can be expressed as:

$$|\psi\rangle = \sum_{\vec{k}} c_{\vec{k}} |k\rangle_1 \otimes \cdots \otimes |k\rangle_n = \sum_{\vec{k}}^{dim\mathcal{H}} c_{\vec{k}} |k_1...k_n\rangle \qquad with \quad \sum_{\vec{k}} |c_{\vec{k}}|^2 = 1, \tag{1.7}$$

and $\{|k\rangle_i\}_{i=1...n}$ being the orthonormal basis of each Hilbert space $\{\mathcal{H}_i\}_{i=1...n}$, in general each having different dimension. The composite system will have density matrix:

$$\rho = \sum_{\vec{k}} \sum_{\vec{m}} c_{\vec{k}} c_{\vec{m}}^* |k_1...k_n\rangle\langle m_1...m_n|. \tag{1.8}$$

An observable $\hat{O}_i$ that lives in the i-th Hilbert space $\mathcal{H}_i$ can be generalized as an observable $\hat{O}$ that lives in the Hilbert space $\mathcal{H}$ but only acts on $\mathcal{H}_i$:

$$\hat{O} = \mathbb{1}_1 \otimes \cdots \otimes \hat{O}_i \otimes \cdots \otimes \mathbb{1}_n. \tag{1.9}$$

Its expectation value is easily calculated once the concept of reduced density matrix [3] is introduced:

$$\rho_i = Tr_{j\neq i}(\rho). \tag{1.10}$$

From which:

$$\langle \hat{O}\rangle_\rho = Tr(\rho\hat{O}) = Tr(\rho_i\hat{O}_i). \tag{1.11}$$

Which is true for any observable acting on one subsystem. This means that $\rho_i$, derived from partial tracing the $n$-body density matrix over the other subsystems, describes its corresponding subsystem [2]. It cannot be overlooked that reducing a density matrix does not always preserve its purity, i.e. if $\rho$ is that of a pure state it does not imply that $\rho_i$ is pure too.

### 1.3.1 Entanglement

We overview the phenomenon for two objects, A and B, in any pure state. Suppose they have never interacted before. Consider them to be respectively in state $|\phi\rangle_A$, defined in the Hilbert space $\mathcal{H}_A$, and $|\chi\rangle_B$, defined in $\mathcal{H}_B$. The composite system will be in state:

$$|\psi\rangle_{AB}^{sep} = |\phi\rangle_A \otimes |\chi\rangle_B, \qquad |\psi\rangle_{AB}^{sep} \in \mathcal{H} \equiv \mathcal{H}_A \otimes \mathcal{H}_B. \tag{1.12}$$

Every state that can be written as $|\psi\rangle_{AB}^{sep}$ is called a separable or product state. Suppose A and B have an interaction. Now, the composite system has some state in $\mathcal{H}$, the most general of which is:

$$|\psi\rangle_{AB}^{gen} = \sum_{i,j} c_{ij}|i\rangle_A \otimes |j\rangle_B. \tag{1.13}$$

Where $\{|i\rangle_A\}_{i=1...dim\mathcal{H}_A}$ is a basis for $\mathcal{H}_A$ and $\{|j\rangle_B\}_{j=1...dim\mathcal{H}_B}$ for $\mathcal{H}_B$.
Not every state in $\mathcal{H}$ is separable. This can be figured out considering the density matrix of the composite system, $\rho_{AB}^{gen}$, that generalizes also to mixed states. From $\rho_{AB}^{gen}$ one can calculate the reduced density matrix of one of the two subsystems, e.g.: $\rho_A = Tr_B(\rho_{AB}^{gen})$.

- If $\rho_A$ is pure, i.e. $Tr(\rho_A^2) = 1$, then $|\psi\rangle_{AB}^{gen} = |\psi\rangle_{AB}^{sep}$: the state is separable.
- Otherwise, if $\rho_A$ is mixed, i.e. $Tr(\rho_A^2) < 1$, then $|\psi\rangle_{AB}^{gen} \neq |\psi\rangle_{AB}^{sep}$: the state is entangled.

An example of an entangled system is the two-qubit Bell state:

$$|\Phi^+\rangle = \frac{1}{\sqrt{2}}(|0\rangle_A \otimes |0\rangle_B + |1\rangle_A \otimes |1\rangle_B) \equiv \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)_{AB}. \tag{1.14}$$

A peculiarity of this state is that, by measuring only one of the two qubits, one already knows that the other will provide the same result even before measuring it. In general, when two objects are entangled measuring one of them immediately provides information on the other one, even if they are space-like separated [2].

### 1.3.2 Quantum registers

In analogy to classical computation, where registers of $n$ bits are manipulated, quantum computation performs logical operations on registers of $n$ qubits. A $n$-qubit quantum register is nothing more than a $n$-body quantum system where each component lives in the single-qubit Hilbert space. So the Hilbert space of a quantum register is $2^n$-dimensional. In it, the state of a quantum computer at a given time can be written, in binary notation, as:

$$|\psi\rangle = \sum_{i_{n-1}=0}^{1} \cdots \sum_{i_0=0}^{1} c_{i_{n-1}...i_0}|i_{n-1}\rangle \otimes \cdots \otimes |i_0\rangle \qquad with \qquad \sum_{i_{n-1}=0}^{1} \cdots \sum_{i_0=0}^{1} |c_{i_{n-1}...i_0}|^2 = 1. \tag{1.15}$$

While in decimal notation as:

$$|\psi\rangle = \sum_{i=0}^{2^n-1} c_i|i\rangle \qquad \textit{with} \qquad \sum_{i=0}^{2^n-1} |c_i|^2 = 1. \tag{1.16}$$

Where, in the last expression, '$i$' is the integer in decimal base obtained by converting the binary digits '$i_{n-1}...i_0$'. A classical $n$-bit register can save only one of the $2^n$ possible combinations of $n$ binary digits, i.e. only one integer '$i$' between 0 and $2^n - 1$. Whereas, a $n$-qubit register can also hold in memory a superposition of all the $2^n$ possible integers. A classical computer needs an execution of the program for each input to be evaluated, while a quantum computer can explore $2^n$ inputs during a single run. Hence, with the same register size, a quantum computer can potentially parallelize its operations on an exponentially greater number of inputs than a classical one [2].

## 1.4 Unitary evolution

Every quantum (pure) state evolves in time according to Schrödinger's equation [12]:

$$i\hbar\frac{\partial|\psi\rangle}{\partial t} = \hat{H}|\psi\rangle. \tag{1.17}$$

Where it was assumed that $|\psi\rangle$ is the state of an isolated system, meaning non-interacting with the surrounding environment, with $\hat{H}$ being the hamiltonian operator of that system. If $\hat{H}$ is time-independent and $t_0 = 0$, then the solution to Schrödinger's equation (1.17) is:

$$|\psi(t)\rangle = U(t)|\psi(0)\rangle, \tag{1.18}$$

where $U(t)$ is the evolution unitary operator [14]. One can notice that any quantum evolution that conserves quantum properties must be unitary.
The above results can be readily generalized to the evolution of the density matrix:

$$\rho(t) = \sum_i p_i|\psi_i(t)\rangle\langle\psi_i(t)| = \sum_i p_i U(t)|\psi_i(0)\rangle\langle\psi_i(0)|U^\dagger(t) = U(t)\rho_0 U^\dagger(t). \tag{1.19}$$

Evolution can be generalized to non-unitary operators, which we will cover in Section (1.6).

### 1.4.1 Quantum gates

Just as in classical computation, in quantum computing the state of the quantum register is manipulated to obtain any of the possible states. Operations on qubits, called quantum gates, are constituted by unitary matrices. Unitarity ensures that the operations carried out are reversible and perfectly describable within the formalism of quantum mechanics. A generic unitary evolution, i.e. any quantum algorithm, can be decomposed in a universal set of quantum gates [2, 3], analogously to the classical case. In this regard, it has been demonstrated that any unitary evolution in the $n$-qubit Hilbert space can be broken down into a finite sequence of one-qubit gates plus a two-qubit gate such as CNOT [2]. We report here the more common gates in the computational basis (1.1):

- BUFFER $\mathbb{1} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$ Also called idle gate, is just an identity.
  It leaves unchanged the basis states, so its output is the starting state: $\mathbb{1}|\psi\rangle = |\psi\rangle$.

- NOT GATE $\sigma_x = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$ Not gate swaps the basis states with each other:

$$\sigma_x|0\rangle = |1\rangle \quad \textit{and} \quad \sigma_x|1\rangle = |0\rangle. \tag{1.20}$$

  Buffer and not gates act just like their classical analogues.

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

  It is used to generate superpositions between the basis state, like:

$$H|0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) = |+\rangle \quad and \quad H|1\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) = |-\rangle. \tag{1.21}$$

  Its action can be pictured as a $\frac{\pi}{2}$ rotation on the Bloch sphere around the y-axis.

- PHASE-SHIFT GATE $\quad R_z(\delta) = \begin{pmatrix}1 & 0\\ 0 & e^{i\delta}\end{pmatrix}$

  It adds a relative phase $\delta$ on a single qubit (written as in Eq.(1.3)), like:

$$R_z(\delta)|\psi\rangle = \cos\left(\frac{\theta}{2}\right)|0\rangle + \sin\left(\frac{\theta}{2}\right)e^{i(\phi+\delta)}|1\rangle. \tag{1.22}$$

  Its action can be pictured as a $\delta$ rotation on the Bloch sphere around the z-axis.
  It is possible to use the last two single-qubit gates just shown to prepare a qubit in any state [2] starting from $|0\rangle$:

$$R_z\left(\frac{\pi}{2}+\phi\right)HR_z(\theta)H|\psi\rangle = e^{i\frac{\theta}{2}}\left(\cos\left(\frac{\theta}{2}\right)|0\rangle + \sin\left(\frac{\theta}{2}\right)e^{i\phi}|1\rangle\right). \tag{1.23}$$

  Where the global phase $e^{i\frac{\theta}{2}}$ is irrelevant [3, 12] and can be eliminated to get to (1.3).

- CONTROLLED-NOT GATE $\quad CNOT = \begin{pmatrix}\mathbb{1} & \mathbb{0}\\ \mathbb{0} & \sigma_x\end{pmatrix}$

  Which is a $4\times 4$ matrix because it acts on two qubits, that live in a four-dimensional Hilbert space. The computational basis for such space is $\{|00\rangle, |01\rangle, |10\rangle, |11\rangle\}$ and CNOT gate acts on it as:

$$CNOT|00\rangle = |00\rangle, \quad CNOT|01\rangle = |01\rangle, \quad CNOT|10\rangle = |11\rangle, \quad CNOT|11\rangle = |10\rangle. \tag{1.24}$$

  The first qubit is called 'control qubit' while the second one 'target qubit'. The action of the CNOT gate can be described as: if the control qubit is in state $|0\rangle$ then a buffer gate is applied to the target qubit, else if the control qubit is $|1\rangle$ then a NOT gate is applied to the target qubit. Similarly, any controlled gate uses a qubit as control and targets another applying that particular gate to the latter. CPHASE($\pi$), for example, applies a $\sigma_z$ (called *Z-Gate* $= R_z(\pi)$).

This process is used to generate entanglement [2]. For example, starting with two qubits in state $|0\rangle$, a Hadamard gate is applied to one of them, which is then used as a control qubit of a CNOT gate that has the other qubit as target:

$$CNOT(H_1 \otimes \mathbb{1}_2)|00\rangle_{12} = CNOT\left[\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)_1 \otimes |0\rangle_2\right] = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)_{12}. \tag{1.25}$$

The result is a Bell State, as in Eq.(1.14), a maximally entangled system of two quantum objects. An example of a universal set of quantum gates is $\{CNOT, R_x, R_y, R_z\}$, where the last three are single qubit rotations around each axis, the last being the Phase-shift gate.

## 1.5 Projective measurement

Observing a quantum object destroys its inherent quantum properties. This is a consequence of the fact that a measurement is always an invasive process in quantum mechanics: to carry out a measure it is necessary to interact with the object, which consequently no longer evolves in a unitary way [3]. This is formalized by Von Neumann's projection postulate [15]. Consider a generic quantum state $|\psi(t)\rangle$. At time $t_0$ the generic observable $\hat{A}$ is measured, getting as a result $a \in \sigma(\hat{A})$, where $\sigma(\hat{A})$ is the spectrum of operator-observable $\hat{A}$. Consequently, the state $|\psi(t_0)\rangle$ is projected in an eigenstate $|a\rangle$ of $\hat{A}$ belonging to the eigenspace of eigenvalue $a$. At time $t_+$, immediately following $t_0$, the system is found in the state:

$$|\psi(t_+)\rangle = \frac{P_a^{\hat{A}}|\psi(t_0)\rangle}{\sqrt{\langle\psi(t_0)|P_a^{\hat{A}}|\psi(t_0)\rangle}} \tag{1.26}$$

Where $P_a^{\hat{A}} = |a\rangle\langle a|$ is the projector on eigenstate $|a\rangle$ [12].
If the system is already in the state $|a\rangle$ then measuring $\hat{A}$ does not change the state of the system: the measurement $a$ will be obtained with probability 1 and the state will continue to be $|a\rangle$.
There are other types of measurements, such as POVM (Positive Operator Valued Measurements) and generalized measurements, which are performed to perturb the system's state as little as possible. In any case, how a measurement collapses a wave function remains an open problem in quantum mechanics.

## 1.6 Quantum channels

Up to now, we discussed only the ideal evolution of a quantum state. However, real quantum computers interact with their surroundings, which results in a noisy evolution of the system. "Noise" is any interaction between the qubits and the environment, such as molecules of the atmosphere at room temperature, cosmic rays, or other energetic particles hitting the quantum processor. When this happens some information is lost in the environment and cannot be accessed any longer. On top, (some of) the quantum properties of the qubits are deleted. It is important to study how this phenomenon affects the qubits to implement hardware solutions tailored to solving the problem. Furthermore, this allows us to guess which are the more likely errors encountered in the evolution, and thus develop protocols to correct them. Assuming that the system, quantum register plus environment, still undergoes a unitary evolution, it is possible to describe the evolution of the quantum register employing the Kraus Representation [2]. Further assuming that the system's state $\rho_{reg+env}$ is initially completely separable: $\rho_{reg+env} = \rho_{reg} \otimes \rho_{env}$ and that the environment is in a pure state $\rho_{env} = |0\rangle_{env}\langle 0|_{env}$ (without loss of generality since one can always purify a state in a larger Hilbert space), then:

$$\rho_{reg}(t) = Tr_{env}(\rho_{reg+env}(t)) = \sum_{k=1}^{dim(\mathcal{H}_{env})} E_k \cdot \rho_{reg} \cdot E_k^\dagger \qquad with \quad \sum_{k=1}^{dim(\mathcal{H}_{env})} E_k^\dagger \cdot \rho_{reg} \cdot E_k = \mathbb{1}_{reg} \tag{1.27}$$

$E_k$ are called Kraus operators while the map S, as below defined, is a superoperator.

$$S : \rho \rightarrow \rho' = \sum_k E_k \cdot \rho \cdot E_k^\dagger \tag{1.28}$$

Superoperators have the following properties:

- S is a linear map that preserves the Hermiticity, trace and non-negativity of the operators, thus evolving a density matrix with a superoperator yields a density matrix.

- S has group properties: given the operators $\rho_0$, $\rho_1$ and $\rho_2$ and the superoperators $S_1$ and $S_2$ so that $\rho_1 = S_1(\rho_0)$ and $\rho_2 = S_2(\rho_1)$ then $S \;/\; S(\rho_0) = (S_2 \circ S_1)(\rho_0) = S_2(\rho_1) = \rho_2$ is also a superoperator.
- The inverse of S exists if and only if S is also unitary. This entails that whenever a non-unitary evolution occurs (for example an interaction of the qubits with the environment) this is not reversible: an arrow of time appears and the lost information is not recoverable.

Superoperators can always be written using Kraus operators due to the Kraus representation theorem [16]. They are employed to define various quantum channels. A quantum channel [2] is a specific non-unitary evolution of one (or more) qubits, typically due to coupling with the environment or instrumentations malfunctions, which unpredictably modifies its initial state. As a result, errors are often produced in the execution of the algorithms that make use of the affected qubits. Below we illustrate some quantum channels to mathematically formalize some of the errors that the QEC algorithms we studied aim to correct.

- BIT-FLIP CHANNEL:

$$\rho' = S(\rho) = P_{bf}(\sigma_x \cdot \rho \cdot \sigma_x^\dagger) + (1 - P_{bf}) \cdot \rho = \sum_{k=0}^{1} E_k \cdot \rho \cdot E_k^\dagger \tag{1.29}$$

$$with \quad E_0 = \sqrt{1 - P_{bf}} \cdot \mathbb{1}, \quad E_1 = \sqrt{P_{bf}} \cdot \sigma_x$$

  It performs a NOT ($\sigma_x$) gate on the qubit with probability $P_{bf}$ or leaves it unchanged in the remaining cases $(1 - P_{bf})$. It acts on the basis states in the following way:

$$\begin{aligned} \rho = |0\rangle\langle 0| \longrightarrow \rho' = S(\rho) = P_{bf}|1\rangle\langle 1| + (1 - P_{bf})|0\rangle\langle 0| \\ \rho = |1\rangle\langle 1| \longrightarrow \rho' = S(\rho) = P_{bf}|0\rangle\langle 0| + (1 - P_{bf})|1\rangle\langle 1| \end{aligned}$$

  It can also be represented as a CNOT gate applied to the affected qubit with an "environmental" qubit $|\psi\rangle_{env} = \sqrt{1 - P_{bf}} \cdot |0\rangle + \sqrt{P_{bf}} \cdot |1\rangle$ acting as control.

- PHASE-FLIP CHANNEL:

$$\rho' = S(\rho) = P_{ph}(\sigma_z \cdot \rho \cdot \sigma_z^\dagger) + (1 - P_{ph}) \cdot \rho = \sum_{k=0}^{1} E_k \cdot \rho \cdot E_k^\dagger \tag{1.30}$$

$$with \quad E_0 = \sqrt{1 - P_{ph}} \cdot \mathbb{1}, \quad E_1 = \sqrt{P_{ph}} \cdot \sigma_z$$

  It acts as a $\sigma_z$ gate on the qubit with probability $P_{ph}$ or as a buffer in the remaining cases $(1 - P_{ph})$. It operates on generic mixed states as follows.

$$\rho_m = \begin{pmatrix} p_0 & \alpha \\ \alpha^* & 1 - p_0 \end{pmatrix} \longrightarrow \rho' = S(\rho_m) = \begin{pmatrix} p_0 & \alpha(1 - 2P_{ph}) \\ \alpha^*(1 - 2P_{ph}) & 1 - p_0 \end{pmatrix}$$

  This means that if phase-flip happens with probability $P_{ph} = \frac{1}{2}$ (totally random) then the final state has all coherence terms equal to zero: random phase-flip is a mathematical model for decoherence. Its action is equivalent to a CPHASE($\pi$) gate applied to the affected qubit with an "environmental" qubit $|\psi\rangle_{env} = \sqrt{1 - P_{ph}} \cdot |0\rangle + \sqrt{P_{ph}} \cdot |1\rangle$ as control qubit.

- BIT+PHASE-FLIP CHANNEL: It has the same form as the previous two with $\sigma_y$ as Pauli operator. It performs both a bit-flip and a phase-flip simultaneously with probability $P_{bpf}$.

- DEPOLARIZING CHANNEL:

$$\rho' = S(\rho) = \frac{P_d}{3}(\sigma_x \cdot \rho \cdot \sigma_x^\dagger + \sigma_y \cdot \rho \cdot \sigma_y^\dagger + \sigma_z \cdot \rho \cdot \sigma_z^\dagger) + (1 - P_d) \cdot \rho = \sum_{k=0}^{3} E_k \cdot \rho \cdot E_k^\dagger \tag{1.31}$$

$$with \quad E_0 = \sqrt{1 - P_d} \cdot \mathbb{1}, \quad E_i = \left\{ \sqrt{\frac{P_d}{3}} \cdot \sigma_i \right\}_{i=x,y,z}$$

  It is a union of the previous three flip channels with probability $P_d$.
  By applying this channel several times it is possible to collapse the entire Bloch sphere on the origin, i.e. transform a generic state into $\rho = \frac{\mathbb{1}}{2}$ which is the maximally mixed state, losing all the information stored in the qubit.
  This channel is implemented in quantum computing simulations to faithfully reproduce the noise of real architectures.

## 1.7 Quantum circuits

A quantum circuit is a diagram of a quantum algorithm and its implementation on a quantum computer. It is consisting of quantum registers that are initialized (1), then evolved through quantum gates (2) and ultimately measured (3). Sometimes, mostly in quantum error correction, a quantum circuit contains also the schematic of a quantum channel (4), representing the errors that could happen during the computation and that one wants to be able to correct.

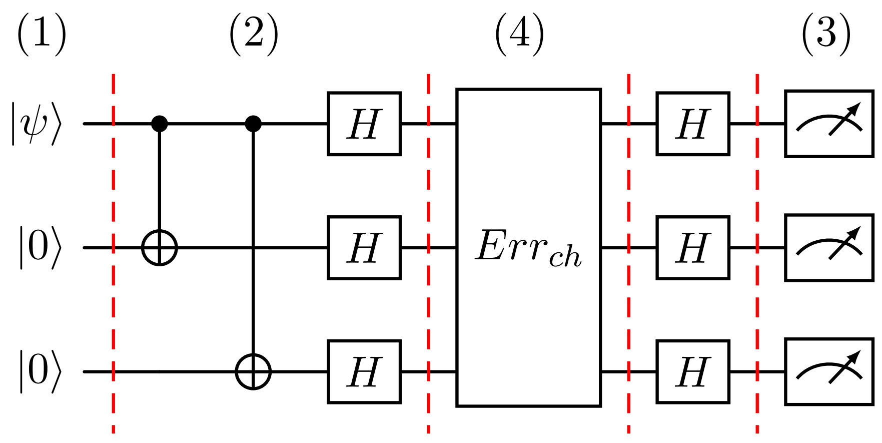

Figure 1.2: *Visual representation of a quantum circuit.* Qubits evolve in time following the horizontal lines from left to right [3]. Single qubit gates are pictured as boxes containing the symbol of the operator (Hadamard in this case). CNOT gate is represented as a dot on the control qubit and a circle with a cross on the target qubit.

Every quantum circuit has an associated quantity called depth. The depth of a quantum circuit corresponds to the number of layers (or groups) of gates that can be applied on the register simultaneously, i.e. in parallel. This quantity is directly proportional to the execution time of the algorithm. When running on real architectures, time is a fundamental resource: the longer time a code takes, the more it risks failing due to the occurrence of errors and decoherence (see Section (2.1)). Therefore, between two equivalent circuits (that produce the same output from a common input), it is advisable to implement the one with a lower depth.

Chapter 2

# Fundamentals of Quantum Error Correction

Implementing error correction algorithms is not a new concept introduced by quantum computing. Even classical computers are subject to random fallacies that can alter the information contained within their memory. Usually, to make a classical computer robust to this type of phenomena, redundancy is used. This is done by saving several copies of the same message so that, if an error occurs in less than half of them, the initial message can be reconstructed by taking the more frequent version. This trick is not directly applicable to the quantum world. First, a generic quantum state cannot be copied, due to the no-cloning theorem [2], so another way to generate redundancy is needed. Furthermore, it is not possible to measure quantum states to evaluate the presence of errors, otherwise, all the non-classical information contained in the qubits would be lost. One way to produce redundancy of quantum information is to store the state of each single qubit in a higher dimensional Hilbert space employing many qubits. A group of qubits containing the data of a single one is called *logical qubit* because it constitutes a single unit of information within the algorithm. Instead, the actual hardware implementation of a qubit is known as *physical qubit*. Another complication, with respect to the classical case, is that here there are many different kinds of errors. In fact, since a bit can only assume the values 0 and 1, the only way in which its content can be altered is the bit-flip: the zero becomes one and vice versa. On the other hand, a qubit can adopt any superposition of the two so consequently there are also many ways in which the information can be modified. Therefore, in order to correct an error in a quantum state it is necessary to predict what types are more common and to which quantum channel they are associated. Then, enforce specialized algorithms to make qubits resistant to those specific errors.

## 2.1 Real hardware quantum errors

We present below some of the most frequent types of error in current NISQ architectures.

### 2.1.1 Coherent q-errors

This type of error occurs whenever a gate is applied incorrectly, a hamiltonian $H$ is assumed while the real system evolves according to $H'$ or again the finite experimental resolution of the instrumentation has non-negligible effects. This results in unwanted but still unitary evolutions of the affected qubits. This means that coherent techniques can be used to mitigate the effects without the loss of information to the environment or of the quantum properties of the processor. One of these techniques is composite pulse sequence. A simple model of coherent q-errors is "Incorrect characterization of control dynamics". Consider a fallacious gate that is applied $N$ times to a qubit within an algorithm. Consequently, the same qubit is rotated $N$ times around the x-axis ($\sigma_x$). Each rotation is modulated by a factor $\epsilon$ that estimates how much the gate calibration is incorrect. For example, if the starting state of the qubit is $|0\rangle$, then:

$$|\psi_f\rangle = \prod_{i=1}^{N} e^{i\epsilon\sigma_x}|0\rangle = cos(N\epsilon)|0\rangle + i \cdot sin(N\epsilon)|1\rangle. \tag{2.1}$$

$$So, \quad P_{success} = P(|0\rangle) = cos^2(N\epsilon) \sim 1 - (N\epsilon)^2,$$

$$P_{error} = P(|1\rangle) = sin^2(N\epsilon) \sim (N\epsilon)^2 \xrightarrow{N\epsilon \ll 1} 0.$$

Note that the rotation of the qubit around $\hat{x}$ always occurs in the same direction: this type of malfunctioning remains constant in each application, therefore the errors produced are systematic. Conversely, random fluctuations of the control fields are non-coherent and analogous to what we describe in the next subsection.

### 2.1.2 Environmental decoherence

Environmental decoherence is a physical process that happens whenever a quantum system interacts with the surrounding environment, leading to a replacement of its quantum coherence with classical correlations. This causes the dispersion of information and entanglement outside of the system. A simple model of it, other than the Phase-flip channel, is the Coupling channel. Consider the environment to be a bipartite system so that its state is:

$$|\psi_E\rangle = \gamma|e_0\rangle + \lambda|e_1\rangle \qquad with \ \langle e_i|e_j\rangle = \delta_{ij}, \ \ |e_0\rangle\langle e_0| + |e_1\rangle\langle e_1| = \mathbb{1}. \tag{2.2}$$

While our qubit is initially in state $|\psi_Q\rangle$ of the form of Eq.(1.2).
The error is modeled with a CNOT applied to $|\psi_E\rangle$ with $|\psi_Q\rangle$ as control, i.e. the qubit and the environment become entangled. For example, starting with our qubit in a quantum superposition, $|\psi_Q\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$, and the environment in state $|\psi_E\rangle = |e_0\rangle$, the initial (product) state of the composite system will be:

$$|\psi\rangle^i_{q+env} = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \otimes |e_0\rangle = \frac{1}{\sqrt{2}}(|0\rangle|e_0\rangle + |1\rangle|e_0\rangle). \tag{2.3}$$

After applying the coupling channel, the final state will be:

$$|\psi\rangle^f_{q+env} = \frac{1}{\sqrt{2}}(|0\rangle|e_0\rangle + |1\rangle|e_1\rangle), \tag{2.4}$$

which has density matrix:

$$\rho^f_{q+env} = \frac{1}{2}(|0\rangle|e_0\rangle\langle 0|\langle e_0| + |0\rangle|e_0\rangle\langle 1|\langle e_1| + |1\rangle|e_1\rangle\langle 0|\langle e_0| + |1\rangle|e_1\rangle\langle 1|\langle e_1|). \tag{2.5}$$

$\rho^f_{q+env}$ is non-separable. Performing the partial trace over the environment, about which we have no information, we get the reduced density matrix relative to our qubit. Confronting it with the qubit's initial density matrix we can see the effects of decoherence:

$$\rho^i_q = \frac{1}{2}(|0\rangle\langle 0| + |0\rangle\langle 1| + |1\rangle\langle 0| + |1\rangle\langle 1|) \xrightarrow{\text{decoherence}} \rho^f_q = \frac{1}{2}(|0\rangle\langle 0| + |1\rangle\langle 1|). \tag{2.6}$$

$\rho^f_q$ is that of a maximally mixed state: the qubit is now in a completely classical state. The difference is that the final state has no quantum correlations, encoded in the coherence terms: the qubit state could be $|0\rangle$ or $|1\rangle$ with equal classical probability due to our ignorance of the past of the system. The coherence terms are those that are present in $\rho_q^i$ but not in $\rho_q^f$. If we want a decoherence model that leaves residual coherence in the final state instead of a maximally mixed state we can use the Phase-flip channel (1.30) with $P_{ph} < \frac{1}{2}$. However, both of these channels have only an implicit dependence on time, while other more realistic models make it explicit. It is important to underline the time dependence of the qubit decoherence because it occurs very rapidly, representing a major limitation of quantum computation.

### 2.1.3 Other non-coherent errors

- **Measurement errors**
  A simple yet effective model for them is the Bit-flip channel (1.29), with $P_{bf}$ being the probability of measurement error, followed by an ideal measurement. Another model employs POVM. The two models are equivalent in terms of measured quantities and their probabilities but differ in the qubit's final state after the measurement. However, a qubit is usually discarded or reset after a measurement, so the two models are considered equivalent. In our discussion later we neglect this type of error as it is not the target of basic QEC.
- **Qubit initialization**
  This error can be modeled either as non-coherent or as coherent. The first case is analogous to imperfect measurement. For example, given $P_I$, the probability of initialization error, if the desired state is $|0\rangle$, then the actual state will be:
  $$\rho_I = (1 - P_I)|0\rangle\langle 0| + P_I|1\rangle\langle 1|. \tag{2.7}$$
  Instead, in the coherent case the actual state will be pure but containing a greater than zero amplitude of an undesired state:
  $$|\psi_I\rangle = \alpha|0\rangle + \beta|1\rangle \qquad \textit{with} \;\; |\alpha|^2 + |\beta|^2 = 1, \;\; |\beta|^2 \ll 1. \tag{2.8}$$
- **Qubit loss and leakage**
  Given $\rho_{reg}^i$, the density matrix of the initial quantum register, after the loss of a qubit it will become $\rho_{reg}^f = Tr_j(\rho_{reg}^i)$, where $j$ is the index of the affected qubit. All information contained in that qubit is lost.
  Leakage happens because many physical systems implementing qubits have more than two levels. If the control frequencies oscillate around target setting or the additional levels are not sufficiently detuned from the qubit resonance, levels other than the intended two can become populated. This results in a transformation like $U|0\rangle = \alpha|0\rangle + \beta|1\rangle + \gamma|2\rangle$.
  The final state is defined on three levels. This leads to unwanted dynamics within the algorithms, which are designed for the evolution of bipartite systems. To detect qubit loss or leakage "non-demolition detection" schemes can be adopted: the presence or confinement of the qubit is verified without carrying out a projective measurement of it.

## 2.2 Repetition code

We begin to illustrate the more common QEC algorithms with the simplest one, directly inspired by the classical redundancy method.

### 2.2.1 3-Qubit Repetition code

We encode the information of a logical qubit into three physical qubits, called data qubits. With this code, we are able to correct a single bit **or** phase flip on one of the data qubits.

Specifically, we depict below the circuit capable of correcting the bit-flip, on which we focused more during the experiments on real hardware (Chapter 3). The algorithm can be divided into three main phases, which we now break down in order.

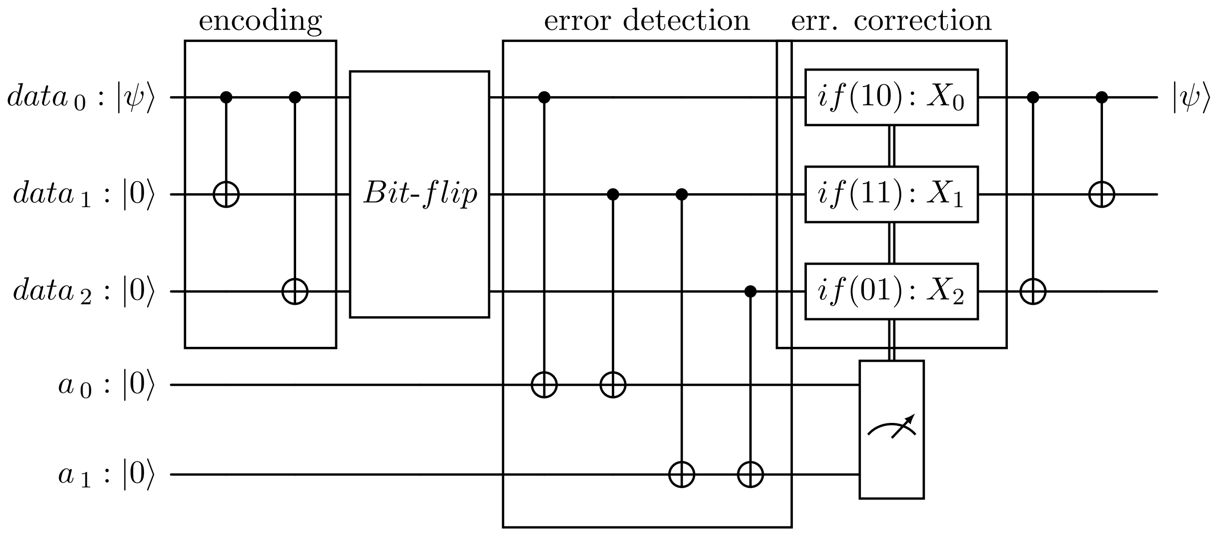

Figure 2.1: *3-Qubit Repetition code for the correction of bit-flip.* The circuit comprises a quantum register of three data qubits and two ancillary qubits that are evolved through CNOTs and classically controlled gates. Data qubits are also subject to the application of the Bit-flip channel, which is responsible for generating errors.

- **State encoding**
  We encode the state of a single data qubit into the entangled state of the three qubits, using the following encoding:
  $$|0\rangle \rightarrow |\tilde{0}\rangle = |000\rangle, \qquad |1\rangle \rightarrow |\tilde{1}\rangle = |111\rangle. \tag{2.9}$$
  So, using two CNOTs as shown in Figure (2.1), the state of the quantum register will be transformed as below:
  $$|\psi\rangle_0 \otimes |0\rangle_1 \otimes |0\rangle_2 = (\alpha|0\rangle + \beta|1\rangle)_0 \otimes |0\rangle_1 \otimes |0\rangle_2 \longrightarrow |\tilde{\psi}\rangle = \alpha|\tilde{0}\rangle + \beta|\tilde{1}\rangle = (\alpha|000\rangle + \beta|111\rangle)_{012}.$$
  Which is an entangled state in a 3-qubit Hilbert space.

After the encoding, the logical qubit is treated as a quantum memory, i.e. no further evolution is applied to the quantum register. In this idle time, the system still interacts with the environment, undergoing the Bit-flip channel, i.e. there is a finite probability of a bit-flip error occurring independently on each data qubit. An error and its corresponding location (which qubit it affects) are called a syndrome.

- **Error detection**
  To detect the error in the logical qubit without projecting its quantum state, we measure the parity of two different pairs of data qubits by means of ancillary qubits. In the circuit shown in Fig.(2.1), the chosen pairs are $\{data_0, data_1\}$, $\{data_1, data_2\}$ to which the ancilla-qubits $a_0$ and $a_1$ are respectively associated. The ancillas are both initialized to $|0\rangle$. Applying the error-detecting circuit reported in Figure (2.1), the ancillary qubit is flipped to $|1\rangle$ if and only if the qubits of the pair are equally likely to be found in opposite states. In fact, if they are both $|0\rangle$, no operation is performed. If they are both $|1\rangle$ two NOT gates are applied to the ancilla, canceling each other and leaving the ancilla in $|0\rangle$. Conversely, if the qubits are in opposite states, i.e. one of them has been flipped, a single NOT gate is applied to the ancilla, which becomes $|1\rangle$. Finally,

by measuring the ancillas, we obtain information about the parity of each pair of data qubits without collapsing their states. By knowing whether and which of the qubits are pairwise equal, one can easily establish whether a bit-flip has occurred and which of the qubits is involved. This constitutes an error detection event.

| **Data qubits ($d_i$) rel.** | $d_0 = d_1 = d_2$ | $d_0 = d_1 \neq d_2$ | $d_0 \neq d_1 = d_2$ | $d_0 = d_2 \neq d_1$ |
|---|---|---|---|---|
| **Ancillas' measures** | 00 | 01 | 10 | 11 |
| **Qubit flagged for correction** | none | $data_2$ | $data_0$ | $data_1$ |

- **Error correction**
  Now all that remains is to flip the qubit in the wrong state via a classically controlled NOT gate. This is well implemented by most modern current architectures with dynamical circuits. This technology allows you to add a series of gates that can be conditionally activated based on the content of a classical register.

Finally, the initial single-qubit state is recovered from the logical qubit by performing the decoding. Due to the unitarity of quantum gates, to undo any operation it is sufficient to apply its corresponding gates in reverse order. So decoding is simply inverse encoding. Alternatively, if the goal is to keep the qubit in memory even longer, decoding can be postponed to perform other rounds of error correction without re-encoding the logical qubit. Furthermore, to implement the algorithm cyclically, the ancillas must be reset between one round and the next. We report in the table below the possible states that the quantum register can assume during the execution of the circuit, with respective probability and consequent success or failure of the algorithm.

| **Post-Bit-flip Ch. state** | **Probability** | **Post-Correction state** | "$a_0 - a_1$" |
|---|---|---|---|
| $\alpha\|000\rangle + \beta\|111\rangle$ | $(1-\epsilon)^3$ | $\alpha\|000\rangle + \beta\|111\rangle$ Success | 00 |
| $\alpha\|100\rangle + \beta\|011\rangle$ | $\epsilon(1-\epsilon)^2$ | | 10 |
| $\alpha\|010\rangle + \beta\|101\rangle$ | $\epsilon(1-\epsilon)^2$ | | 11 |
| $\alpha\|001\rangle + \beta\|110\rangle$ | $\epsilon(1-\epsilon)^2$ | | 01 |
| $\alpha\|110\rangle + \beta\|001\rangle$ | $\epsilon^2(1-\epsilon)$ | $\alpha\|111\rangle + \beta\|000\rangle$ Failure | 01 |
| $\alpha\|101\rangle + \beta\|010\rangle$ | $\epsilon^2(1-\epsilon)$ | | 11 |
| $\alpha\|011\rangle + \beta\|100\rangle$ | $\epsilon^2(1-\epsilon)$ | | 10 |
| $\alpha\|111\rangle + \beta\|000\rangle$ | $\epsilon^3$ | | 00 |

Table 2.1: *Data qubit register state progression in the 3-Qubit code.* Corresponding ancillas' measurement are indicated as the two-bit string "$a_0 - a_1$".

Although we are considering only one type of error, which the algorithm is specifically designed to correct, we do not always get the desired output. When an error occurs on two separate data qubits, the algorithm flags and flips the third, that was unaffected. When all three suffer a bit-flip, there is no detection event. These eventualities lead to a wrong final state and are so-called logical errors. The chances that multiple errors occur are however small assuming that the probability of a single error is $P_{bf} = \epsilon \ll 1$. From which:

$$P_{success} \simeq (1-\epsilon)^3 + 3\epsilon(1-\epsilon)^2, \qquad P_{fail} \simeq \epsilon^3 + 3\epsilon^2(1-\epsilon). \tag{2.10}$$

$$So, \quad P(0\,or\,1\,errors) \gg P(2\,or\,3\,errors).$$

Note also the theoretical gain of the code, since:

$$P_{fail}(3\text{-}Qubit\ Code) \simeq \epsilon^3 + 3\epsilon^2(1-\epsilon) \simeq O(\epsilon^2) \ll P_{fail}(Single\ Qubit) = \epsilon. \tag{2.11}$$

Probabilities of success/failure are estimated neglecting the occurrence of more than three errors, which are higher order terms. We are also neglecting the favorable possibility of two errors happening on the same qubit, canceling each other out. This description is far simpler compared to the real case. Further errors on applied gates and channels different from bit-flip are not considered. Also, ancillas are qubits themselves and, as such, their real implementations are flawed, from initialization to measurement. We will cover the limitations of this code in the next chapter.
We should also mention that this circuit can be readily modified to correct phase-flip instead of bit-flip. Since a phase-flip is simply a bit-flip in $\sigma_x$-basis $\{|+\rangle, |-\rangle\}$, the circuit can be adapted by just adding a change of basis after the encoding. This is achieved by applying a Hadamard gate on every data qubit, exactly as displayed by the circuit in Figure (1.2). The classically controlled operation undoing the flip will in this case be a $\sigma_z$ gate. Lastly, to extrapolate the post-correction single-qubit state is sufficient to reapply the Hadamards before decoding. While analogous, the two circuits employ different gates so the 3-Qubit code cannot correct both error types at the same time.

### 2.2.2 Scaling the algorithm: $n$-Qubit Repetition code

The Repetition code can be scaled up to $n$ (odd) data qubits encoding the logical one. Such a code can correct up to $\frac{n-1}{2}$ bit/phase-flips on data qubits instead of a single one. The underlying idea is to evaluate the parity of $(n-1)$ pairs of qubits. Then, flip the qubits that have the opposite state with respect to the majority. We report below the Repetition code circuit generalized to $n$ qubits.

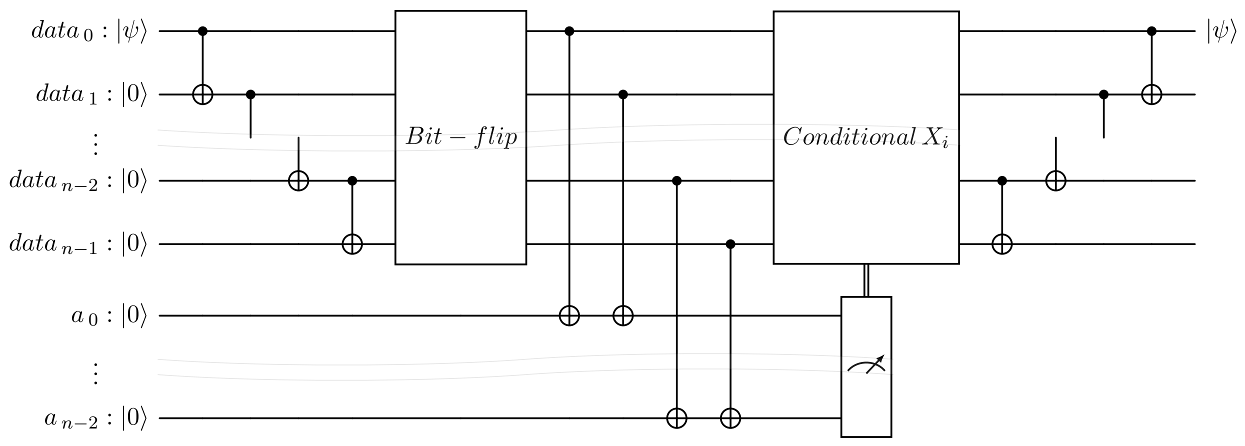

Figure 2.2: *$n$-Qubit Repetition code for the correction of bit-flip.* Dots indicate the presence of additional qubits. The quantum register is now composed of $n$ data qubits and $n-1$ ancillas.

There are some details to consider when scaling the code.

- The encoding can be generalized to $n$ qubits using a series of cascading CNOTs. This method avoids many swaps compared to always using $data_0$ as control qubit and all the others as targets, as we did with the 3-Qubit code. A swap is used to allow two distant qubits to interact with each other. These operations are time-consuming and can be fallacious, introducing a greater risk of computational failures. We proceed using $data_i$ as control and $data_{i+1}$ as target, up to the last data qubit as target. This is the type of encoding with the lowest possible depth. However, if an error occurs in one of the qubits during the encoding, then there will be a catastrophic ripple effect on all the following ones.
- For the study of parity, the $(n-1)$ pairs $\{i, i+1\}_{i=0\ldots n-2}$ of data qubits must be correlated to $(n-1)$ ancillas. The ancillas are measured in order and the result is stored in a $(n-1)$bit-

string. For $n$ data qubits there are $n_{syn} = \sum_{i=1}^{(n-1)/2} \binom{n}{i}$ different syndrome bit-strings, each of which corresponds to the location of 1 to $(n-1)/2$ affected qubits. $n_{syn} \sim O(n!)$, therefore for large repetition code sizes it is necessary to find an efficient method to calculate all the possible syndromes. This is known as the lookup-table decoding (classical) problem. The Python program we wrote for a general $n$-Qubit Repetition code [17] addresses the problem as follows. We take a string of $n$ zeros and transform part of them (from a minimum of 1 to a maximum of $\frac{n-1}{2}$) into ones, thus producing all possible incorrect strings of "qubits". Subsequently, the XOR between each successive pair of bits is calculated, generating the strings of ancillas' measures corresponding to each syndrome. The latter and the corresponding positions of the altered bits are preserved. All the possible syndrome strings must be evaluated during the execution of the Repetition code, possibly in ascending order of errors contained, since fewer errors are more probable.

While $n$-Qubit codes are able to correct more errors than a 3-Qubit Repetition code, they require more gates too. Thus, as we will see in Chapter 3, a more complex code leads to better results only in specific gate-error regimes. We will elaborate on this topic in the next chapter.
To evaluate the amount of errors that a code can detect and correct as its size increases, it is useful to introduce the concept of code distance. The code distance is a metric characterizing the minimum number of physical qubit errors needed to yield an uncorrectable logical one. The distance of the code is directly proportional to its robustness to errors. This property of QEC algorithms is often included in the following representation: $[n,\, k,\, d]$, where $n$ is the number of physical qubits encoding $k$ logical qubits and $d$ is the code distance. For repetition codes, the following applies:

$$d = \frac{n+1}{2} \tag{2.12}$$

## 2.3 Stabilizers

Up to now we discussed the repetition codes using the quantum circuit formalism and the state-vector representation. However, they belong to a more general class of QEC codes: stabilizer codes. The formalism of stabilizers lets us describe more complex algorithms for error correction, able to detect and correct more than one kind of error, for example, both phase **and** bit flip [18]. Stabilizer formalism also provides a generalized procedure for constructing encoding and error-correcting circuits, regardless of the code. Most of all known QEC algorithms belong to this standardized class. We define it using the Heisenberg view of quantum mechanics, which characterizes a quantum state by an operator rather than a state-vector. A state $|\psi\rangle$ is said to be stabilized by the operator $K$ if $|\psi\rangle$ is an eigenvector of $K$ with eigenvalue 1:

$$K|\psi\rangle = |\psi\rangle. \tag{2.13}$$

To generalize to multi-qubit stabilizer states, we analyze the group properties of many-qubit operators. Consider the Pauli group, $\mathcal{P}$, subgroup of all possible single qubit operators:

$$\mathcal{P} = \{\pm\mathbb{1},\, \pm i\mathbb{1},\, \pm\sigma_x,\, \pm i\sigma_x,\, \pm\sigma_y,\, \pm i\sigma_y,\, \pm\sigma_z,\, \pm i\sigma_z\}. \tag{2.14}$$

Knowing the commutation rules of the Pauli matrices, it can be proved that $\mathcal{P}$ forms a group under matrix multiplication. The Pauli group can be extended over $n$-qubits taking the $n$-fold tensor product of all its elements, denoted as $\mathcal{P}_n$ *or* $\mathcal{P}^{\otimes n}$.
Finally, a $n$-qubit stabilizer state $|\psi\rangle_n$ is identified by $n$ generators of $\mathcal{G}$, an Abelian subgroup of $\mathcal{P}^{\otimes n}$. All elements of an Abelian group of operators commute with each other.

$$\mathcal{G} = \{K_i \mid K_i|\psi\rangle_n = |\psi\rangle_n,\;\; [K_i\,,\, K_j] = 0,\;\; \forall(i,j)\} \subset \mathcal{P}^{\otimes n}. \tag{2.15}$$

Every generator is a $n$-qubit stabilizer operator composed of the tensor product of $n$ single-qubit operators. Since every stabilizer is Hermitian and satisfies Equation (2.13), then: $K^2 = \mathbb{1}$. Defining $|\psi\rangle_n$ by specifying its generators or vector-state is perfectly equivalent. Many notable states, such as Bell or GHZ states, are stabilizer states. So are the encoding states of the 3-Qubit code, to which the following generators are associated: $K_1 = XXI, \quad K_2 = IXX \quad$ with $X = \sigma_x$, $I = \mathbb{1}$, $Z = \sigma_z$.

### 2.3.1 QEC stabilizer codes

The stabilizer formalism facilitates creating QEC algorithms and determining which logical operations can be applied to encoded data. Stabilizers define a relevant coding subspace inside the many-qubit Hilbert space relative to the data qubits, reducing its size to a single qubit system, relative to the logical qubit. Each generator halves the dimension of the multi-qubit Hilbert space until it becomes a two-dimensional space. A further operator differentiates the two-dimensional space into two states, called codewords, which will encode the basis states. In QEC, the stabilizers defining the logical subspaces are measured for state preparation and error detection. If a code has the generators of the stabilizer sets that are composed only of operators $X$ and $Z$, it is said to belong to the CSS (Calderbank-Shore-Steane) codes. Algorithms of this type include an encoding of data that allows for the straightforward application of several logical gate operations directly on the logical qubit [10]. Many well-known algorithms fall into this category.
Below are the two main phases of QEC with stabilizer codes.

- **State preparation**
  To perform the encoding is necessary to define valid codeword states, i.e. the basis states of the logical qubit. In a stabilizer code, valid codewords are simultaneous eigenvectors with eigenvalue $+1$ of all generators of the stabilizer group. To project a generic state into a $\pm 1$ eigenstate of a unitary and hermitian operator, $U$, one can implement the following circuit.

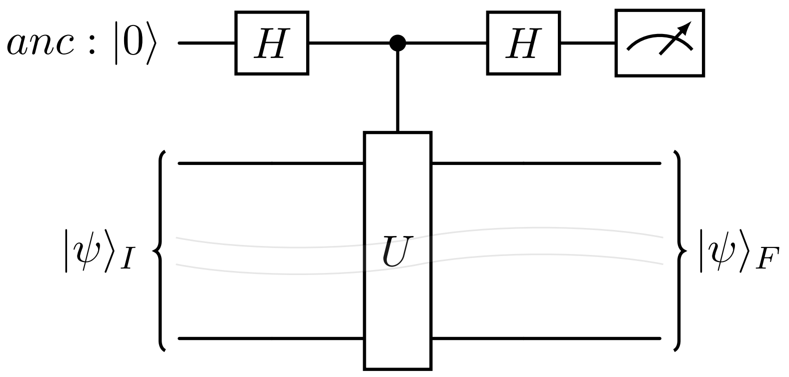

Figure 2.3: *Parity/operator measurement:* circuit capable of projecting any state into an eigenstate of operator $U$.

Measuring the ancilla, one can determine the eigenvalue of $|\psi\rangle_F$ for $U$:

$$|\psi\rangle_F = \frac{1}{2}(|\psi\rangle_I + U|\psi\rangle_I)|0\rangle + \frac{1}{2}(|\psi\rangle_I - U|\psi\rangle_I)|1\rangle. \tag{2.16}$$

Therefore, if the ancilla is measured to be in $|0\rangle$, then:

$$|\psi\rangle_F = |\psi\rangle_I + U|\psi\rangle_I \xrightarrow{U=U^\dagger,\, UU^\dagger = U^\dagger U = \mathbb{1}} U|\psi\rangle_F = |\psi\rangle_F \to AVL = +1. \tag{2.17}$$

Otherwise, if it's in $|1\rangle$:

$$|\psi\rangle_F = |\psi\rangle_I - U|\psi\rangle_I \xrightarrow{U=U^\dagger,\, UU^\dagger = U^\dagger U = \mathbb{1}} U|\psi\rangle_F = -|\psi\rangle_F \rightarrow AVL = -1. \tag{2.18}$$

From a practical point of view, to carry out the encoding it is necessary to prepare as many ancillas as the number of generators and to apply a Hadamard to each one. Then, for each generator, apply every single-qubit operator on the corresponding qubit in order, using the ancilla as the control qubit for each gate. Finally, reapply the Hadamards to the ancillas before measuring them.

- **Error correction**
  Error correction is just an extension of state preparation. After the encoding, suppose that one or more errors occur on the logical qubit. An error is identified by an operator $E$ which we assume to be a combination of $X$ and/or $Z$ gates on the $n$ physical qubits constituting the logical state. In this way $E \in \mathcal{P}$ and, given Equation (2.13), we can see that the altered state $E|\psi\rangle_n$ satisfies:

$$K_i E|\psi\rangle_n = (-1)^m E K_i|\psi\rangle_n = (-1)^m E|\psi\rangle_n \qquad with \quad \begin{cases} m=0 & if \ \ [E, K_i] = 0 \\ m=1 & if \ \ \{E, K_i\} = 0 \end{cases} . \tag{2.19}$$

  Because both $E, K \in \mathcal{P}$ and any pair of elements in $\mathcal{P}$ either commutes or anticommutes. So, if $E$ commutes with the i-th stabilizer, $E|\psi\rangle_n$ remains $+1$ eigenvector of $K_i$. Otherwise, if E anticommutes with $K_i$ then $E|\psi\rangle_n$ becomes -1 eigenvector of $K_i$. Therefore of all the possible errors, the former do not modify the information contained in the state while the latter alter the logical qubit, however leaving a trace in the eigenvalue of the respective generator. To complete the correction, all generators must be measured again, in the same way as done for the encoding. An error-free state will be $+1$ eigenvector of each stabilizer. A state that is affected by an error anticommuting with a stabilizer will lead to the respective ancilla being measured in $|1\rangle$. Through the syndrome bit-strings, the involved data qubit is subsequently traced and corrected. When an $X$ stabilizer is flagged, the error is of $Z$-type and vice versa.

### 2.3.2 Steane's [5,1,3] code

We now describe in detail the 5-Qubit Steane's code, an example of a stabilizer circuit, which we have tested in our simulations. This algorithm encodes one logical qubit employing five physical ones. It can correct one single error but that being of any type between bit-flip ($\sigma_x$), phase-flip ($\sigma_z$), or bit+phase-flip ($\sigma_y$), also called Pauli errors. The generators identifying its codeword states are:

$$K_0 = XZZXI, \quad K_1 = IXZZX, \quad K_2 = XIXZZ, \quad K_3 = ZXIXZ.$$

The logical Z-operator $K_4 = ZZZZZ$ discerns the two basis states.
The measurement of each stabilizer, at the basis of both encoding and error detection, takes place as in Figure (2.4), which we schematize to represent the complete algorithm, Fig.(2.5).

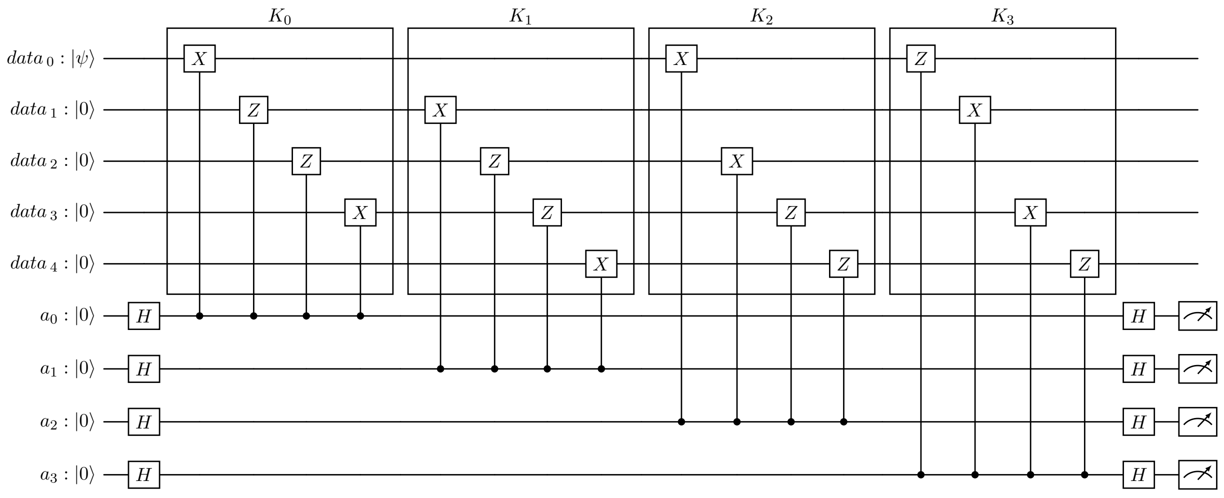

Figure 2.4: *Parity/operator measurement of Steane's [5,1,3] code's generators.* $X$ and $Z$ in CNOTs and CPHASEs are indicated to explicit the operators of each stabilizer. If one of them is the identity then no gate is applied to the corresponding qubit.

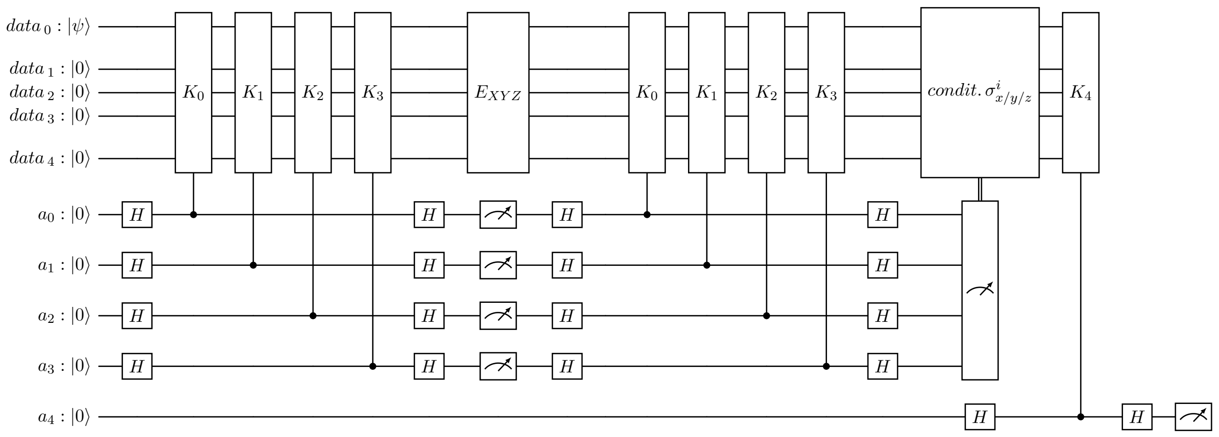

Figure 2.5: *Steane's [5,1,3] code circuit.* A quantum register composed of 5 data qubits and 4+1 ancillas is evolved by measuring the stabilizers. The quantum channel disturbing the data qubits can be any one of the Bit/Phase/Bit+Phase-flip channels, consequently, the correction is done with conditional Pauli operators.

The stabilizers are measured to encode the logical qubit and detect errors just as previously explained. Then, the correction of the Pauli error is carried out by applying the corresponding gate, classically controlled. At last, the success of the algorithm is verified by skipping the decoding and measuring the logical qubit through operator $K_4$. Equivalently, one could check the state of $data_0$ post-inverse-encoding instead. A possible improvement to be made, especially to run the algorithm on real processors, is to realize an encoding with lower depth. The generators uniquely identify the state of the logical qubit, which can be reached employing gates other than the aforementioned ones. A shorter and ancilla-free encoding is possible for this algorithm. We use it in the simulation of [5,1,3] Steane's code in the next chapter. Still employing five data qubits, this code can correct more types of error than the 5-Qubit code, however it includes many more gates. Therefore, it is not immediate to conclude which one is more efficient in faulty real applications.

# Chapter 3

# Real hardware and simulations results

We now present the results we obtained by testing the QEC algorithms just illustrated, running classical simulations and experimenting with IBM quantum computers. Qiskit supplies Python libraries and routines for devising and implementing quantum circuits, pulses and algorithms that can be run both on classical simulators and quantum hardware. All the programs we wrote can be found at [17].
In this chapter we use as a reference the paper ”Exponential suppression of bit or phase errors with cyclic error correction” [19]. Starting from their result on the exponential decrease of logical errors with increasing size of the repetition code, our goal is to study how the probability of a successful correction varies as the repetition code scales. To this aim, we try to reproduce Google's results on some accessible IBM q-processors.
Google's team analyzes error detection events of 50-round 3-to-21 qubit Repetition codes both for bit-flip and phase-flip. They characterize the type of errors studying the correlations between each detection event plotted in matrix form and tracing them back to the phenomena elucidated in Section (2.1). They proceed by showing error mitigation and post-selection methods. An example is the rejection of spikes in detection event fraction which they attribute to cosmic rays hitting the processor and other sporadic events which greatly decrease the performance of the code. Then, they plot the processed data as shown in Fig.(3.1).

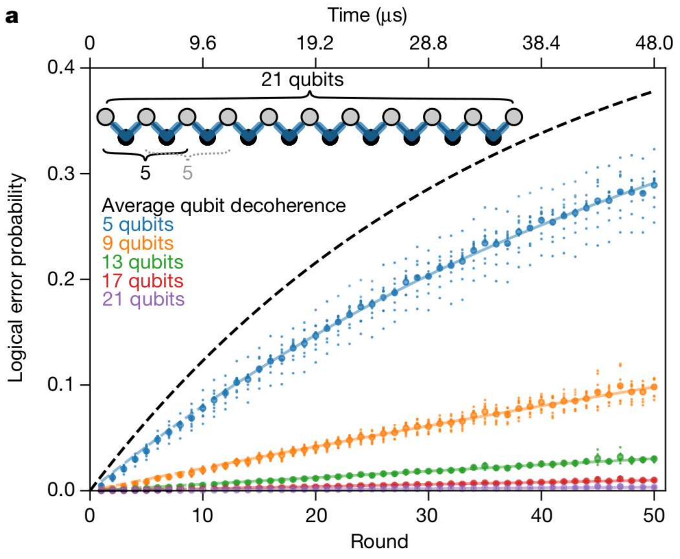

Figure 3.1: *Logical errors in the Repetition code.* Logical error probability versus number of detection rounds and number of qubits for the phase-flip code. Image from [19].

The graph Fig.(3.1) demonstrates experimentally that the success of the code increases significantly with increasing distance, which is proportional to the number of physical qubits encoding the logical qubit.

## 3.1 Simulations

To classically simulate QEC protocols we need to take into account a noise model and suitable simulation methods that admit non-unitary effects. We select a density matrix simulator. In our discussion, we adopt two different types of noise models. The first is a set of identity gates affected by Bit-flip channel; the noisy identities are applied to the data qubits after the encoding. With each application, each data qubit has an independent probability of being flipped. This model represents a faulty memory where qubits are held for a time of $n_i u$, where $n_i$ is the number of identity applications and the unit $u$ is the application time of a single identity. The second type of noise model consists in evolving a qubit via Depolarizing channel (1.31) with each application of a gate. This method is much more realistic as the real applications of quantum gates are flawed and can affect the ancillary qubits as well. Other types of error channels such as qubit initialization and measurements are neglected.
In Subsections (3.1.1) to (3.1.3) we test 3, 5-Qubit Repetition and Steane's [5,1,3] codes in their cyclic version. This approach is meaningful in the case of quantum memories, which are expected to keep the information for as much time as possible. Thus, it is preferable not to measure qubits at each error detection and correction round. Instead, cyclic QEC prevents the accumulation of errors to exceed the number of syndromes that can be corrected by the code, which would consequently fail. We simulate a quantum memory of one logical qubit for $n_i = 32$ time units. We call it "bit-flip memory" or "depolarizing memory" depending on the quantum channel affecting the $n_i$ identity gates applied to the data qubits. As well as the identities, when simulating depolarizing noise all the QEC algorithm's gates are also affected. To have a comparison, we do not only study the logical quantum memory encoded by the QEC protocol but also the physical quantum memory of a single qubit. In this way, we can understand when the encoding is needed. One variable in the implementation of a cyclic algorithm is the number of error correction rounds, $n_r$. The total $n_i$ time units of quantum memory can be interspersed with various rounds of correction with $dt$ waiting time between each other so that $n_r \cdot dt = n_i u = 32u$. By measuring the success probability of each configuration, $P_{success}$, we identify the optimal way to divide the time in memory, i.e. the optimal correction frequency. In our discussion, we consider the following configurations. Starting from one round of error correction after 32 applications of noisy identities, gradually multiplying by two the previous $n_r$, we get to 32 rounds of correction interspersed with one single noisy identity. We estimate $P_{success}$ as the ratio between the number of times a correct state is measured and the total counts. While $P^{bf}_{sq}(n_i)$, the probability of a single qubit surviving $n_i$ time units in bit-flip memory, is the probability that an even number of errors occur, including zero. An even number of bit-flips cancel each other out leaving the qubit in the correct state.

$$P^{bf}_{sq}(n_i) = \sum_{j=0}^{n_i} \binom{n_i}{j} (P_{bf})^j (1 - P_{bf})^{n_i - j}, \tag{3.1}$$

where $P_{bf}$ is the probability of a single time-unit/identity flipping the qubit. Whereas the probability of a single qubit surviving a depolarizing memory, $P^{depol}_{sq}(n_i)$ is the probability of no errors for $n_i$ units of time. That is because depolarizing errors don't cancel each other but, conversely, the more they happen, the more qubit information is lost.

$$P^{depol}_{sq}(n_i) = (1 - P_d)^{n_i}, \tag{3.2}$$

where each identity applies a Depolarizing channel affecting the qubit with $P_d$ probability.
Comparing the success probability of a QEC algorithm with the survival probability of a single

qubit is crucial to understand when QEC becomes effective as opposed to avoiding any correction due to the high error rates.

### 3.1.1 Cyclic 3-Qubit code

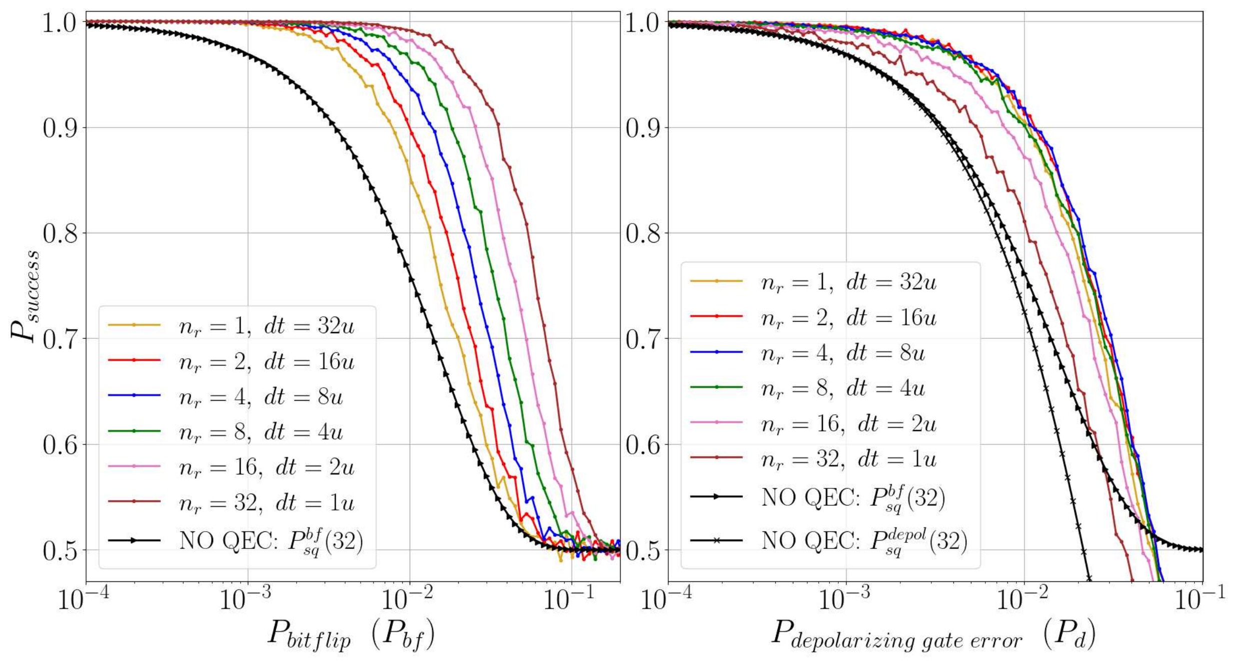

Figure 3.2: *Cyclic 3-Qubit code simulation for different noise models.* Left: Success probability as a function of bit-flip probability in logscale. Right: Success probability as a function of depolarizing gate error probability. In the legend, '$n_r$' specifies the number of correction rounds of the QEC code, '$dt$' is the waiting time between each round. Unencoded single qubit surviving probabilities are in black.

In Fig.(3.2) we study the probability of reaching the target state after cyclic 3-Qubit code as a function of error probability, for each aforementioned noise model and configuration of the code. The left panel considers a bit-flip memory, with $P_{bf}$ the probability of flipping a data qubit when applying a single identity to it. The right panel repeats the same analysis for a depolarizing memory, with $P_d$ the probability of depolarizing error during the application of each gate, including the identities in between rounds. Our results suggest that the 3-Qubit QEC algorithm is more effective than leaving a qubit encoded for bit-flip probability up to 10%. No QEC is effective above this threshold. For $P_{bf} = 1\%$, this Repetition code has more than 90% success probability, while a single uncorrected qubit has about 75%. Under the depolarizing noise model, it is always better performing QEC than leaving the single qubit in a depolarizing memory. Figure (3.2)(right) also contains the case of a single qubit surviving the bit-flip memory. This way we have common ground when comparing performances of the more realistic implementation of the code and the simpler, more idealized case of bit-flips only. The algorithm against depolarizing error produces a gain even over the qubit in a bit-flip memory up to a $P_d = P_{bf} = 6\%$.
The repetition of multiple rounds of error correction causes different outcomes for the two error models. For the bit-flip noise (on the left of Fig.(3.2)) more rounds are better because the QEC code can reveal more errors. The theoretical prediction that we have previously proposed is respected: the success probability increases with the rounds independently from $P_{bf}$. Instead, for the depolarizing noise model, Fig.(3.2)(right), the previous effect is balanced by the fact that more rounds require more gates, which can generate more errors. The ideal configuration has neither the maximum nor the minimum amount of rounds: there is a sweet spot between the two. The graph

shows that success probabilities are better when the correction occurs every quarter/half of the total time in memory. We can conclude that there is a range of errors where applying the 3-Qubit code is actually advantageous. This occurs when data qubit bit-flip error probability is less than 10% and depolarizing gate error is less than 6%. Other types of errors are neglected.

### 3.1.2 Cyclic 5-Qubit code

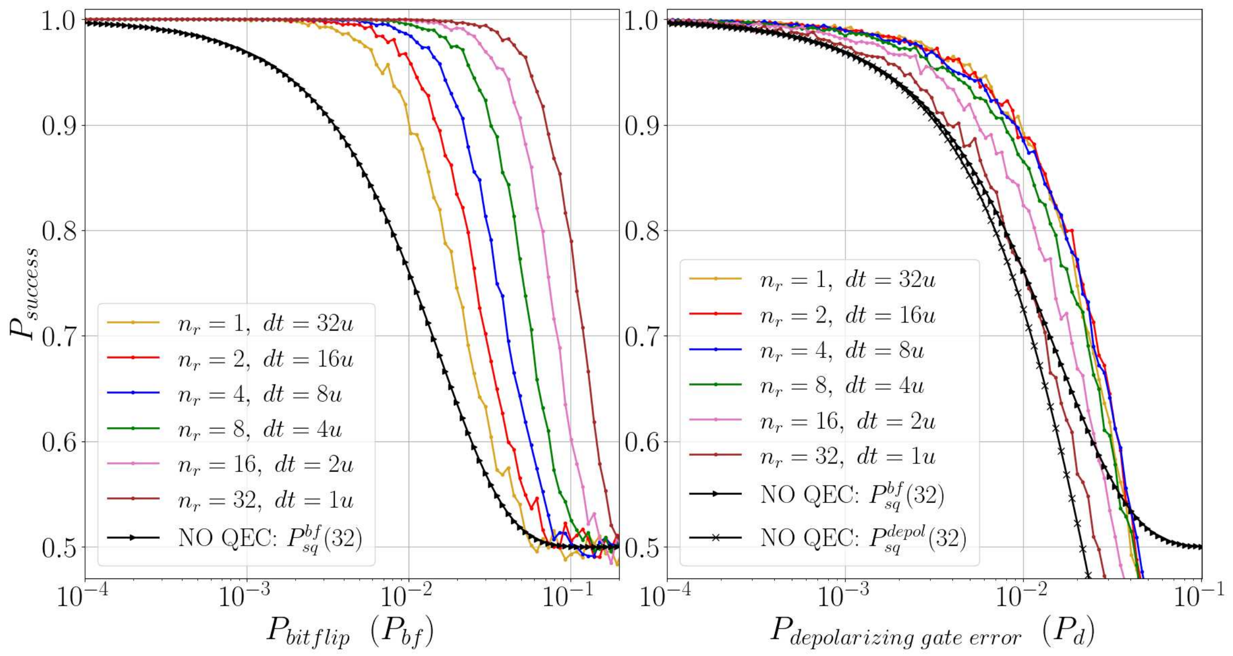

Figure 3.3: *Cyclic 5-Qubit code simulation for different noise models.* Left: Success probability as a function of bit-flip probability in logscale. Right: Success probability as a function of depolarizing gate error probability. In the legend, '$n_r$' represents the number of rounds, '$dt$' the waiting time between each round. Unencoded single qubit surviving probabilities are in black.

By scaling the algorithm to 5 qubits we observe similar behaviors, except that more qubits allow more more errors to be fixed. For this reason, the success probabilities in Fig.(3.3) are greater than those of the 3-Qubit code at the same ideal error rates. In the case of the bit-flip noise model, with maximum rounds of correction ($n_r = 32$), Fig.(3.3)(left), this QEC circuit is still effective 80% of the time for $P_{bf} = 10\%$, while the success probability for the 3-Qubit code is around 58%. Instead, getting the correct measurement from a single qubit under these conditions is a coin toss. The advantages of the correction in this case go up to a bit-flip probability of almost 20%. Under these conditions, we still observe that more rounds of corrections correspond to more corrected errors. Containing more gates, at high rates of depolarizing errors the 5-Qubit code performs slightly worse than its 3-Qubit counterpart. In this range, the plot shows that it is best to carry out the correction in just 1-2 rounds, to avoid adding even more gates. For the comparison between the algorithms at lower error rates see Subsection (3.1.4).

### 3.1.3 Cyclic [5,1,3] Steane's code

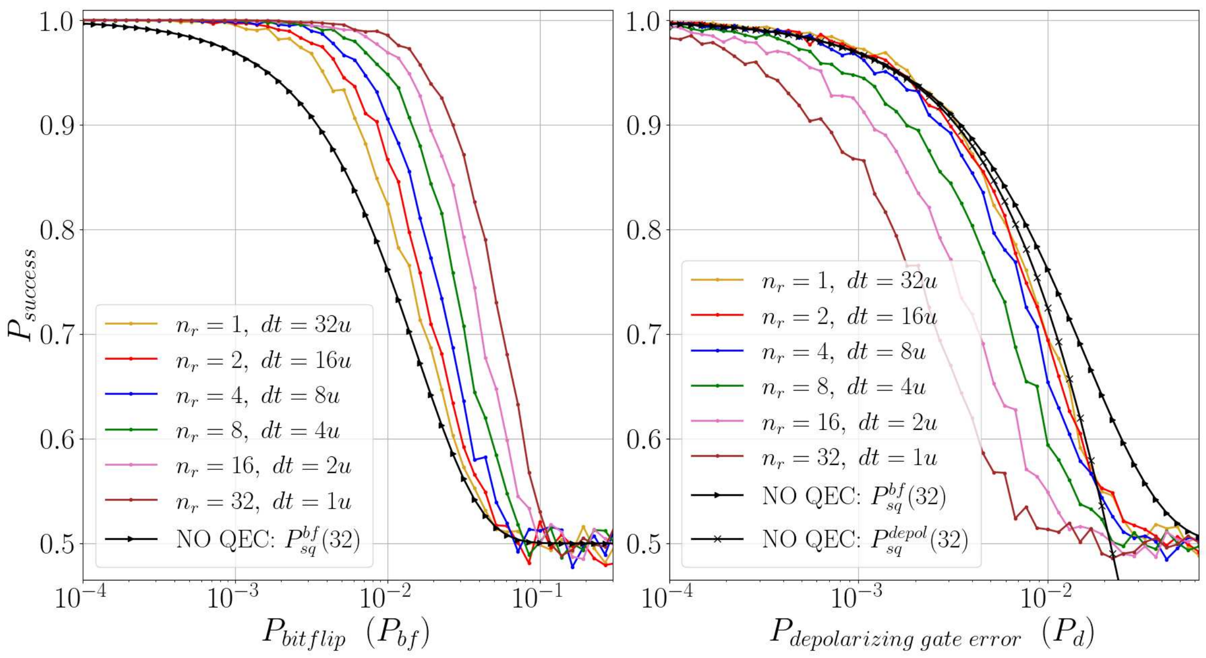

Figure 3.4: *Cyclic [5,1,3] Steane's code simulation for different noise models.* Left: Success probability as a function of bit-flip probability in logscale. Right: Success probability as a function of depolarizing gate error probability. In the legend, '$n_r$' represents the number of rounds, '$dt$' the waiting time between each round. Unencoded single qubit surviving probabilities are in black.

The graph Fig.(3.4)(left) shows a trend of QEC's $P_{success}$ that is closer to $P_{sq}^{bf}(32)$ than the one in Figures (3.2, 3.3)(left). This means that 5-Qubit Steane's code has lower success probabilities at correcting bit-flips than repetition codes. In fact, unlike the previous algorithms, the Steane code is not specific for correcting bit-flips only. However, a logical qubit encoded with this [5,1,3] QEC protocol still outperforms a single qubit in a bit-flip memory up to $P_{bf} = 10\%$, in its configuration with the most rounds (32).
In the case of depolarizing noise model, its gain range is much smaller. The algorithm in this conditions improves the success rate of a qubit in quantum memory only up to a $P_d = 0.2\%$. Above this threshold the QEC is useless compared with both the single qubit in depolarizing and bit-flip memory, $P_{sq}^{bf}(32)$ and $P_{sq}^{depol}(32)$, Fig.(3.4)(left). The presence of many gates in the algorithm evidently outbalances its capability of correcting all three flip channels composing the depolarizing error. For the same reason, we get the best success probabilities against this noise model employing only one round of error correction. While, with bit-flip noise, more rounds are always better. Steane's [5,1,3] code is a great example to introduce stabilizer codes but not as efficient as repetition codes according to these terms of evaluation.

### 3.1.4 Codes performance under realistic conditions

Aiming to test the three codes against each other, we modify the simulation to reflect a situation closer to reality. According to Google's paper [19], most data qubit errors occur during the idle times between one round and the next one, when the ancillas are measured and reset. They estimate the bit-flip probability of any data qubit to be around 4.4%. The new goal is therefore to evaluate how a single round of each algorithm responds to that bit-flip probability on data qubits when the gates used for their implementation are flawed. To do this we employ the depolarizing noise model on all gates and add, after the encoding, a single identity on each data qubit with fixed bit-flip

probability $P_{bf}$. Then, we vary the depolarizing probability parameter within the range [0.01%, 3%] and calculate the algorithm's success probability for each setting. We compare the success probability under these circumstances with the probability of an unencoded qubit surviving one single time unit in a bit-flip memory with $P_{bf}$ error rate: $P_{sq}^{bf}(1) = 1 - P_{bf}$.
We set $P_{bf} = (1 + 0.2)4.4\% = 5.28\%$ because Google's team ultimately finds that the actual measured error rates are 20% higher than their initial predictions.

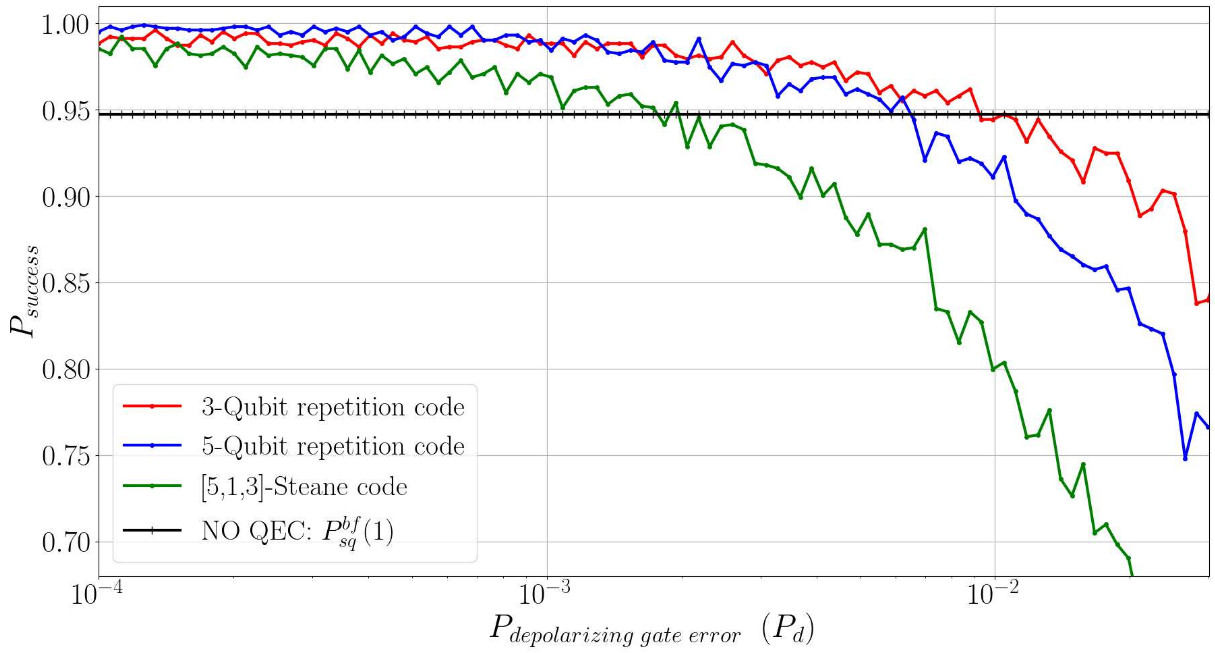

Figure 3.5: *Success probability as a function of the depolarizing error probability for three QEC codes:* 3-Qubit Repetition (red), 5-Qubit Repetition (blue) and Steane's code (green). All gates composing the algorithms are affected by depolarizing noise. A noisy identity with $P_{bf} = 5.28\%$ is applied to the data qubits of every circuit after its encoding. In black the probability of a single qubit surviving one noisy identity.

The trend of the results can be divided into five main zones dependent on $P_d$.

1. $P_d < 0.1\%$. Here all three codes offer an advantage over no correction at all. The 5-Qubit code over than the 3-Qubit code which in turn is more successful than Steane's code. This area allows us to conclude that applying these QEC algorithms could be beneficial even in real NISQ architectures if their error rates are low enough. Furthermore, we have some evidence of the fact that scaling the size of the Repetition code at low errors improves its performance.

2. $P_d \in [0.1\%, 0.2\%]$. Here the two Repetition codes are equivalent and they are both advantageous. The Steane code instead intersects the performance of the unencoded qubit.

3. $P_d \in [0.2\%, 0.7\%]$. In this zone, the 3-Qubit code overcomes the 5-Qubit code while Steane's [5,1,3] code definitively subsides below the single qubit. From here the number of gates of each circuit becomes a preponderant variable for its success.

4. $P_d \in [0.7\%, 1\%]$. Here the only algorithm that is slightly beneficial is the 3-Qubit code. We are getting to the end of the gain zone for QEC.

5. $P_d > 1\%$. Finally, all three codes become useless when error rates are high enough that the algorithms produce more errors than they correct.

## 3.2 Real hardware

Running an algorithm on real quantum architectures involves many more challenges than a simulation. One aspect that can be taken into consideration without relying on the compiler is the actual arrangement of the qubits inside the processor. This is relevant since, usually, the qubits are arranged in a two-dimensional pattern, connected to 1-3 first neighbors. Therefore, care must be taken to match the register's qubits that interact with each other with the physical qubits that are connected to each other. Repetition codes can be implemented on a one-dimensional chain that alternates data qubits with ancillary qubits, thus avoiding this issue. By using cascading CNOTs throughout the whole chain and then, in the opposite direction, other CNOTs controlled by the data qubits targeting the ancillas, this circuit is capable of both encoding and error detection without further swaps. The program to run the algorithms on real hardware has the same structure and commands we used before. Instead of using a simulator with noise models, we send a job to the backend of a quantum computer. IBM provides several superconducting transmon qubits-architectures [20, 21]. The algorithm is executed on real hardware and the output is sent back. The results are organized as counts of each obtained measurement. If the initial qubit state was simple, like $|0\rangle$ or $|1\rangle$, we sum the counts where all data qubits are measured to be 0 or 1 and divide them by the total number of shots. If instead, we stored in the logical qubit a more complex state, like $|+\rangle$, we resort to the Kullback-Leibler divergence, $D_{KL}$ [22]. $D_{KL}$ is an asymmetric type of statistical distance that measures how much a probability distribution, $P$, is different from another, $Q$. For two discrete probability distributions, $D_{KL}$ is defined as:

$$D_{KL}(P||Q) = \sum_i P(i) log_2\left(\frac{P(i)}{Q(i)}\right), \tag{3.3}$$

where $P$ is the theoretical probability distribution of the target state and $Q$ is the measured one. For example, when measuring $|+\rangle$, the results should have $P(0) = P(1) = 0.5$. From the K.-L. Divergence we estimated the success probability as $P_{success} = 2^{-D_{KL}}$. This choice is motivated by the following limit cases.

- When $P$ and $Q$ are exactly the same, i.e. the results match perfectly the target state, $D_{KL} = 0$, so $P_{success} = 2^0 = 1$.
- When $P$ and $Q$ are completely different, i.e. the results don't match at all with the expectations, $D_{KL} \to \infty$, so $P_{success} \to 0$.
- Furthermore, when the encoded state is $|0\rangle$, we have $P(0) = 1, P(1) = 0$ and $Q(0) = P_{success}$. So, $D_{KL}(P||Q) = P(0)log_2(\frac{P(0)}{Q(0)}) = log_2(\frac{1}{Q(0)})) = -log_2(P_{success})$.

The same stands when the encoded state is $|1\rangle$, so it is reasonable to generalize to any state.
Those qubits on the qpu that are not necessary to implement the Repetition code are used as references for qubits without the QEC. We initialize each qubit to a desired state, leave them idle during the execution of the code, and then measure them at the end to evaluate if they stayed the same.
As a starting point, we also test the algorithms without their encoding to see how well they protect classical information. That requires far fewer gates to be stored: an unencoded '0' is just some initialized data qubits, an unencoded '1' can be easily stored performing a not gate on each data qubit after initialization.

### 3.2.1 Cyclic 3-Qubit code on 7-qubit IBM architectures

We first examine the 3-Qubit Repetition code on IBM's 7-Qubit processors: Lagos, Nairobi and Perth. While these are not as powerful as some of IBM's more recent ones, they are useful for testing and devising quantum circuits and they are readily available to the public. The tables below show what we obtained.

| **PROCESSOR: IBM_** | **NAIROBI** | **LAGOS** | **PERTH** |
|---|---|---|---|
| **rounds** | 2 | 2 | 2 |
| **encoding** | no | no | no |
| **initial = target state** | $\lvert 0\rangle\lvert 0\rangle\lvert 0\rangle$ | $\lvert 0\rangle\lvert 0\rangle\lvert 0\rangle$ | $\lvert 0\rangle\lvert 0\rangle\lvert 0\rangle$ |
| **initial_layout** | [0, 1, 2, 3, 4, 5, 6] | [0, 1, 2, 3, 4, 5, 6] | [0, 1, 2, 3, 4, 5, 6] |
| **data** | [0, 3, 6] | [2, 3, 6] | [2, 3, 6] |
| **ancilla** | [1, 5] | [1, 5] | [1, 5] |
| **shots** | 3072 | 3072 | 3072 |
| $P_{success}$ | 92.29% | 96.13% | 92.29% |
| **% corrected states** | 5.73% | 3.52% | 5.86% |
| $\mathbf{D}_{KL}$**(target‖meas.)** | 0.1158 | 0.057 | 0.1158 |

Table 3.1: *2-round 3-Qubit Code on IBM 7-Qubit architectures.* Specifications of the experiments and success probability of correcting unencoded '0'.

| **PROCESSOR: IBM_** | **LAGOS** | | **LAGOS** | |
|---|---|---|---|---|
| **rounds** | 2 | | 2 | |
| **encoding** | yes | | no | |
| **initial state** | $\lvert 111\rangle$ | | $\lvert 1\rangle\lvert 1\rangle\lvert 1\rangle$ | |
| **shots** | 4096 | | 7168 | |
| **P(000)** | 8.01% | | 2.71% | |
| $P_{success}$ **= P(111)** | 68.95% | | 72.67% | |
| $\mathbf{D}_{KL}$**(target‖measure)** | 0.5364 | | 0.4606 | |
| | $\mathbf{P}_{1qubit}^{success}$ | $\mathbf{D}_{KL}$**(t‖m)** | $\mathbf{P}_{1qubit}^{success}$ | $\mathbf{D}_{KL}$**(t‖m)** |
| **TEST QUBIT 1:** $\lvert 1\rangle$ | 95.83% | 0.0615 | 96.05% | 0.0581 |
| **TEST QUBIT 2:** $\lvert 1\rangle$ | 97.71% | 0.0334 | 97.64% | 0.0345 |

Table 3.2: *2-round 3-Qubit Code on IBM Lagos.* Success probability of correcting '1': encoded (left) and unencoded (right).

| **PROCESSOR: IBM_** | **LAGOS** | |
|---|---|---|
| **rounds** | 2 | |
| **encoding** | yes | |
| **initial state** | $\lvert 000\rangle + \lvert 111\rangle$ | |
| **shots** | 4096 | |
| **P(000)** | 39.06% | |
| **P(111)** | 34.79% | |
| $\mathbf{D}_{KL}$**(target‖measure)** | 0.4397 | |
| $P_{success} = 2^{-D_{KL}}$ | 73.73% | |
| | $\mathbf{P}_{1qubit}^{success}$ | $\mathbf{D}_{KL}$**(t‖m)** |
| **TEST QUBIT 1:** $\lvert 0\rangle$ | 93.87% | 0.0913 |
| **TEST QUBIT 2:** $\lvert 0\rangle$ | 93.77% | 0.0928 |

Table 3.3: *2-round 3-Qubit Code on IBM Lagos.* Success probability of correcting an encoded $\lvert +\rangle$ state.

The results show that the success probabilities of the algorithm are lower than those of single unencoded qubits. The error rates of these architectures are not low enough to allow QEC algorithms to correct more errors than they generate: we are in zone (5) of the graph in Figure (3.5). In fact, CNOT errors of IBM Lagos, for example, are between 0.57% and 2.56%. Furthermore, its median readout assignment error is 1.58%, which on the contrary we have neglected in the simulations. We have obtained a successful correction about 93-96% of the time when $|0\rangle$ is saved among the data qubits without encoding, while 73% of the time when $|1\rangle$ is saved unencoded. They are both classical states stored with just a few gates, but saving the latter has far lower success than the former. This is because a quantum channel called Amplitude Damping, caused by interactions with the environment, tends to collapse any generic quantum state into $|0\rangle$. Storing encoded states has a similar success probability, about 70%, which is promising because the ultimate goal of QEC is preserving quantum information, which must necessarily go through encoding. These results do not show the success of the code but are consistent with expectations, being obtained from primitive computers.

### 3.2.2 n-Qubit Repetition code on IBM Cairo

Finally, we try to scale the size of the Repetition code to evaluate the variation of its success probability. The architecture chosen for this experiment is IBM Cairo, which has 27 qubits. We can construct chains containing up to 21 qubits, enough for the 11-Qubit code. We test if each size of the code corrects successfully both an encoded $|+\rangle$ state and an unencoded '1'. We compare them with test qubits left idling for the duration of the codes during each run of the experiment.

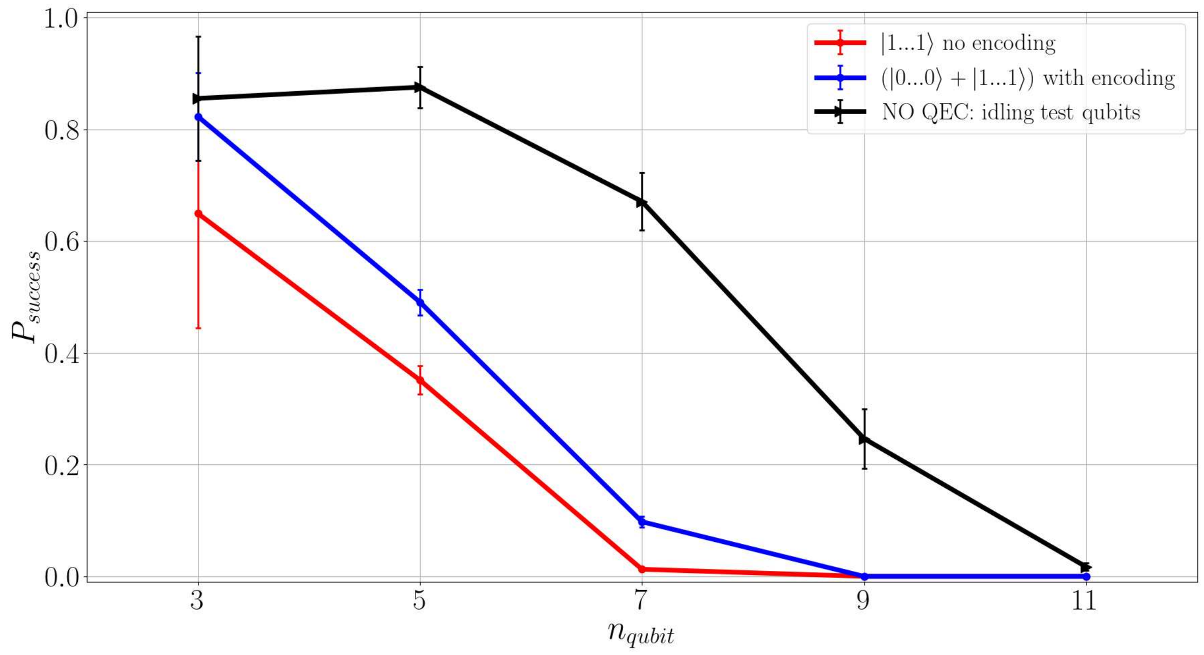

Figure 3.6: *Repetition code on IBM Cairo:* Success probability as a function of the number of data qubits in each code. In red are the results of testing each algorithm without encoding. In blue the success probabilities when encoding and correcting a $|+\rangle$ state. In black the performances of single test-qubits idling in state $|1\rangle$ for the duration of the Repetition code. Each point is obtained as the average of 10 experiments, each consisting of 5120 shots.

Also in this case the architecture is still too prone to errors to have a gain in applying large QEC algorithms. We are again in zone (5) of the graph in Figure (3.5). Gate errors are so high that adding more gates totally crushes the ability of a larger code to correct more errors. Also, a longer algorithm means a longer execution time which causes more errors on the qubits due to decoherence.

This is also pointed out by the trend of single-qubits: the larger the algorithm the longer their idle time. Test qubits left buffering for the duration of the 11-Qubit code remain in state $|1\rangle$ only 1-2% of the time. Such error rates certainly do not permit a useful employment of error-correcting algorithms. The Repetition code on IBM Cairo corrects the encoded $|+\rangle$ state more successfully than the unencoded $|1\rangle$. That is due to the fact that Amplitude Damping incorrectly makes $|0...0\rangle$ more likely to be measured. However, this also means that the encoding does not significantly reduce the performance of the algorithm, which is encouraging for the future application of algorithms safeguarding quantum information. So, this processor is more powerful than the previous ones but there are still improvements to be made to make it resistant to errors via QEC.

# Conclusions

This thesis is meant as an introduction to the field of Quantum Error Correction. First, we formalize the most common errors in quantum computers by defining them as quantum channels and then listing the corresponding physical phenomena. Our study starts from one of the simplest, the Repetition code, inspired by the redundancy method used on classical computers. We expand the discussion by introducing the class of stabilizer codes, a general formalism for the standardized construction of circuits capable of correcting errors of different types. Then, we provide some examples of the implementation of these algorithms. We simulate 3, 5-Qubit Repetition codes and Steane 5-Qubit code varying the error parameters in search of the ranges that allow a gain in the application of QEC. We find that QEC is beneficial in realistic conditions when depolarizing errors afflict gates less than 0.7-1% of the time for Repetition codes and 0.2% for Steane's code. Furthermore, when the probability of this error is lower than 0.1% increasing the size of the $n$-Qubit Repetition code leads to proportionally higher correction performances. We finally test the Repetition codes on real quantum hardware. When trying the 3-Qubit code on 7-qubit architectures, we find them to be too primitive to employ QEC. When scaling the $n$-Qubit code on IBM Cairo, our results suggest that also in this case error rates are still too high to benefit from QEC. In spite of our results, experimental demonstrations of effective Quantum Error Correction have been made. In 2023, Google's team has shown even an exponential suppression of code failure probability as the size of the Repetition code increases [19].
The takeaway of this work is that an efficient Quantum Error Correction is fundamental to achieve in order to develop fault-tolerant quantum computers for quantum supremacy. Therefore, it is necessary to expand the research in this area of Quantum Information Theory.

# Bibliography

[1] R. P. Feynman, “Simulating physics with computers,” vol. 21, no. 6, pp. 467–488.

[2] G. Benenti, G. Casati, D. Rossini, and G. Strini, *Principles of quantum computation and information: a comprehensive textbook.* World Scientific, 2019.

[3] M. A. Nielsen and I. L. Chuang, *Quantum computation and quantum information.* Cambridge university press, 2010.

[4] D. Deutsch, “Quantum theory, the church–turing principle and the universal quantum computer,” *Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences*, vol. 400, no. 1818, pp. 97–117, 1985.

[5] D. Deutsch and R. Jozsa, “Rapid solution of problems by quantum computation,” *Proceedings of the Royal Society of London. Series A: Mathematical and Physical Sciences*, vol. 439, no. 1907, pp. 553–558, 1992.

[6] P. W. Shor, “Algorithms for quantum computation: discrete logarithms and factoring,” in *Proceedings 35th annual symposium on foundations of computer science*, pp. 124–134, Ieee, 1994.

[7] L. K. Grover, “A fast quantum mechanical algorithm for database search,” in *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, pp. 212–219, 1996.

[8] J. Tilly, H. Chen, S. Cao, D. Picozzi, K. Setia, Y. Li, E. Grant, L. Wossnig, I. Rungger, G. H. Booth, *et al.*, “The variational quantum eigensolver: a review of methods and best practices,” *Physics Reports*, vol. 986, pp. 1–128, 2022.

[9] J. Preskill, “Quantum computing in the nisq era and beyond,” *Quantum*, vol. 2, p. 79, 2018.

[10] S. J. Devitt, W. J. Munro, and K. Nemoto, “Quantum error correction for beginners,” *Reports on Progress in Physics*, vol. 76, no. 7, p. 076001, 2013.

[11] D. Gottesman, “Theory of fault-tolerant quantum computation,” *Physical Review A*, vol. 57, no. 1, p. 127, 1998.

[12] L. E. Ballentine, *Quantum mechanics: a modern development.* World Scientific Publishing Company, 2014.

[13] U. Fano, “Description of states in quantum mechanics by density matrix and operator techniques,” *Reviews of modern physics*, vol. 29, no. 1, p. 74, 1957.

[14] K. Konishi and G. Paffuti, *Quantum mechanics: a new introduction.* OUP Oxford, 2009.

[15] J. Von Neumann, *Mathematical foundations of quantum mechanics: New edition*, vol. 53. Princeton university press, 2018.

[16] K. Kraus, A. Böhm, J. D. Dollard, and W. Wootters, *States, Effects, and Operations Fundamental Notions of Quantum Theory: Lectures in Mathematical Physics at the University of Texas at Austin.* Springer, 1983.

[17] G. Bartolucci, "github.com/giovibart/bartolucci_giovanni_qiskit_thesis." `https://doi.org/10.5281/zenodo.8330753`, 2023.

[18] D. Gottesman, *Stabilizer codes and quantum error correction.* California Institute of Technology, 1997.

[19] "Exponential suppression of bit or phase errors with cyclic error correction," *Nature*, vol. 595, no. 7867, pp. 383–387, 2021.

[20] M. Kjaergaard, M. E. Schwartz, J. Braumüller, P. Krantz, J. I.-J. Wang, S. Gustavsson, and W. D. Oliver, "Superconducting qubits: Current state of play," *Annual Review of Condensed Matter Physics*, vol. 11, pp. 369–395, 2020.

[21] P. Krantz, M. Kjaergaard, F. Yan, T. P. Orlando, S. Gustavsson, and W. D. Oliver, "A quantum engineer's guide to superconducting qubits," *Applied physics reviews*, vol. 6, no. 2, 2019.

[22] S. Kullback, *Information theory and statistics.* Courier Corporation, 1997.

[23] F. Tacchino, A. Chiesa, S. Carretta, and D. Gerace, "Quantum computers as universal quantum simulators: state-of-the-art and perspectives," *Advanced Quantum Technologies*, vol. 3, no. 3, p. 1900052, 2020.

[24] A. J. Daley, I. Bloch, C. Kokail, S. Flannigan, N. Pearson, M. Troyer, and P. Zoller, "Practical quantum advantage in quantum simulation," *Nature*, vol. 607, no. 7920, pp. 667–676, 2022.

[25] M. A. Rowe, D. Kielpinski, V. Meyer, C. A. Sackett, W. M. Itano, C. Monroe, and D. J. Wineland, "Experimental violation of a bell's inequality with efficient detection," *Nature*, vol. 409, no. 6822, pp. 791–794, 2001.