UNIVERSITÀ DEGLI STUDI DI MILANO

FACOLTÀ DI SCIENZE E TECNOLOGIE

Master's degree in Physics

Development of an open-source calibration framework for superconducting qubits

Supervisor:
Prof. Dr. Stefano Carrazza

Co-supervisor:
Dr. Alessandro Candido

Co-supervisor:
Dr. Andrea Pasquale

Co-supervisor:
Dott. Edoardo Pedicillo

Elisa Stabilini
Matricola n° 28326A
A.A. 2024/2025

A coloro che

mi hanno ispirato e supportato.

Contents

Abstract

Achieving high-fidelity gate operations is a central challenge in gate-based quantum computing. In superconducting qubit platforms, this requires both long coherence times and low-error gate implementations to ensure scalable, fault-tolerant computation. This thesis, carried out within the Qibo collaboration, specifically in the Qibocal and Qibolab groups, focuses on the development of calibration routines within the `Qibocal` library, with the aim of improving both single- and two-qubit gate fidelities.

Chapter 1 introduces the theoretical foundation of superconducting qubits, beginning with Josephson junctions and their role in defining the qubit subspace. It then reviews early qubit implementations, from Cooper Pair Boxes to flux-tunable transmons, and discusses how these systems support quantum gate operations. The chapter concludes with a description of qubit readout mechanisms and sources of decoherence.

In Chapter 2, I describe the experimental setup on which I worked: a superconducting qubit chip developed by QuantWare and operated at the Technology Innovation Institute (TII), using control electronics provided by Quantum Machines. The chapter also covers the calibration of single-qubit gates using the routines available in `Qibocal`. These routines were initially used to gain confidence with the calibration process before starting the core of the project. The results obtained are summarized in a table reporting the main performance metrics, including readout fidelity, coherence times (T_1, T_2), and average gate infidelity measured via Randomized Benchmarking.

The work described in Chapter 3 concerns an attempt to automate the recalibration of single-qubit gates using modern optimization libraries. In particular, I used tools such as `Optuna` and `CMA-ES` to tune key parameters of the $R_X(\pi)$ gate. This effort was inspired by the ORBIT protocol (Optimized Randomized Benchmarking for Immediate Tune-up), which relies on Randomized Benchmarking to evaluate fidelity as a cost function. While improvements were achieved, the work highlighted several challenges, primarily the high cost of fidelity evaluation and the lack of stability in convergence when broader parameter searches were used. On the other hand, this work also demonstrates the potential of such approaches: with more advanced tools for qubit control, the automation of the recalibration process is not only feasible but also highly desirable, particularly in systems involving a large number of qubits.

Chapter 4 describes some enhancements and additions to the calibration infrastructure. First, native support for the $R_X(\pi/2)$ gate was introduced in `Qibolab` and its calibration procedure added to `Qibocal`, replacing less accurate approximations. Second, a protocol for correcting flux pulse distortions was implemented using the Cryoscope method [1]. This correction improves the fidelity of flux-based gates such as CNOT and iSWAP, which are particularly sensitive to control-line imperfections.

Chapter 1

Superconducting qubits

The electronics that modern computers rely on contain components that operate based on quantum mechanics; however, their computational processes are still governed by classical laws. For this reason, they are referred to as "classical computers."

Quantum computing emerged from Richard Feynman's idea that simulating quantum systems efficiently requires quantum mechanical resources [2]. Classical computers struggle to model complex quantum interactions due to the exponential growth of computational requirements with system size, making exact simulations infeasible beyond small systems [3]. Quantum computers, taking advantage of quantum mechanics phenomena like superposition and entanglement, offer a natural framework for such simulations and have been demonstrated to provide exponential speedups for certain quantum systems [4].

Beyond quantum simulation, current theoretical advancements suggest that quantum algorithms can outperform classical counterparts in solving specific problems [5].

1.1 Introduction

The physical realization of quantum computing necessitates the development of a system capable of functioning as quantum bits (qubits).
Similarly to classical logic, where the bits 0 and 1 are associated with two physical levels, typically represented by high and low voltage states, a qubit can, to a first approximation, be considered as a two-level physical system.

Mathematically, this system is described with a two-dimensional complex Hilbert space, where the basis states $|0\rangle$ and $|1\rangle$ correspond to two orthonormal vectors. Any general state of the qubit can be expressed as a superposition of these basis states:

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle \rightarrow \begin{pmatrix}\alpha\\ \beta\end{pmatrix}, \tag{1.1}$$

where $\alpha, \beta \in \mathbb{C}$. If the normalization condition $|\alpha|^2 + |\beta|^2 = 1$ holds, the state $|\psi\rangle$ represents a qubit. The basis $\{|0\rangle, |1\rangle\}$ is called computational basis and the information is stored in the complex numbers α and β.

A possible geometric representation of qubit states is given by the Bloch sphere, which offers a visualization of two-level quantum systems as vectors on a unit sphere. A qubit state is depicted as a vector originating from the center of the sphere, with the computational basis states $|0\rangle$ and $|1\rangle$ positioned at the north and south poles, respectively. The axis connecting these states defines the z-axis. The transverse x- and y- axes correspond to the equal superposition states $|\pm\rangle = \frac{1}{\sqrt{2}}(|0\rangle \pm |1\rangle)$ and $|\pm i\rangle = \frac{1}{\sqrt{2}}(|0\rangle \pm i\,|1\rangle)$, respectively.
A vector of unit length on the Blooch sphere is characterized by the polar angle θ, with $0 \leq \theta \leq \pi$ and the azimuthal angle φ, with $0 \leq \varphi \leq 2\pi$, each unit vector represent a possible pure state of the qubit.

The qubit states $|0\rangle$ and $|1\rangle$ can also be associated with energy eigenstates of a physical system, where $|0\rangle$ represents the ground state with energy E_0 and $|1\rangle$ represents the excited state with

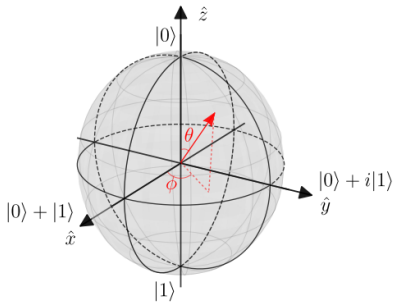


Figure 1.1: Example of qubit state representation on the Bloch sphere
Source: Metrology of Quantum Control and Measurement in Superconducting Qubits [6]

energy E_1, assuming $E_0 < E_1$. In this energy eigenbasis, the Hamiltonian of the qubit is given by

$$\hat{H}_q = E_0 \left|0\right\rangle \left\langle 0\right| + E_1 \left|1\right\rangle \left\langle 1\right| = \begin{pmatrix} E_0 & 0 \\ 0 & E_1 \end{pmatrix}. \tag{1.2}$$

Since only energy differences are physically relevant, it is possible to redefine the zero-point energy by subtracting the constant term $E_0(\left|0\right\rangle \left\langle 0\right| + \left|1\right\rangle \left\langle 1\right|)$, leading to the simplified Hamiltonian

$$\hat{H}_q = (E_1 - E_0) \left|1\right\rangle \left\langle 1\right| = \hbar\omega_q \left|1\right\rangle \left\langle 1\right| = \hbar\omega_q \hat{\sigma}^+ \hat{\sigma}^- = \begin{pmatrix} 0 & 0 \\ 0 & \hbar\omega_q \end{pmatrix}, \tag{1.3}$$

where $\omega_q = (E_1 - E_0)/\hbar$ is the qubit transition frequency, and was used the relation $\hat{\sigma}^+\hat{\sigma}^- = \left|1\right\rangle \left\langle 1\right|$. For convenience, the Hamiltonian can also be rewritten in terms of the Pauli z-matrix, $\hat{\sigma}_z$, by adding a term proportional to the identity:

$$\hat{H}_q = \hbar\omega_q \left|1\right\rangle \left\langle 1\right| - \frac{\hbar\omega_q}{2}\mathbb{I} = \begin{pmatrix} -\frac{\hbar\omega_q}{2} & 0 \\ 0 & \frac{\hbar\omega_q}{2} \end{pmatrix} = -\frac{\hbar\omega_q}{2}\hat{\sigma}_z. \tag{1.4}$$

Qubits can be implemented through various physical mechanisms; however, their practical realization remains a significant challenge due to their susceptibility to environmental interactions, which lead to decoherence and reduce their coherence time. Despite the diversity of possible physical implementations, any functional quantum computing system must satisfy a set of fundamental criteria. These requirements, known as the DiVincenzo criteria, establish the essential conditions for the construction and operation of a viable quantum computer [7], [8]:

1. The physical system used as a quantum computer must comprise a set of qubits, meaning that the quantum system must be well-characterized, and scalable such that quantum computing can be realized.
2. It must be possible to initialize the qubits in a reliable state, such as the ground state.
3. The coherence time of the qubits must be longer than the typical gate time, ideally should be possible to perform $> 10^4$ operations, which is the number that allows for realizing effective error corrections.
4. It must be possible to implement a universal set of quantum gates.
5. It must be possible to measure the qubits in the computational basis.

In the present work, we will focus on superconducting qubits, which constitute the hardware we have worked on and where the experiments were conducted. However, several of the experiments described later can also be implemented using different physical systems.

1.2 Transmon qubits

In this section, we provide a review of the structure and operation of superconducting transmon qubits. The content of this section is based on the *Quantum Information Science* manual [8], *The Metrology of Quantum Control and Measurement in Superconducting Qubits* [6], the notes from quantum computing lectures held by Professor Olivares [9], and the original transmon paper [10].

1.2.1 Josephson Junctions

The Josephson junction (JJ) is formed by a thin oxide layer positioned between the two superconductors which acts as an insulating barrier. An example of Josephson junction is shown in figure (1.2), a side view in image (1.2a) and a top view in image (1.2b).

(a) (b)

Figure 1.2: Figure (1.2a): Side view of a Josephson junction, the two superconducting pads are colored in red and blue and indicated by the letter S. In grey, indicated with the letter I, is the insulating barrier of oxide. The superconductors and the oxide are layered over a substrate.
Figure (1.2b): Electron microscope image of a 2μm $\times$ 2μm cross-type junction: I. Josephson junction. II. Base electrode. III. Contact to the top electrode.
Source:`https://www.ims.kit.edu/english/2551.php`

Superconductivity is a phenomenon observed in certain materials where, when cooled well below a critical temperature T_c, which depends on the material, their electrical resistance drops to zero, allowing them to behave as perfect conductors. According to the BCS (Bardeen-Cooper-Schrieffer) theory, superconductivity arises, from the formation of Cooper pairs, which are bound states of electrons with opposite momenta and spins. These pairs collectively form a macroscopic quantum state described by a single waveform $\psi(\mathbf{r})$ which can be expressed as

$$\psi(\mathbf{r}) = \sqrt{\rho(\mathbf{r})}e^{i\theta(\mathbf{r})}, \tag{1.5}$$

where $\rho(\mathbf{r})$ is the density of Cooper pairs in the metal, which is typically uniform in the bulk of the superconductor, and $\theta(\mathbf{r})$ is the macroscopic phase of the superconducting wavefunction.

For this reason, the wavefunctions on the two sides of the JJ can be denoted as

$$\psi_1(\mathbf{r},t) = \sqrt{\rho_1(\mathbf{r},t)}e^{i\theta_1(\mathbf{r},t)}, \quad \psi_2(\mathbf{r},t) = \sqrt{\rho_2(\mathbf{r},t)}e^{i\theta_2(\mathbf{r},t)}. \tag{1.6}$$

The dynamics of the system can be described by the two equations

$$i\hbar\frac{d\psi_1}{dt} = E_1\psi_1 + K\psi_2, \tag{1.7}$$

$$i\hbar\frac{d\psi_2}{dt} = E_2\psi_2 + K\psi_1. \tag{1.8}$$

By substituting the expression of ψ_i into the Schrödinger equation (1.7), (1.8) we obtain

$$\frac{d\rho_1}{dt} = \frac{2K}{\hbar}\sqrt{\rho_1\rho_2}\sin(\theta_2 - \theta_1), \tag{1.9}$$

$$\frac{d\rho_2}{dt} = -\frac{2K}{\hbar}\sqrt{\rho_1\rho_2}\sin(\theta_2 - \theta_1). \tag{1.10}$$

Since the derivative of the charge density is current, from equations (1.9) and (1.10) we obtain the first Josephson equation

$$I = I_c \sin\phi, \tag{1.11}$$

where $I_c = \frac{2K}{\hbar}\sqrt{\rho_1\rho_2}$ is the critical current and ϕ is the superconducting phase difference $\theta_2 - \theta_1$.

Instead, from the real part of the Schrödinger equation (1.7), (1.8) and a few calculations, we obtain the second Josephson equation

$$\frac{d\phi}{dt} = \frac{2e}{\hbar}V(t). \tag{1.12}$$

which can be rewritten as

$$\frac{d\phi}{dt} = \frac{2\pi}{\Phi_0}V(t). \tag{1.13}$$

where $\Phi_0 = \frac{h}{2e}$ is the superconducting flux quantum, with h is the Planck's constant and $2e$ is the charge of a Cooper pair.

The time derivative of the first Josephson equation (1.11) yields:

$$\dot{I_J} = I_C \cos\phi \frac{\partial\phi}{\partial t}, \tag{1.14}$$

equation (1.14) suggests a nonlinear relation between the current the voltage. Using the Josephson voltage-phase relation and the fact that $\dot{I} = \frac{V}{L}$ it is possible to define an effective nonlinear inductance for the Josephson junction:

$$L_J = \frac{1}{\cos\phi}\frac{\Phi_0}{2\pi I_c}. \tag{1.15}$$

In addition to the inductive behavior the Josephson junction also exhibits capacitive properties due to its inherent capacitance C_J with a corresponding energy of

$$E_{C_J} = \frac{Q^2}{2C_J}, \tag{1.16}$$

.

Fom equation (1.15) it is possible to compute the energy stored in the nonlinear inductance as

$$E_{L_J} = \int_0^t \mathrm{d}\tau I_J(\tau)V(\tau) = \int_0^t \mathrm{d}\tau I_c \sin\phi(\tau)\frac{\partial\phi(\tau)}{\partial\tau}\frac{\Phi_0}{2\pi} \tag{1.17}$$

$$= \frac{\Phi_0 I_c}{2\pi}(1-\cos\phi) = E_J(1-\cos\phi), \tag{1.18}$$

where E_J represents the energy due to the behavior of the junction as a nonlinear inductor.

1.2.2 CPB qubit

A first example of superconducting qubit is the Cooper Pair Box (CPB), which consists of a small superconducting island connected to a reservoir of superconducting electrons through a Josephson junction [11], with an external gate voltage controlling the charge state. The circuit corresponding to CPB is similar to the circuit of a parallel resonator where the linear inductance is substituted by a Josephson junction which simply acts as a nonlinear inductance.

Combining the energy associated to the capacitance C and the energy of the Josephson junction (1.17) it is possible to write the classical Hamiltonian of the circuit

$$H_J = 4E_C n^2 - E_J \cos\phi, \tag{1.19}$$

where the constant term was ignored as it acts simply as a constant offset without influencing the dynamics of the system and where E_C is the charging energy defined as

$$E_C = \frac{e^2}{2C}. \tag{1.20}$$

To control the number of Cooper pairs on the island, it is possible to connect a DC voltage source V_g to the system through a gate capacitor C_g, as shown in Figure (1.3b). When $V_g = 0$, both the gate and qubit capacitors remain uncharged. As V_g increases, a charge $Q_g = C_g V_g$ accumulates on the gate capacitor, inducing an equal and opposite charge on the island to maintain charge neutrality. When $Q_g \approx 2e$, a Cooper pair tunnels from the reservoir to the island, discharging the qubit capacitor.

The presence of the external voltage source introduces an additional control parameter for the number of Cooper pairs on the island, modifying the system's Hamiltonian. The resulting Hamiltonian of the Cooper Pair Box (CPB) takes the form

$$\hat{H} = 4E_C(\hat{n} - n_g)^2 - E_J \cos\hat{\phi}, \tag{1.21}$$

(a) (b)

Figure 1.3: Figure (1.3a): corresponding circuit of a CPB which consists of Josephson junction shunted by a capacitor. Surce: [8]. Figure (1.3b): electrical circuit of a CPB capacitively coupled to a DC voltage source through a capacitor C_g. Source: [8].

where $n_g = \frac{C_g V_g}{2e}$ represents the normalized gate charge.

A key feature of the system is the presence of a Josephson junction in place of a linear inductance. Unlike a standard LC circuit—corresponding to a quantum harmonic oscillator—this results in a non-equidistant energy level spectrum. In particular, the energy levels are anharmonic, allowing the first two levels to be spectrally isolated from the higher ones. This anharmonicity enables the use of the subspace spanned by the ground state $|0\rangle$ and the first excited state $|1\rangle$ as a qubit.

The qubit frequency is defined as the frequency associated with the energy difference between these two states: $f_{01}(n_g) = f_q = \frac{(E_1 - E_0)}{h}$. This frequency can be tuned by varying the externally applied DC voltage, which modifies the system's parameters and, consequently, its energy level spacing.

In practice, however, fluctuations in the offset charge caused by the random motion of nearby charges or coupling to the electromagnetic environment introduce dephasing in the qubit. To minimize this sensitivity to charge noise, the qubit can be biased at the charge sweetspot, defined by $n_g = \frac{1}{2}$. At this point, the qubit transition frequency reaches a local maximum and the first derivative of the energy levels with respect to n_g vanishes. As a result, since the charge dispersion has no slope there, linear noise contributions cannot change the qubit transition frequency.

1.2.3 Transmon qubit

One of the main drawbacks of the Cooper Pair Box (CPB) qubit, which ultimately led to its replacement by other qubit architectures, is its limited coherence time. The transmon qubit was introduced specifically to address this issue, with the goal of improving the dephasing time of the CPB. The key idea behind the transmon is to reduce the sensitivity of the energy levels to fluctuations in the gate charge—effectively flattening the energy bands—by increasing the ratio between the Josephson energy $E_J/E_C \gg 1$, this architecture was first proposed in [10], the first and more straightforward method to increase this ratio is to enlarge the capacitance of the qubit, which reduces the charging energy E_C.

Since the CPB and the transmon qubit have the same electrical circuit they are also described by the same Hamiltonian (1.19). The difference is that in this case the transmon satisfies the condition $E_J/E_C \gg 1$ it is possible to expand the cosine term in (1.19) with a Taylor series and neglect the higher order terms:

$$\hat{H} \approx 4E_C\hat{n}^2 + \frac{1}{2}E_J\hat{\phi}^2 - \frac{E_J}{4!}\hat{\phi}^4, \tag{1.22}$$

where the last term, proportional to $\hat{\phi}^4$, makes the potential of the transmon slightly anharmonic.

As happens in the standard harmonic oscillator case, the operators $\hat{\phi}$ and $\hat{n}$ satisfy the canonical commutation relation $[\hat{\phi}, \hat{n}] = i\mathbb{I}$, it is possible to introduce the raising and lowering operators $\hat{b}, \hat{b}^\dagger$ as

$$\hat{\phi} = \sqrt{\xi}(\hat{b} + \hat{b}^\dagger), \qquad \hat{n} = -\frac{i}{2\sqrt{\xi}}(\hat{b} - \hat{b}^\dagger), \tag{1.23}$$

where $\xi = \sqrt{2E_C/E_J}$.

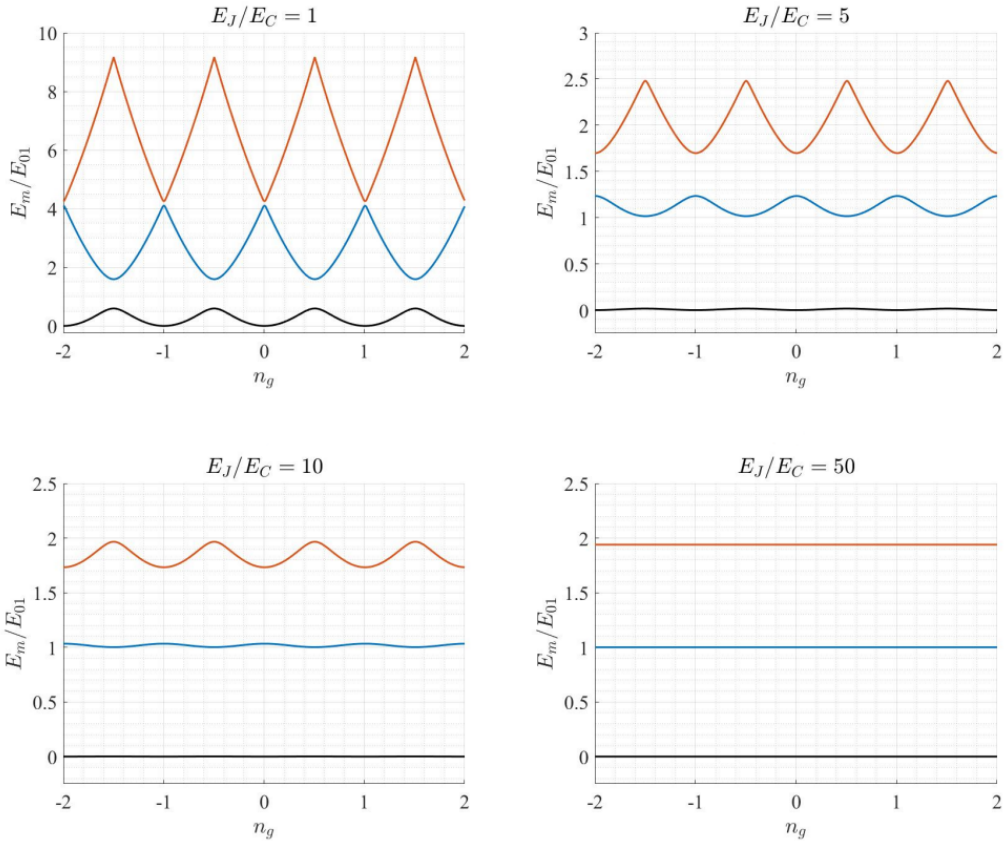


Figure 1.4: Energy levels E_m of the hamiltonian as a function of n_g normalized with respect to E_{01} for different values of the ratio E_J/E_C.
Credits: [12]

Substituting equations (1.23) in the Hamiltonian, equation (1.22) becomes

$$\hat{H} = \sqrt{8E_J E_C}\hat{b}^\dagger\hat{b} - \frac{E_C}{12}(\hat{b} + \hat{b}^\dagger)^4. \tag{1.24}$$

Given equation (1.24) it is possible to solve the eigenvalue problem $\hat{H}\,|k\rangle = E_k\,|k\rangle$ and calculate the energy levels E_k. The first term of Hamiltonian (1.24) is the harmonic oscillator contribution with eigenstates $|k\rangle$ and eigenvalues $\sqrt{8E_J E_C}k$. Since $E_C \ll E_J$, the second term $\hat{V} = -\frac{E_C}{12}(\hat{b} + \hat{b}^\dagger)^4$ represents a small perturbative contribution to the Hamiltonian and can be treated using perturbation theory. The first-order correction to the energy levels is given by the diagonal matrix elements of the perturbation operator: $\Delta E_k^{(1)} = \langle k|\hat{V}|k\rangle$. It is possible to verify that $\langle k|\,\hat{V}\,|k\rangle = -\frac{E_C}{12}(6k^2 + 6k + 3)$. Thus the eigenenergies of the transmon Hamiltonian are

$$E_k \approx \sqrt{8E_J E_C}k - \frac{E_C}{2}(k^2 + k). \tag{1.25}$$

As mentioned before, the qubit frequency is defined as $f_q = f_{01} = (E_1 - E_0)/h$ which yields

$$f_0 1 \approx (\sqrt{8E_J E_C} - E_C)/h \tag{1.26}$$

As explained at the beginning of this section, a large E_J/E_C ratio makes the transmon qubit significantly less sensitive to charge noise. However, this improvement comes at the expense of reduced anharmonicity in the energy level spectrum. The anharmonicity η is defined as the difference between the second and first transition energies, relative to the first transition energy:

$$\eta = \frac{(E_2 - E_1) - (E_1 - E_0)}{\hbar} = \omega_{12} - \omega_{01}. \tag{1.27}$$

For a transmon, the anharmonicity η is negative, reflecting the fact that the level spacing decreases with increasing energy. Ideally, the absolute value $|\eta|$ should be sufficiently large to allow external

microwave drives to selectively address the $|0\rangle \leftrightarrow |1\rangle$ transition without inadvertently exciting higher-energy states. Typical values for the frequencies of superconducting qubits lie between 4 - 8 GHz with an anharmonicity ranging from 100 to 300 MHz.z

1.2.4 Flux-tunable transmon

To implement certain two-qubit gate schemes, such as swap interactions, it is essential to tune the qubit frequency. A common approach to achieving this is by adding an extra junction to the transmon, the most common configuration is the SQUID (Superconducting Quantum Interference Device). In the SQUID configuration two Josephson junctions are connected in parallel on a superconducting loop, as shown in Figure (1.5).

Starting from the Hamiltonian of the single Josephson junction it is possible to write the Hamiltonian of a SQUID:

$$\hat{H} = 4E_C\hat{n}^2 - E_{J1}\cos\hat{\phi}_1 - E_{J2}\cos\hat{\phi}_2, \tag{1.28}$$

where E_{J1} and E_{J2} are the Josephson energies of the two junctions, and the operators $\hat{\phi}_1$ and $\hat{\phi}_2$ are the phase differences across the junctions.

Because of the quantization of the magnetic flux through the SQUID, the quantities $\hat{\phi}_1$ and $\hat{\phi}_2$ are not independent. In particular, as show in [8], the difference between $\hat{\phi}_1$ and $\hat{\phi}_2$ follows the following relation:

$$\hat{\phi_1} - \hat{\phi_2} = \frac{2\pi}{\Phi_0}\Phi_{\text{ext}}\mathbb{I}(\text{mod}2\pi), \tag{1.29}$$

where Φ_{ext} is the flux of the external magnetic field defined as the integral of the magnetic field over the SQUID area.

Equation (1.29) can be simplified and rewritten as

$$\hat{H} = 4E_C\hat{n}^2 - E_J(\Phi_{\text{ext}})\cos\hat{\varphi}, \tag{1.30}$$

where $\hat{\varphi} = \frac{\hat{\phi_1}+\hat{\phi_2}}{2}$, and the Josephson energy is flux-dependent:

$$E_J(\Phi_{\text{ext}}) = (E_{J1} + E_{J2})\left|\cos\left(\pi\frac{\Phi_{\text{ext}}}{\Phi_0}\right)\right|\sqrt{1 + d^2\tan^2\left(\pi\frac{\Phi_{\text{ext}}}{\Phi_0}\right)}, \tag{1.31}$$

where $d = \frac{E_{J1}-E_{J2}}{E_{J1}+E_{J2}}$ is the relative junction asymmetry.

Then, it's easy to see that the frequency f_q of a two-junction transmon depending on the magnetic flux $\Phi_q(t)$ through the SQUID loop, the relation between the magnetic flux and qubit frequency is given by [13]

$$f_q(\Phi) = \left(f_q^{\max} + \frac{E_C}{h}\right)\sqrt[4]{d^2 + (1-d^2)\cos^2\left(\pi\frac{\Phi}{\Phi_0}\right)} - \frac{E_C}{h}, \tag{1.32}$$

where $f_q^{\max} = (\sqrt{8E_CE_J} - E_C)/h$ is the maximum qubit frequency. For symmetric junctions this relation can be approximated by [1]

$$f_q(\Phi_q) \approx \frac{1}{h}\left(\sqrt{8E_JE_C\cos\left(\pi\frac{\Phi_q}{\Phi_0}\right)} - E_C\right). \tag{1.33}$$

The main drawback of having the qubit frequency depending on the magnetic flux is that the qubit frequency becomes also sensitive to spurious magnetic currents. To mitigate this issue, the qubit can be biased at a special value of external magnetic flux, known as flux sweetspot where the first derivative of the qubit frequency with respect to flux vanishes. As a consequence small fluctuations in flux lead only to second-order changes in frequency, thereby reducing dephasing due to flux noise.

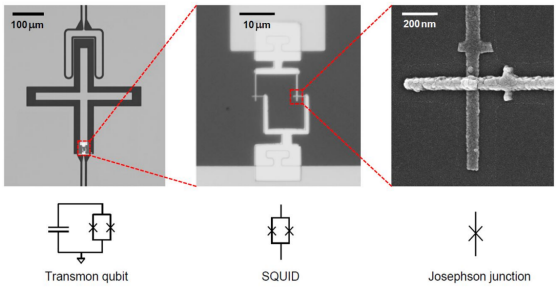
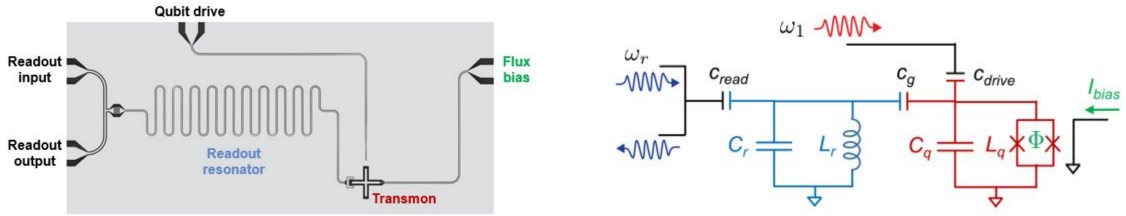


Figure 1.5: Images of a flux tunable transmon qubit. From left to right: the flux tunable transmon qubit, consisting of a large cross-shaped capacitance in parallel with a SQUID to ground, and its corresponding circuit. A zoom in of the SQUID (center), a single Josephson junction (right). Source: [12]

1.3 Qubit readout

Up to this point, only the physical structure of a transmon qubit was discussed. However, in order to perform quantum computing, it is essential to be able to control and measure its quantum state. One approach to achieve this is to capacitively couple the qubit to both a drive line and a readout line. The drive line is used to manipulate the state of the qubit, while the readout line is employed to measure it. An example of such a system is shown in Figure (1.6), along with the corresponding circuit diagram.



(a) (b)

Figure 1.6: Figure (1.6a) shows an example of a single transmon device. Figure (1.6b) shows the equivalent lumped element circuit model of the device in (1.6a), in blue is represented the resonator circuit while in red the transmon qubit circuit. Source: [12]

As mentioned in the introduction of this section, in circuit quantum electrodynamics (cQED), the qubit state is measured via a dispersive interaction between a qubit and a far-detuned microwave resonator.
In Figure (1.6b), in blue is represented the resonator circuit capacitively coupled to the (red) qubit that is used for the readout of the qubit state. The resonator circuit is characterized by an inductance L_r and a capacitance C_r, then the characteristic frequency is $\omega_r = 1/\sqrt{L_r C_r}$.

The classical Hamiltonian of the resonator can be written as

$$H_r = \frac{Q^2}{2C_r} + \frac{1}{2}C\omega_r^2\Phi^2, \tag{1.34}$$

where Φ is the generalized flux, defined as the time integral of the voltage across the capacitor:

$$\Phi(t) = \int_{-\infty}^{t} V(t')dt'. \tag{1.35}$$

.

The quantization of the Hamiltonian involves replacing the classical conjugate variables with their corresponding hermitian operators $\hat{\Phi}$ and $\hat{Q}$ with $[\hat{\Phi}, \hat{Q}] = i\hbar\mathbb{I}$ which leads to

$$\hat{H}_r = \frac{\hat{Q}^2}{2C_r} + \frac{1}{2}C\omega_r^2\hat{\Phi}^2. \tag{1.36}$$

To find the eigenvalues and eigenvectors of $\hat{H}_r$ it is often convenient to introduce the raising and lowering operators, $\hat{a}$ and $\hat{a}^\dagger$ which satisfy $[\hat{a}, \hat{a}^\dagger] = \mathbb{I}$. These two non-hermitian operators are defined as

$$\hat{\Phi} = \Phi_{\text{zpf}}(\hat{a} + \hat{a}^\dagger), \tag{1.37}$$

$$\hat{Q} = -iQ_{\text{zpf}}(\hat{a} - \hat{a}^\dagger), \tag{1.38}$$

where $\Phi_{\text{zpf}} = \sqrt{\frac{\hbar}{2C_r\omega_r}}$ and $Q_{\text{zpf}} = \sqrt{\frac{C_r\hbar\omega_r}{2}}$ are the zero-point fluctuations. It is then possible to write the Hamiltonian of a microwave resonator in a quantum regime:

$$\hat{H}_r = \hbar\omega_r \left(\hat{a}^\dagger\hat{a} + \frac{1}{2}\right), \tag{1.39}$$

The system of the capacitively coupled transmon and resonator is built in such a way that there is a maximum coupling between the qubit and the resonator. The Hamiltonian of the system reads

$$\hat{H} = \hbar\omega_r\hat{a}^\dagger\hat{a} + 4E_C(\hat{n} + \frac{C_{\text{read}}\hat{V}}{2e})^2 - E_J\cos\hat{\phi} \tag{1.40}$$

$$= \hbar\omega_r\hat{a}^\dagger\hat{a} + 4E_C\hat{n}^2 - E_J\cos\hat{\phi} + \frac{4E_C}{e}\hat{n}_C\hat{V}, \tag{1.41}$$

where $\hat{V} = \hat{Q}/C_r$ is the voltage across the resonator capacitor. Using equations (1.23) and (1.39) the Hamiltonian becomes

$$\hat{H} = \hbar\omega_r\hat{a}^\dagger\hat{a} + \sqrt{8E_JE_C}\,\hat{b}^\dagger\hat{b} - \frac{E_C}{12}(\hat{b} + \hat{b}^\dagger)^4 + \hbar g(\hat{b}^\dagger - \hat{b})(\hat{a} - \hat{a}^\dagger) \tag{1.42}$$

where the parameter g was introduced. This parameter, known as coupling strength quantifies the strength of the coupling between the qubit and the resonator, it is defined as:

$$g = \frac{2E_C}{\hbar e}\frac{C_{\text{read}}}{C_r}Q_{\text{zpf}}\sqrt{\xi} = \frac{E_C}{\hbar e}\left(\frac{E_J}{2E_C}\right)^{1/4}\frac{C_{\text{read}}}{C_r}\sqrt{2\hbar\omega_rC_r}. \tag{1.43}$$

The coupling strength can be adjusted by varying the capacitance coupling the qubit and the resonator.

When $g \ll \omega_r$ and $g \ll \omega_q$ it is possible to use the rotating wave approximation (RWA) and write the Hamiltonian in the form

$$\hat{H} = \hbar\omega_r\hat{a}^\dagger\hat{a} + \sqrt{8E_JE_C}\hat{b}^\dagger\hat{b} - \frac{E_C}{12}(\hat{b} + \hat{b}^\dagger)^4 + \hbar g(\hat{b}^\dagger\hat{a} + \hat{b}\hat{a}^\dagger). \tag{1.44}$$

Focusing on the first two levels of the transmon we obtain the Jaynes-Cummings Hamiltonian [14] that reads

$$\hat{H} = \hbar\omega_r\hat{a}^\dagger\hat{a} - \frac{\hbar\omega_{01}}{2}\hat{\sigma}_z + \hbar g(\hat{\sigma}_+\hat{a} + \hat{\sigma}_-\hat{a}^\dagger), \tag{1.45}$$

and describes the interaction between an atom, in this case, an artificial atom, with an electromagnetic field in the approximation of the two-level system.

When the coupling strength g is much smaller than the detuning between the qubit and the resonator $\Delta = \omega_q - \omega_r$ the system operates in the dispersive regime. The Jaynes-Cummings Hamiltonian in the dispersive regime, which is the condition in which we operate to perform the qubit readout, can be approximated as

$$\hat{H}_{\text{disp}} = \hbar(\omega_r - \chi\hat{\sigma}_z)\hat{a}^\dagger\hat{a} - \frac{\hbar}{2}(\omega_{01} + \chi)\hat{\sigma}_z, \tag{1.46}$$

where χ is the dispersive shift defined as

$$\chi = \frac{g^2}{\Delta}. \tag{1.47}$$

Equation (1.46) shows that there is a shift in the resonator frequency from ω_r to $\omega_r - \chi$ if the qubit is in the ground state and a shift from ω_r to $\omega_r + \chi$ if the qubit is in the excited state.

The dispersive shift equation (1.47) was derived assuming that the qubit can be approximated as a two-level system. Considering also the higher energy levels of the qubit a more accurate expression of the dispersive shift is given by

$$\chi = \frac{g^2}{\Delta(1+\Delta/\eta)}, \tag{1.48}$$

where η is the qubit anharmonicity.

1.4 Qubit control

Another necessary element for performing quantum computation is the implementation, starting with single-qubit gates. A qubit can be driven into any arbitrary superposition state by applying an electrical microwave pulse with a carefully tuned amplitude, duration, and phase. This pulse is generated by an AC voltage source located outside the dilution refrigerator that hosts the qubit. The drive pulse is brought to the qubit by an on-chip waveguide which is capacitively coupled to the qubit as shown in Figure (1.6b), where the coupling capacitance is indicated with c. This signal path is commonly referred to as a control line or an XY line. The pulse that arrives at the device has the analytical form

$$V_d(t) = A\varepsilon(t)\sin(\omega_d t + \alpha), \tag{1.49}$$

where A is the pulse amplitude in volts, ω_d is the drive frequency in rad/s, α is the phase of the pulse and $\varepsilon(t)$ is the modulation of the pulse; the maximum of $\varepsilon(t)$ is fixed at one. As a first approximation, the envelope of the drive pulse is often chosen to have a Gaussian shape, which is preferred over a square pulse due to its smaller frequency bandwidth, minimizing the excitation of higher energy levels. However, the study of pulse shapes that minimize leakage to states outside the computational basis remains an active area of research [15].

In a similar way to what was previously done to study the capacitive coupling between the qubit and the resonator, it is possible to derive the Hamiltonian of the transmon capacitively coupled to the control line starting from the circuit shown in Figure (1.6b)

$$\hat{H} = 4E_C\left(\hat{n} + \frac{C_d V_d(t)}{2e}\right)^2 - E_J\cos\hat{\phi}, \tag{1.50}$$

where $E_C = e^2/2C_\Sigma$ and $C_\Sigma = C_d + C_q$. By expanding the parenthesis and dropping a constant term the Hamiltonian can be re-written as

$$\hat{H} = 4E_C\hat{n}^2 - E_J\cos\hat{\phi} + 2e\frac{C_d}{C_\Sigma}V_d(t)\hat{n}. \tag{1.51}$$

Since $\hat{n} = -i(\hat{b}-\hat{b}^\dagger)/2\sqrt{\xi}$ (from equation (1.23)), the last term of the Hamiltonian can be rewritten:

$$\begin{aligned}
\hat{H}_d &= -2e\frac{C_d}{C_\Sigma}V_d(t)\hat{n} && (1.52)\\
&= -i\frac{e}{\sqrt{\xi}}\frac{C_d}{C_\Sigma}V_d(t)(\hat{b}-\hat{b}^\dagger) && (1.53)\\
&= -i\frac{e}{\sqrt{\xi}}\frac{C_d}{C_\Sigma}V_d(t)(|0\rangle\langle 1| - |1\rangle\langle 0|) && (1.54)\\
&= \frac{e}{\sqrt{\xi}}\frac{C_d}{C_\Sigma}V_d(t)\hat{\sigma}_y, && (1.55)
\end{aligned}$$

Regarding the first part of equation (1.51), by focusing on the firs two levels of the transmon the qubit Hamiltonian can be rewritten as $\hat{H}_q = -\frac{1}{2}\hbar\omega_q\hat{\sigma}_z$. Then, by substituting the qubit and drive Hamiltonians (equation (1.52)) and the analytical form of $V_d(t)$ in equation (1.51) we obtain

$$\begin{aligned}
\hat{H} = \hat{H}_q + \hat{H}_d &= -\frac{\hbar\omega_q}{2}\hat{\sigma}_z + \frac{e}{\sqrt{\xi}}\frac{C_g}{C_\Sigma}A\varepsilon(t)\sin(\omega_d t+\alpha)\hat{\sigma}_y && (1.56)\\
&= -\frac{\hbar\omega_q}{2}\hat{\sigma}_z + \hbar\Omega\varepsilon(t)\sin(\omega_d t+\alpha)\hat{\sigma}_y, && (1.57)\\
& && (1.58)
\end{aligned}$$

where Ω is the Rabi frequency,

$$\Omega = \frac{e}{\hbar\sqrt{\xi}}\frac{C_d}{C_\Sigma}A. \tag{1.59}$$

The Rabi frequency quantifies the coupling between the control line and the qubit.
To study the dynamics of the system it is convenient to use the rotating frame, the Hamiltonian becomes:

$$\hat{H}' = -\frac{\hbar(\omega_q-\omega_d)}{2}\hat{\sigma}_z + \hbar\Omega\varepsilon(t)\sin(\omega_d t+\alpha)\left(\hat{\sigma}_x\sin\omega_d t + \hat{\sigma}_y\cos\omega_d t\right) \tag{1.60}$$

$$= -\frac{\hbar(\omega_q-\omega_d)}{2}\hat{\sigma}_z + \hbar\Omega\varepsilon(t)\left(\frac{\hat{\sigma}_x}{2}\left(-\cos(2\omega_d t+\alpha)+\cos\alpha\right) + \frac{\hat{\sigma}_y}{2}\left(\sin(2\omega_d t+\alpha)+\sin\alpha\right)\right) \tag{1.61}$$

$$= -\frac{\hbar(\omega_q-\omega_d)}{2}\hat{\sigma}_z + \frac{\hbar\Omega}{2}\varepsilon(t)\left(\hat{\sigma}_x\cos\alpha + \hat{\sigma}_y\sin\alpha\right), \tag{1.62}$$

where in the last step we used the rotating wave approximation, meaning that as $\omega_d \approx \omega_q$ the terms $\cos(2\omega_d t+\alpha)$ and $\sin(2\omega_d t+\alpha)$ oscillate rapidly and their contribution to the dynamics can be neglected.

For example, if $\omega_d = \omega_q$ and $\alpha = \frac{\pi}{2}$, the state evolution is described as follows:

$$|\psi\rangle = \exp\left\{\frac{i}{\hbar}\int_0^{+\infty}\hat{H}(t')\right\}|0\rangle = \exp\left\{-i\frac{\Omega}{2}\int_0^{+\infty}\varepsilon(t')dt'\right\}|0\rangle = e^{-i\frac{\theta}{2}\hat{\sigma}_y}|0\rangle, \tag{1.63}$$

where

$$\theta = \Omega\int_0^{+\infty}\varepsilon(t')dt', \tag{1.64}$$

In general, a microwave pulse given by $V_d = A\varepsilon(t)\sin(\omega_d t+\alpha)$ implements a single-qubit rotation $R_{\hat{n}(\alpha)}(\theta)$ around an axis $\hat{n}$ that lies on the equator of the Bloch sphere, so that

$$R_{\hat{n}(\alpha)}(\theta) = e^{-\frac{i}{2}\hat{n}(\alpha)\cdot\vec{\sigma}\theta} = e^{-\frac{i}{2}(\hat{\sigma}_x\cos\alpha+\hat{\sigma}_y\sin\alpha)\theta}, \tag{1.65}$$

1.5 Qubit state degradation

The ability to apply quantum gates to qubits is essential for executing complex circuits and operations. However, to ensure their effective implementation, it is crucial to study the qubit decoherence time, as it directly impacts the reliability and performance of quantum computations.

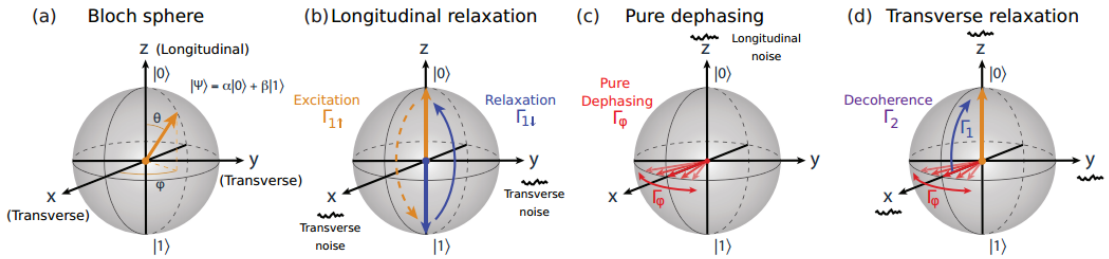


Figure 1.7: Transverse and longitudinal noise represented on the Bloch sphere. (a) Bloch sphere representation of the quantum state $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$, the z-axis is longitudinal in the qubit frame, while the $x-y$ plane is transverse in the qubit frame. (b) Longitudinal relaxation results from energy exchange between the qubit and its environment, due to transverse noise. (c) Pure dephasing in the transverse plane arises from longitudinal noise along the z-axis that fluctuates the qubit frequency. A Bloch vector along the x-axis will diffuse clockwise or counterclockwise around the equator due to the stochastic frequency fluctuations. (d) Transverse relaxation results in a loss of coherence due to a combination of energy relaxation and pure dephasing.
Credits: [16]

1.5.1 Qubit decoherence

By definition, decoherence refers to the loss of coherence in a quantum system, meaning that the relative phase between quantum states becomes randomized due to interactions with the environment. Decoherence is typically characterized by two key time constants: the energy relaxation time T_1 and the dephasing time T_2. The energy relaxation time describes the characteristic time over which a qubit in an excited state decays to its ground state due to interactions with the environment. The decay follows an exponential low:

$$p_e(t) = p_e(0)e^{-\Gamma_1 t}, \tag{1.66}$$

where $p_e(t)$ is the probability of the qubit being in the excited state at time t, $p_e(0)$ the probability of the qubit being in the excited state at time $t = 0$ and $\Gamma_1 = 1/T_1$ is the decay rate of the qubit from state $|0\rangle$ to state $|1\rangle$. Energy relaxation is primarily due to transverse noise, namely noise that couples the qubit through $\hat{\sigma}_x$ and/or $\hat{\sigma}_y$, it is caused by coupling to electromagnetic noise and other dissipative processes that induce energy exchange with the environment.

Longitudinal noise instead is due to environmental fluctuations coupling to $\hat{\sigma}_z$ that randomize the phase of the superposition state without causing energy exchange with the environment. The characteristic dephasing time of a qubit is $T_2^* = 1/\Gamma_\phi$ dove Γ_ϕ is the pure dephasing rate.

The transverse relaxation rate Γ_2 describes the combined effects of pure dephasing and energy relaxation and is defined as

$$\Gamma_2 = \frac{\Gamma_1}{2} + \Gamma_\phi, \tag{1.67}$$

or equivalently

$$\frac{1}{T_2} = \frac{1}{2T_1} + \frac{1}{T_2^*}. \tag{1.68}$$

T_2 is known as decoherence time and quantifies how quickly a quantum superposition state loses phase coherence due to environmental fluctuations. For this reason, in an ideal system where pure dephasing is absent, T_2 is simply limited by energy relaxation via the relaxation

$$T_{2,\max} = 2T_1. \tag{1.69}$$

This result follows from the Bloch equations [16], where both energy relaxation and phase damping contribute to coherence loss. However, in real systems, additional dephasing mechanisms, such as low-frequency noise from charge or flux fluctuations, reduce T_2 below this upper bound and limit it to (1.68).

1.5.2 Noise models

Quantum noise mentioned in the previous subsection can be described using quantum operations known as quantum channels. A quantum operation is a mathematical transformation that describes how a quantum state changes as a consequence of a physical process. Formally, it is a map $\mathcal{E}$ that transforms a quantum state described by a density operator $\hat{\rho}$ into another state described by a new density operator $\hat{\rho}'$:

$$\mathcal{E}(\hat{\rho}) = \hat{\rho}'. \tag{1.70}$$

Amplitude damping channel

The amplitude-damping channel models energy dissipation due to interaction with a zero-temperature environment. The quantum map which describes the process is

$$\mathcal{E}_{\text{ad}}(\hat{\rho}) = \hat{E}_0\hat{\rho}\hat{E}_0^\dagger + \hat{E}_1\hat{\rho}\hat{E}_1^\dagger, \tag{1.71}$$

where the Kraus operators:

$$\hat{E}_0 = \begin{bmatrix} 1 & 0 \\ 0 & \sqrt{1-\gamma} \end{bmatrix}, \quad \hat{E}_1 = \begin{bmatrix} 0 & \sqrt{\gamma} \\ 0 & 0 \end{bmatrix}, \tag{1.72}$$

where $p = 1 - e^{-t/T_1}$ represents the probability of relaxation.

Generalized amplitude damping channel

The generalized amplitude damping channel models energy dissipation in a thermal environment where both relaxation and thermal excitation occur.

$$\mathcal{E}_{\text{gad}}(\hat{\rho}) = p\left(\hat{E}_0\hat{\rho}\hat{E}_0^\dagger + \hat{E}_1\hat{\rho}\hat{E}_1^\dagger\right) + (1-p)\left(\hat{E}_2\hat{\rho}\hat{E}_0^\dagger + \hat{E}_3\hat{\rho}\hat{E}_1^\dagger\right), \tag{1.73}$$

where:

$$\hat{E}_0 = \sqrt{p}\begin{bmatrix}1 & 0\\ 0 & \sqrt{1-\gamma}\end{bmatrix}, \quad = \sqrt{p}\begin{bmatrix}0 & \sqrt{\gamma}\\ 0 & 0\end{bmatrix}, \tag{1.74}$$

$$\hat{E}_2 = \sqrt{1-p}\begin{bmatrix}\sqrt{1-\gamma} & 0\\ 0 & 1\end{bmatrix}, \quad \hat{E}_3 = \sqrt{1-p}\begin{bmatrix}0 & 0\\ \sqrt{\gamma} & 0\end{bmatrix}, \tag{1.75}$$

where γ represents the thermal excitation probability and p describes the system-bath interaction strength.

Phase Damping channel

The phase damping channel describes pure dephasing, where the qubit phase is randomized without energy exchange. The Kraus operators for this channel are:

$$\hat{E}_0 = \begin{bmatrix}1 & 0\\ 0 & \sqrt{1-\lambda}\end{bmatrix}, \quad \hat{E}_1 = \begin{bmatrix}0 & 0\\ 0 & \sqrt{\lambda}\end{bmatrix}, \tag{1.76}$$

where $\lambda = 1 - e^{-t/T_2^*}$ quantifies the probability of phase randomization.

Depolarizing channel

The depolarizing channel captures the combined effects of relaxation (T_1) and pure dephasing (T_2^*), making it a natural model for total decoherence. The transformation applied by this channel is given by:

$$\hat{\rho} \to (1-q)\hat{\rho} + \frac{q}{3}(\hat{\sigma}_x\hat{\rho}\hat{\sigma}_x + \hat{\sigma}_y\hat{\rho}\hat{\sigma}_y + \hat{\sigma}_z\hat{\rho}\hat{\sigma}_z), \tag{1.77}$$

meaning that the depolarizing channel leaves $\hat{\rho}$ unchanged with probability $(1-q)$, while with probability $q/3$ one of the Pauli operators is applied to it. The corresponding Kraus operators are:

$$\hat{E}_0 = \sqrt{1-q}\mathbb{I}, \quad \hat{E}_1 = \sqrt{\frac{q}{3}}\hat{\sigma}_x, \quad \hat{E}_2 = \sqrt{\frac{q}{3}}\hat{\sigma}_y, \quad \hat{E}_3 = \sqrt{\frac{q}{3}}\hat{\sigma}_z. \tag{1.78}$$

This channel models both energy relaxation and phase randomization, effectively representing total decoherence.

An alternative representation of the depolarizing channel describes the process in which the system state ρ is replaced by the maximally mixed state $\frac{\mathbb{I}}{2}$ with probability d:

$$\mathcal{E}_{dc}(\rho) = p\frac{\mathbb{I}}{2} + (1-p)\rho, \tag{1.79}$$

where the following relation holds between p and q, $q = 3p/4$.

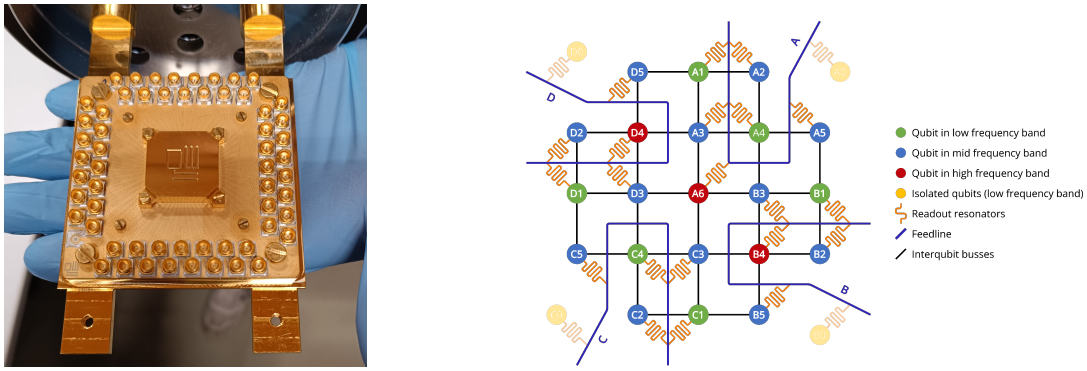
Chapter 2

Qubit calibration

In this chapter we will describe the process of calibration for superconducting flux-tunable transmon on the hardware located in the QRC (Quantum Research Center) Laboratory of the TIwe (Technology & Innovation Institute) in Abu Dhabi.

2.1 Experimental setup

Most of the results presented in this work were obtained using the Contralto-D chip [17], which offers up to 21 fully connected qubits and 4 isolated qubits, for a total of 25 physical qubits. The distinction between fully connected and isolated qubits is important as only the fully connected subset supports native two-qubit gate operations, which are essential for implementing entangling gates and complex quantum circuits. Isolated qubits, while still operational for single-qubit tasks, do not participate in multi-qubit interactions and thus are not functionally equivalent in terms of computational capabilities. The topology of the qubit is shown in Figure (2.1b).



(a) (b)

Figure 2.1: Figure (2.1a): Picture of the Contralto-D chip from QuantWare. Figure (2.1b): Topology of the Contralto-D chip from QuantWare.

As discussed in the previous chapter, the behavior of Josephson junctions and SQUIDs relies critically on the superconducting state of the materials involved. To achieve and maintain this regime, it is essential that the superconducting elements operate well below their critical temperature. For this reason, the Contralto-D chip is installed at the lowest temperature stage of the cryostat, where the required thermal conditions for superconductivity are met. This ensures the proper functioning of the quantum hardware and enables the realization of coherent quantum operations.

These systems achieve ultra-low temperatures by exploiting the unique quantum properties of helium-3 (^{3}He) and helium-4 (^{4}He) isotopes in a dilution process. At the core of a dilution refrigerator is a mixing chamber, where the cooling mechanism takes place. When a mixture of ^{3}He and ^{4}He is cooled below approximately 870 millikelvin, the two isotopes phase-separate into a ^{3}He-rich phase and a ^{3}He-dilute phase. The key principle is that when ^{3}He atoms cross the phase boundary—from the concentrated phase into the dilute phase—they absorb energy from

their surroundings. This process is endothermic and is the fundamental source of cooling in the dilution refrigerator.
The system operates as a closed loop: ^{3}He gas is circulated using a combination of sorption pumps and still pumps, which remove ^{3}He vapor from the still (typically at 600–800 mK), recondense it at a higher stage, and reintroduce it into the mixing chamber. The refrigerator includes several thermalization stages—typically at 50 K, 4 K, 800 mK, 100 mK, and finally below 20 mK—each connected to a corresponding cooling stage and separated by radiation shields and thermal filters to minimize heat load and noise from higher-temperature stages. Dilution refrigerators are highly stable and capable of reaching base temperatures below 10 mK, with hold times on the order of days or even weeks. These temperatures are crucial for achieving the low thermal noise and long coherence times necessary for high-fidelity quantum operations in superconducting circuits. Specifically, the cryostat employed in the lab is the XLD from Bluefors [18], an image of the cryostat is shown in Figure (2.2).

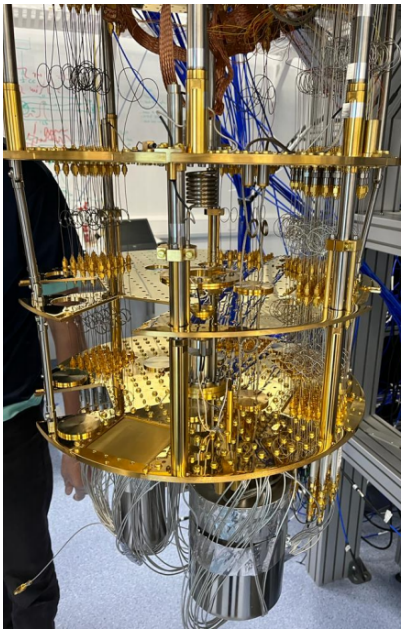


Figure 2.2: Picture of the XLD dilution refrigerator at the QRC Lab

Outside the cryostat, the control and readout of superconducting qubits are managed by dedicated room-temperature electronics. These systems are responsible for generating the microwave pulses used to drive single- and two-qubit gates, as well as for acquiring and processing the output signals that encode the qubit states Typically they include arbitrary waveform generators (AWGs), microwave sources, mixers, digitizers, and field-programmable gate arrays (FPGAs). The generated microwave pulses are shaped and modulated at room temperature before being attenuated and routed to the cryogenic environment. Similarly, signals returning from the qubits are amplified and digitized for state discrimination and further processing. The electronics employed in the lab for the control of the `qw11q` is the OPX1000 platform by Quantum Machines [19].

The software we used for the calibration of the qubits and the subsequent experiments is `Qibocal` ([20], [21], [22]), while the backend for communication with the laboratory instruments is `Qibolab` ([23], [24], [25]). `Qibolab` is the control layer responsible for managing and executing low-level instructions on the hardware, bridging high-level quantum models and physical quantum platforms. It is designed to support diverse experimental setups and allows the researcher to define custom hardware configurations through a platform abstraction and to execute custom pulse sequences using both commercial and open-source firmware. The communication between `Qibolab` and the quantum hardware is structured and modular, relying on a stack that includes instrument drivers, pulse control logic, and a compiler that translates abstract quantum gates into hardware-specific instructions. This structure enables compatibility with heterogeneous platforms and facilitates the development of experimental drivers tailored to different laboratory environments. `Qibocal` interfaces directly with `Qibolab` to apply calibration protocols on the physical device. The routine deployment takes place through the interpretation of declarative runcards

written in YAML. `Qibocal` allows easy execution of pulse sequences, collection of measurement data, and interpretation of the results through the reports that are automatically generated upon completion of the routine.

2.2 Single qubit calibration experiments

The first task that we needed to complete at the beginning of my thesis work was the calibration of at least a line of the superconducting qubits of the Contralto-D chip using the `Qibocal` library. From this point onward, for the sake of brevity, we will refer to the chip interchangeably as Contralto-D or `qw11q`, which is the name of the node under which it is registered on the QRC computing cluster. In the following, we will describe the experiments that we performed and comment on the results.

2.2.1 Resonator calibration

Before starting with the calibration of the gates necessary for quantum computing it is necessary to characterize the qubit and calibrate the readout pulses. For this reason, the calibration process starts with the characterization of the resonator coupled to the qubit that will be used to perform non-destructive measurements of the qubit state.

Resonator spectroscopy

The first step to calibrate the readout pulse is to characterize the resonator is to find the resonator frequency, that is the transition frequency for the resonator. At this frequency, a distinct difference in the transmitted signal can be observed depending on the type of resonator used. In the case of a 3D cavity resonator, the signal appears amplified, whereas for a 2D planar resonator, the signal tends to be more strongly absorbed. Regardless of the resonator type, the response typically exhibits an approximately Lorentzian-shaped peak: this peak is positive for 3D cavities and negative for 2D resonators.

The outcome of this experiment is strongly influenced by the amplitude of the excitation pulse. To reliably determine the resonator frequency, the pulse duration can be fixed on the order of microseconds, which is sufficient to observe the relevant signal response. However, selecting an appropriate amplitude requires more careful consideration. When the amplitude is high, the signal becomes more prominent, improving the signal-to-noise ratio (SNR) and making it easier to identify the resonator's response. If the amplitude is increased too much, however, it can drive the system out of the superconducting regime. In this case, the resonator becomes effectively decoupled from the qubit, and the frequency observed corresponds to the so-called bare resonator frequency. During the first calibration operations, it is not necessary to have the qubit coupled to the resonator but is helpful to improve the SNR to have a more defined peak in the signal, for this reason, the first calibration routines are run in a high-power regime.

In this experiment the main variables on which the experimenter can operate are the frequency range that is willing to scan and the step of the scan. Regarding the frequency range a very wide scan can be useful if nothing is known about the studied resonator, but usually the design parameters are provided by the manufacturer; t is possible that they are not exact, but can give an idea of the region to scan. Regarding the step for the scan, usually a step of 200 MHz can be used to probe the resonator frequency but can be reduced if a more precise scan is needed. An example of measurement of the bare resonator frequency is shown in Figure (2.3).

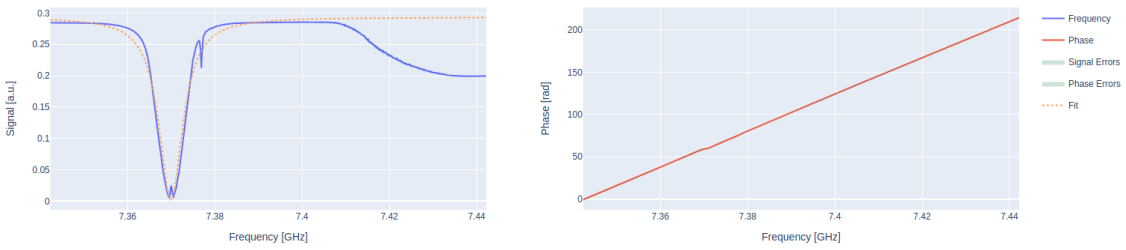


Figure 2.3: Output of resonator spectroscopy with high power on qubit B2.

An example of the measurement of the resonator frequency in the low-power regime is shown in Figure (2.4).

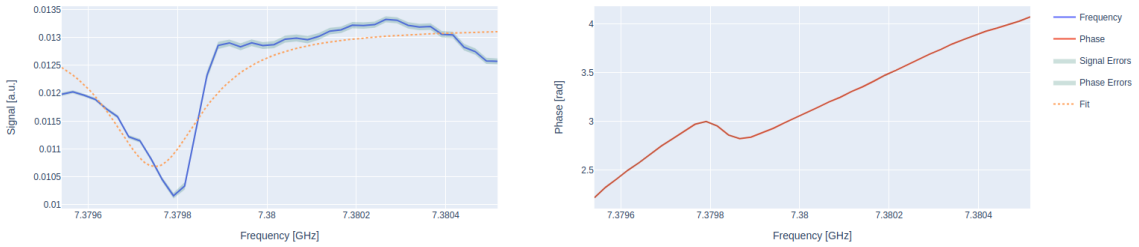


Figure 2.4: Output of resonator spectroscopy with low power on qubit B2.

A more precise study of the resonator's response can be carried out by performing the resonator spectroscopy routine using a different fitting function, the one referred to as `s21` in `Qibocal` . This variation of the experiment is described in Appendix (A).

Resonator punchout

After having found the bare resonator frequency it is necessary to identify the the frequency of the resonator when coupled to the qubit. To do this it is possible to repeat the spectroscopy, this time over a narrower frequency range and for varying pulse amplitudes. The resonator frequency is expected to depend strongly on amplitude: it remains constant in the high-power regime, shifts during an intermediate transition phase, and stabilizes again at a different value once the qubit-resonator interaction becomes significant. For this calibration protocol, the experimenter must carefully choose the frequency and amplitude range to scan to avoid extremely long experiments, at the same time the frequency and amplitude steps must be small enough to clearly identify the frequency and amplitude in which the shift happens.

The expected result from a resonator punchout experiment is shown in Figure (2.5)

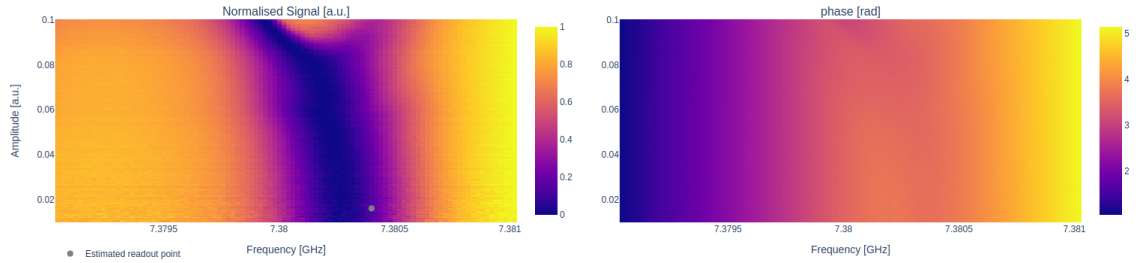


Figure 2.5: Output of resonator punchout on qubit B2.

Resonator flux dependence

As explained in section (1.2.4) it is suggested to work at the qubit sweetspot, where the qubit frequency is less sensitive to magnetic flux fluctuations. For this reason, it is useful to estimate the location of the sweetspot prior to performing qubit spectroscopy, enabling measurements to be carried out near the optimal bias point. This can be accomplished by exploiting the qubit-resonator coupling: since the qubit frequency depends on the magnetic flux, as shown in equation (D.1), and the qubit is coupled to the resonator, the resonator frequency also exhibits a flux dependence. The resonator detuning in the transmon regime ($E_J \gg E_C$) can be computed as

$$f_r(\Phi) = f_r^{\text{bare}} + \frac{g^2 \sqrt[4]{d^2 + (1-d^2)\cos^2\left(\pi \frac{\Phi}{\Phi_0}\right)}}{f_r^{\text{bare}} - f_q(\Phi)}, \tag{2.1}$$

where f_r^{bare} is the bare resonator frequency, g is the coupling between the transmon and the resonator, d is the junctions asymmetry, E_C is the charging energy, E_J the Josephson energy and $\Phi_0 = h/2e$ is the flux quanta.

In `Qibocal` it is possible to perform a resonator flux dependence experiment to measure and fit the curve described by Eq. (2.1). With this routine the experimenter performs a scan of both the external bias and frequency, this experiment provides an approximate indication of the sweetspot. An example of the output of this experiment is shown in Figure (2.6)

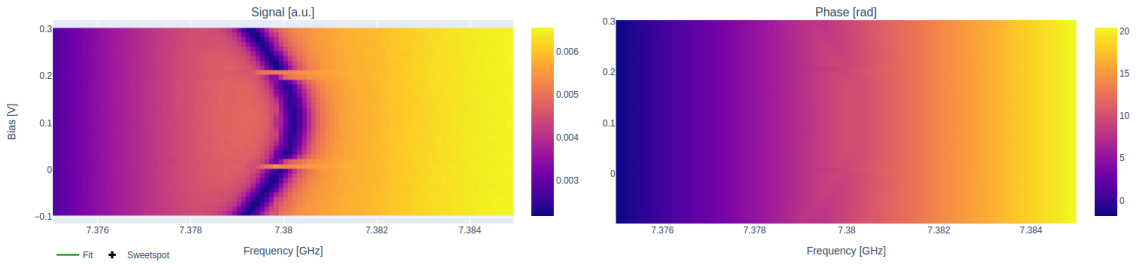


Figure 2.6: Output of resonator flux dependence routine on qubit B2.

2.2.2 Qubit calibration

After having determined all the readout parameters it is possible to continue the calibration process by calibrating the qubit.

Qubit spectroscopy

To determine the resonance frequency of a qubit, a qubit spectroscopy experiment is performed, which, unlike resonator spectroscopy, requires a two-tone approach. While resonator spectroscopy is typically a single-tone measurement used to identify the resonator's response, qubit spectroscopy involves applying a drive tone to the qubit followed by a readout tone to detect the qubit state. This method becomes essential after an initial estimate of the readout frequency and amplitude has been obtained from a resonator punchout experiment. In this protocol, a drive pulse of variable frequency ω is sent through the qubit drive line. If the drive frequency is far detuned from the qubit transition frequency ω_q, it will have no appreciable effect on the qubit state, and the measured signal will remain unchanged. However, as ω approaches ω_{01} the drive pulse can induce transitions between the qubit states. This excitation modifies the qubit population and, consequently, the resonator response, which is sensitive to the qubit state due to their dispersive coupling. When the drive frequency is near resonance and the pulse is sufficiently long, the qubit may reach a maximally mixed state leading to a detectable change in the readout signal amplitude. By sweeping the drive frequency and recording the corresponding readout amplitudes, one can plot a spectroscopy curve that reveals a Lorentzian dip or peak centered at the qubit transition frequency—opposite in direction to the Lorentzian feature observed in the resonator spectroscopy, due to the nature of the state-dependent dispersive shift.

An example of the output of a qubit spectroscopy experiment is shown in Figure (2.7)

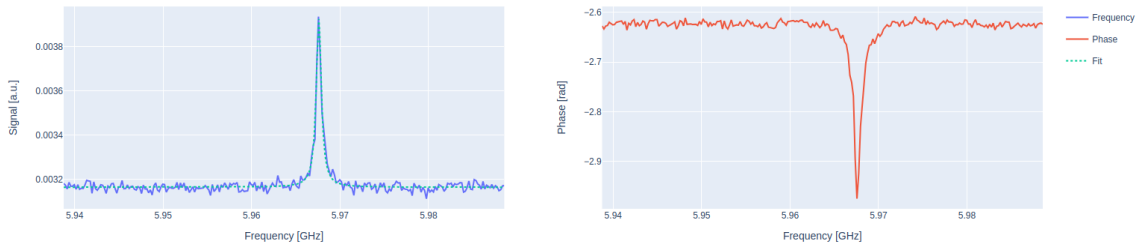


Figure 2.7: Output of qubit spectroscopy on qubit B2.

Qubit spectroscopy for second excited state

Qubit spectroscopy can also be extended to probe transitions to higher excited states beyond the first excited state. Directly observing these higher-level transitions typically requires significantly increased drive power, which may exceed the safe operational limits of the experimental setup. An

alternative and more controlled approach involves first preparing the qubit in state $|1\rangle$, followed by a standard spectroscopy sequence to induce the $|1\rangle \leftrightarrow \langle 2|$ transition.

Note that the description of this calibration routine has been included here for the sake of clarity and continuity of exposition; however, since it requires the qubit to be in the $|1\rangle$ state at the beginning of the spectroscopy, it can only be performed after the execution of a single-shot classification (see Section (2.2.4)).

Qubit flux dependence

By performing a resonator flux dependence the experimenter obtained a first estimate for the sweetspot, which can be improved by performing a qubit flux spectroscopy. In this experiment, a constant DC current signal is sent to the qubit through the flux line with voltage and amplitude fixed for the bias level and then a qubit spectroscopy is performed. This procedure is repeated for different bias levels and different drive frequencies. The expected result of the routine is shown in Figure (2.8).

When calibrating the qubit sweetspot it is important to consider that on a superconducting quantum chip qubits are not completely isolated systems; rather, they interact with one another through both intentional couplings (e.g capacitive or inductive couplers) and unintended cross-talk mechanisms. The mutual interaction between qubits affects their effective operating parameters, particularly their flux-dependent transition frequencies. This means that also the location of the sweetspot can be affected by the biasing of neighboring qubits, this occurs because the flux-tuning circuitry is not perfectly isolated; biasing one qubit can induce a spurious flux in nearby qubits, effectively shifting their frequency spectra and hence their sweetspots. Consequently, calibrating each qubit in isolation may yield misleading results, therefore, to accurately determine and operate all qubits at their true sweetspots under realistic experimental conditions, it is important to perform simultaneous calibration. This ensures that all mutual interactions and cross-couplings are properly accounted for, leading to a more stable and predictable multi-qubit operation.

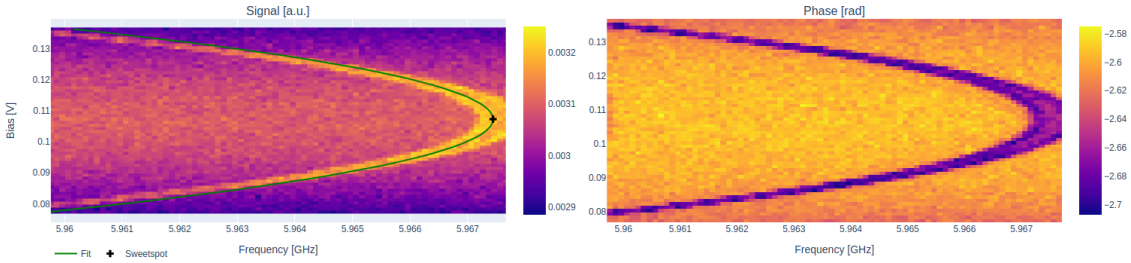


Figure 2.8: Output of the qubit flux dependence on qubit B2.

2.2.3 Drive pulse calibration

After defining all the readout parameters and the parameters for qubit control, it is necessary to calibrate the parameters of the drive pulses to be able to control rotations on the Bloch sphere. With Rabi experiments are possible to calibrate the drive pulses to perform precise rotations (see [26], [27]) within approximately 40 ns. This duration is typical for single-qubit gates in superconducting qubit systems, as it represents a balance between fast control and minimal spectral leakage. Faster gates would require higher drive amplitudes, which can increase errors due to leakage into higher transmon levels or induce unwanted transitions in nearby qubits through cross-resonance or crosstalk effects [10], [28]. On the other hand, significantly longer gates are more susceptible to decoherence, particularly due to T_1 relaxation and T_2 dephasing processes, which limit overall gate fidelity [29].

Usually, the first step of calibrating drive pulses consists of calibrating a π-pulse, namely an X-gate. The goal of the Rabi experiment is to tune the amplitude or duration of the drive pulse, to excite the qubit from the ground state up to state $|1\rangle$.

Rabi oscillation experiments

To experimentally probe the coherent control of a qubit in `Qibocal` it is possible to perform different variations of the Rabi oscillation experiments, which provides direct evidence of the qubit's

ability to undergo controlled rotations under a resonant microwave drive, as described in Section (1.4). In this experiment, a microwave pulse of the form

$$V_d(t) = A\varepsilon(t)\sin(\omega_d t + \alpha), \tag{2.2}$$

is applied through the control line (XY line), and coupled capacitively to the qubit. The qubit is initialized in its ground state $|0\rangle$, and the drive frequency ω_d is set close to the qubit transition frequency ω_q. The envelope $\varepsilon(t)$ is typically Gaussian, and its shape is kept constant throughout the experiment. The key parameter that is varied is the amplitude A of the drive pulse (or alternatively, the duration of the pulse while keeping amplitude constant).

To better understand the evolution of the system, it is possible to consider the driven qubit in the interaction picture. The qubit state can be written as:

$$|\psi(t)\rangle = C_0(t)e^{+i\omega_q t/2}|0\rangle + C_1(t)e^{-i\omega_q t/2}|1\rangle\,. \tag{2.3}$$

Solving the time-dependent Schrödinger equation for state (2.3) under Hamiltonian(1.58) yields the following expressions:

$$C_0(t) = e^{-i\Delta_d t/2}\left[\cos\left(\frac{\Omega_R t}{2}\right) + i\frac{\Delta_d}{\Omega_R}\sin\left(\frac{\Omega_R t}{2}\right)\right], \tag{2.4}$$

$$C_1(t) = i\frac{\Omega}{\Omega_R}e^{i\Delta_d t/2}\sin\left(\frac{\Omega_R t}{2}\right), \tag{2.5}$$

where $\Delta_d = \omega_q - \omega_d$ is the detuning, and

$$\Omega_R = \sqrt{\Omega^2 + \Delta_d^2}, \tag{2.6}$$

is the generalized Rabi frequency.

The probability of finding the qubit in the excited state at time t is given by:

$$P_1(t) = |C_1(t)|^2 = \frac{\Omega^2}{\Omega^2 + \Delta_d^2}\sin^2\left(\frac{\Omega_R t}{2}\right). \tag{2.7}$$

This result shows that the probability oscillates in time with frequency Ω_R, and the amplitude of the oscillation is reduced when the drive is off-resonant ($\Delta_d \neq 0$). The closer the drive is to resonance, the higher the probability of full excitation.

In the resonant case, when $\omega_d = \omega_q$ and $\Delta_d = 0$ the generalized Rabi frequency reduces to $\Omega_R = \Omega$ and the excited state probability becomes

$$P_e(t) = P_1(t) = \sin^2\left(\frac{\Omega t}{2}\right) = \frac{1}{2}(1 - \cos(\Omega t)), \tag{2.8}$$

which is the curve that is used in the fit when performing a Rabi amplitude experiment.

An example of the output of a Rabi amplitude experiment is shown in figure (2.9).

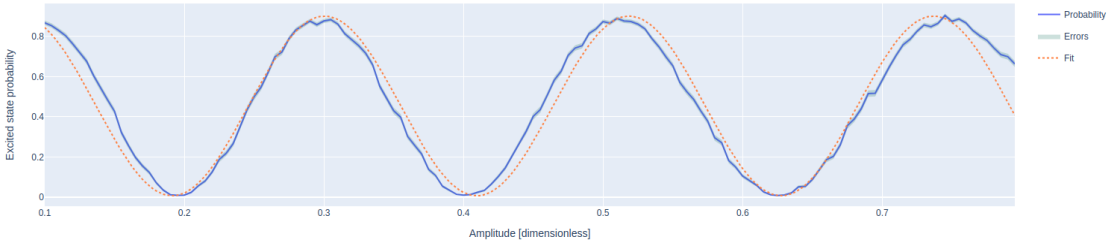


Figure 2.9: Output of the Rabi amplitude routine on qubit B2.

A variation of the Rabi amplitude experiment consists of the Rabi length experiment, where instead of the amplitude, the parameter that is varied is the duration of the drive pulse. The qubit is again initialized in the ground state $|0\rangle$ and subjected to a resonant microwave drive ($\omega_d = \omega_q$). In this case, the pulse envelope $\varepsilon(t)$ and amplitude A are held constant, while the total duration of the pulse is incrementally varied. This enables the observation of coherent oscillations in the

excited state population as a function of pulse length. However, unlike the idealized scenario, in realistic conditions, the qubit experiences energy relaxation and dephasing, which attenuate the oscillation amplitude over time. To account for these effects, the excited state probability is modeled as represents the effective decay time, incorporating both energy relaxation and dephasing processes.

$$P_e(t) = P_1(t) = \frac{1}{2}\left(1 - e^{-t/\tau}\cos\left(\Omega_R \frac{t}{2}\right)\right), \tag{2.9}$$

where Ω_R is the generalized Rabi frequency and τ

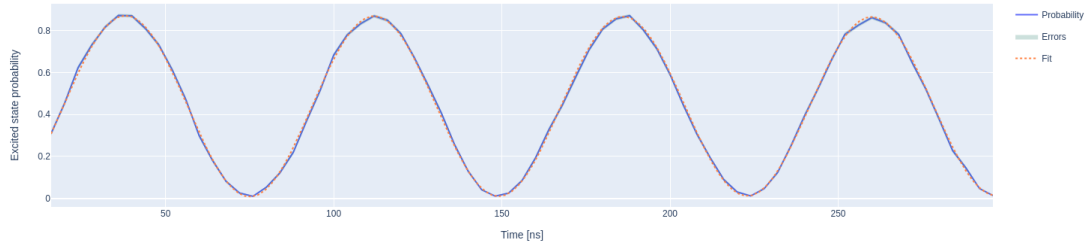


Figure 2.10: Output of the Rabi length routine on qubit B2.

Sometimes, when the qubit frequency is not precisely known, or when systematic effects such as pulse distortions, frequency drifts, or cross-talk may affect the calibration it can be useful to perform a Rabi measurement in which the amplitude and frequency of the drive pulse are scanned. In `Qibocal` this experiment is implemented as the Rabi amplitude-frequency routine, this experiment provides a comprehensive view of the qubit's response to a range of drive conditions and is particularly useful during initial calibration. This experiment reveals a characteristic chevron-shaped interference pattern in the excitation probability as a function of drive frequency and amplitude. Figure (2.11) shows a possible output of this experiment, in the plot in figure in particular it is possible to see only the first region of the chevron shape with high probability of finding the qubit in an excited state.

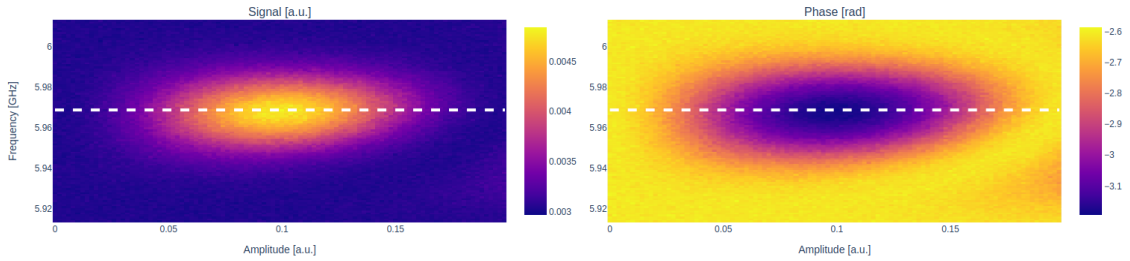


Figure 2.11: Output of the Rabi amplitude-frequency routine on qubit B2.

2.2.4 Single shot classification

At this stage of the calibration procedure, the drive and measurement pulses have been individually calibrated, while a measurement now yields a clear analog output, this alone does not indicate whether the qubit was in state $|0\rangle$ or $|1\rangle$. To establish this correspondence and enable digital state assignment, it is necessary to perform a single-shot readout experiment.

To characterize the readout configuration of a superconducting qubit, the measurement is performed using a heterodyne detection setup (see [30], [16]) which acquires the transmitted microwave signal from the readout resonator. Although the average signal clearly distinguishes the two states, in a single instance, often called a "single shot", the signal is noisy. This noise arises both from quantum fluctuations intrinsic to the electromagnetic field inside the resonator and from added noise contributed by amplifiers and other electronics in the detection chain. The raw signal, sampled over a finite window, is digitally integrated to produce a single complex number corresponding to one measurement instance. Repeating this process after preparing the qubit in a known state produces a cloud of points in the complex IQ plane [16]. For both state $|0\rangle$ and state $|1\rangle$ we typically observe one Gaussian-shaped distribution of IQ values The separation between these two clusters

originates from the dispersive interaction between the qubit and the resonator, which induces a state-dependent frequency shift and consequently a distinguishable phase shift in the transmitted microwave tone. Since the acquisition integrates the signal over time, each shot produces a single point whose location reflects the qubit state but is blurred by the noise that generates the Gaussian distributions.

The readout calibration experiment starts by preparing the qubit in state $|0\rangle$, letting it relax to its ground state, and collecting IQ data without averaging. The experiment is then repeated after applying a π-pulse to prepare the qubit in state $|1\rangle$ and the IQ-values are recorded. Plotting all these single shots in the IQ plane reveals two roughly circular clusters.

To classify the measured IQ points, the centroids of the two Gaussian-like distributions corresponding to the qubit prepared in $|0\rangle$ and $|1\rangle$ are first identified. The axis passing through these centroids is used to define the direction along which the state information is most distinguishable. The IQ plane is then rotated so that this axis aligns with the horizontal (real) axis, concentrating the relevant state-dependent variation along a single dimension. This rotation is characterized by an angle, measured in radians, between the centroid-connecting axis and the original Q-axis.

Following the rotation, the coordinate system is translated such that the centroid of the ground state distribution is placed at the origin. All IQ points are then projected onto the real axis, and the resulting one-dimensional distributions for each state are analyzed. The cumulative distribution functions of these projections are computed, and the optimal threshold is determined by finding the point that maximizes the absolute difference between the two distributions. This threshold defines the decision boundary used to assign a binary qubit state to any new measurement, based on its projection along the rotated axis.

The quality of the classification is then quantified by the assignment fidelity, defined as [30]

$$\mathcal{F} = 1 - \frac{1}{2}(P(0,1) + P(1,0)), \tag{2.10}$$

where $P(i,j)$ is the probability of measuring the qubit in state i but prepared in state j; this fidelity is typically expected to exceed 90–95% for a well-calibrated readout. It serves as a crucial figure of merit for the qubit, reflecting how reliably one can distinguish its quantum states in a single-shot measurement.

The result of the classification routine should be similar to the one shown in Figure (2.12).

Figure 2.12: Output of the single shot classification on qubit B2.

2.2.5 Fine-tuning calibration

Ramsey experiment

After calibrating the readout and π-pulses, technically should be possible to execute algorithmic experiments on the qubit. However, essential characteristics of the qubit, such as its coherence properties and the precise drive frequency, remain to be determined; without this information, circuit fidelities would be suboptimal. The Ramsey experiment is a simple yet powerful protocol that allows simultaneous investigation of multiple aspects of qubit behavior, including the fine-tuning of the drive frequency and verification of coherent control and signal phase consistency. It also enables the extraction of the qubit's coherence time T_2^*.

In its standard implementation, the Ramsey sequence consists of two $\frac{\pi}{2}$-pulses separated by a variable delay τ.
The first $\frac{\pi}{2}$-pulse brings the qubit from the ground state $|0\rangle$ onto the equatorial plane of the Bloch sphere, placing it in a superposition of $|0\rangle$ and $|1\rangle$. During the delay τ, he qubit state accumulates a relative phase due to its Larmor precession and environmental noise, effectively evolving around the z-axis.
The second $\frac{\pi}{2}$-pulse projects the final state back onto the measurement basis, and the probability of measuring the excited state $|1\rangle$ is recorded. Repeating this procedure for different delay times τ allows us to monitor the decay of coherence in the qubit state.

In the ideal, non-detuned case, where the drive frequency matches the qubit transition frequency, the observed signal exhibits a purely exponential decay toward a baseline value, from which the characteristic decoherence time T_2^* can be extracted. However, small deviations from the ideal $\frac{\pi}{2}$-pulse, due to imperfect calibration or a mismatch between the drive and qubit frequency lead to more complex behavior.

Usually, a small intentional detuning is applied, in this case, the measurement outcomes show a sinusoidal modulation superimposed on the exponential decay so that the resulting curve can be fitted with the equation [31]

$$p_e(\tau) = \frac{1}{2} + \frac{1}{2} e^{-\tau/T_2^*} \cos(\Delta\omega\tau), \tag{2.11}$$

An example of the output of a Ramsey experiment is shown in Figure (2.13)

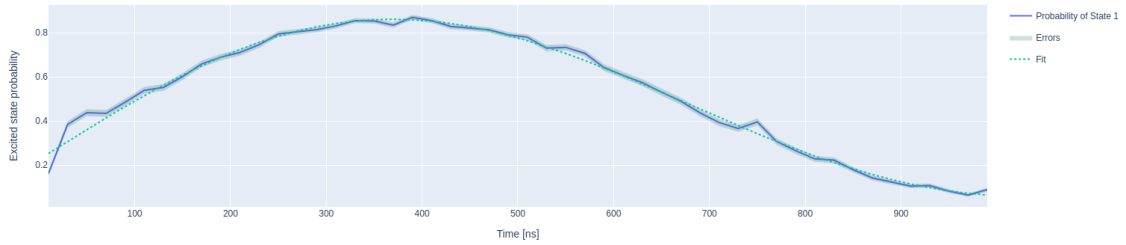


Figure 2.13: Output of the Ramsey amplitude routine on qubit B2.

Flipping experiment

While the Ramsey experiment is typically used to fine-tune the drive frequency of a qubit, the Flipping experiment serves as an effective routine to calibrate the amplitude of the π-pulse, ensuring accurate implementation of $R_x(\pi)$ rotations. This calibration is fundamental for achieving high-fidelity gate operations, and the flipping experiment can often reveal discrepancies in the amplitude that are not evident in Rabi oscillation measurements.
In this protocol, a flip is defined as a pair of consecutive π-pulses, which ideally return the qubit to its initial state due to a full 2π rotation. The experiment begins with the qubit initialized in the ground state $|0\rangle$, followed by an $R_x(\pi/2)$ rotation that places it in an equal superposition state on the equator of the Bloch sphere. Without the initial $R_x(\pi/2)$ rotation, errors in the π-pulses would lead only to a global phase difference, which is not detectable via projective measurements. With the superposition state, however, the qubit's evolution becomes sensitive to amplitude miscalibrations, allowing over- and under-rotations to be distinguished through changes in measurement probability.

Following the initial $\pi/2$-pulse, the qubit is subjected to a number N of flips chosen by the experimenter. After the flips, a projective measurement is performed. This procedure is repeated for increasing values of N, effectively building a scan of how the qubit state evolves under repeated application of imperfect π-pulses.

In the ideal case, where the π-pulse amplitude is perfectly calibrated, the repeated flips return the qubit to the equatorial plane after each cycle, resulting in a constant measurement probability. The recorded signal should appear as a flat line; however, if the amplitude is miscalibrated, the qubit state will drift on the Bloch sphere, leading to oscillatory behavior in the measured population as a function of the number of flips. This deviation is modeled by a sinusoidal function, from which the amplitude error can be quantitatively extracted. Figure (2.14) shows the result of a flipping routine.

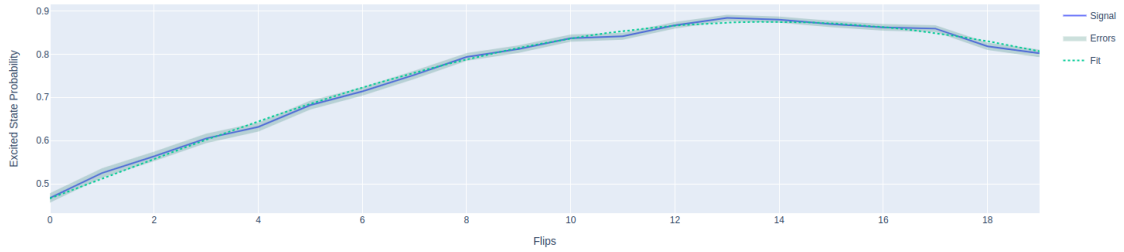


Figure 2.14: Output of the flipping routine on qubit B2.

Dispersive shift

A possible strategy to improve the assignment fidelity is to perform a dispersive shift measurement which helps calibrate the readout frequency. To calibrate the readout frequency based on the dispersive shift, two resonator spectroscopy measurements are performed. In the first, the resonator response is measured while the qubit remains in its ground state $|0\rangle$, effectively treating the system as a bare resonator. In the second measurement, a calibrated π-pulse is applied before each acquisition to prepare the qubit in the excited state $|1\rangle$, and the spectroscopy is repeated under otherwise identical conditions. The transmission or reflection spectra from both configurations are then plotted together as a function of probe frequency. At each frequency point, single-shot measurements are collected, and the resulting IQ data are analyzed. Specifically, the distance between the centroids of the two resulting IQ distributions, one corresponding to the ground state and the other to the excited state, is computed. The readout frequency that maximizes this centroid separation is selected, as it provides the greatest distinguishability between the two qubit states and thus optimizes the readout contrast.

A typical result for this routine is shown in Figure (2.15)

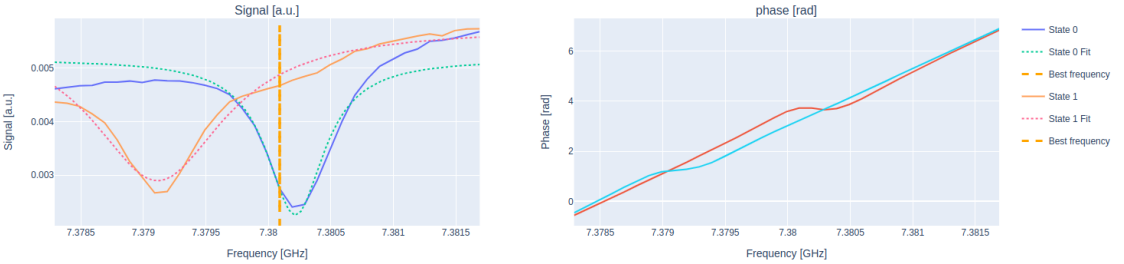


Figure 2.15: Output of the Rabi amplitude routine on qubit B2.

2.2.6 Qubit characterization

T1 & T2 measurement

After having calibrated all the parameters concerning the qubit it is possible to proceed with characterization experiments such as $T1$ and $T2$ measurements.

As explained in Section (1.5.1), due to coupling to the environment the qubit in state $|1\rangle$ will decay to the state $|0\rangle$. To measure the characteristic decay time T_1 it is possible to perform a simple experiment where the qubit is initialized in state $|1\rangle$ using a previously calibrated π pulse. After a variable delay $\Delta\tau$, during which the qubit undergoes spontaneous decay due to coupling to the environment, a projective measurement is performed. For $\Delta\tau = 0$ we expect to fin the qubit in state $|1\rangle$ while for $\Delta\tau \to \infty$ the system relaxes fully to the ground state $|0\rangle$. r intermediate times, the probability of finding the qubit in $|1\rangle$ decays exponentially, and the measured excited state population $p_e(t)$ as a function of time t is fitted to the model

$$p_e(t) = A + Be^{-\frac{t}{T_1}}, \tag{2.12}$$

where T_1 is the energy relation time and A, B are the fitting parameters. The expected result from this routine is shown in Figure (2.16).

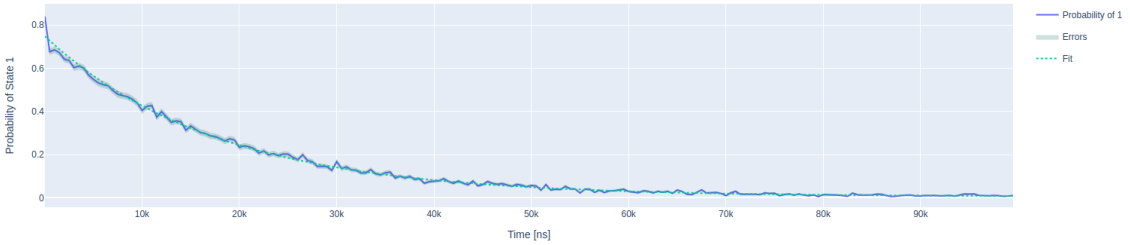


Figure 2.16: Output of the $T1$ measurement on qubit B2.

After measuring T_1 it is necessary to measure also the dephasing time of the qubit, T_2; to do this in `Qibocal` , it is possible to perform a T2 experiment. The acquisition sequence of the experiment is the same as the Ramsey experiment described in Section (2.2.5): a pair of $\pi/2$-pulses is applied with a variable delay τ in between, during which the qubit freely evolves under a slightly detuned drive. As a result, the measured excited state population exhibits oscillations in time, caused by the accumulation of phase due to the detuning. These oscillations are enveloped by an exponential decay that reflects the qubit's loss of phase coherence over time.

In the T2 experiment the same pulse sequence is used with the caveat that the drive pulse is not detuned, the protocol assumes that any error on the drive frequency has already been corrected through a Ramsey experiment. For this reason, the curve that describes the excited state population is expected to follow a simple exponential decay:

$$p_e(t) = A + Be^{-\frac{t}{T_2}}, \tag{2.13}$$

where A and B are two fitting parameters.

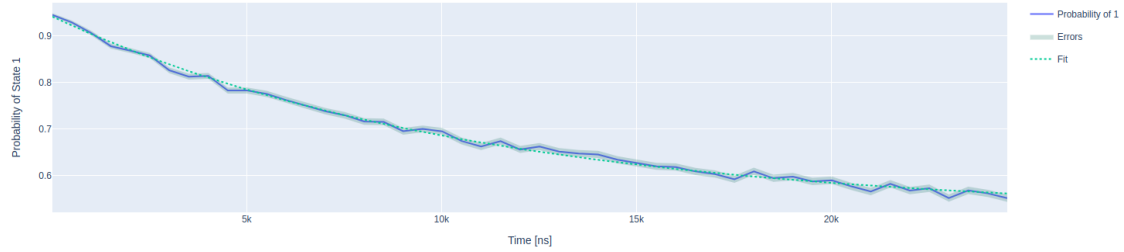


Figure 2.17: Output of the $T2$ measurement on qubit B2.

2.2.7 Standard Randomized Benchmarking

The standard randomized benchmarking experiment will be described in more detail in the following chapter (see Section (3.1)). For now, it is enough to say that in this routine, sequences of randomly selected Clifford gates are applied to the qubit, followed by an inverting gate that ideally returns the system to its initial state. By measuring the survival probability as a function of sequence length and fitting the decay curve, an estimate of the average gate fidelity is obtained. An example of the output expected from the randomized benchmarking protocol is shown in Figure (2.18).

2.2.8 DRAG experiment

DRAG pulse shape

As mentioned in section (1.4), the study of optimal pulse shapes (especially for drive and control pulses) is an active area of research. The Derivative Reduction by Adiabatic Gate (DRAG) technique [28][32] is a widely used method to mitigate leakage errors during high-fidelity single-qubit gate operations. The DRAG is the analytical solution to force the interaction Hamiltonian of the system (1.58) to be restricted to the computational space.

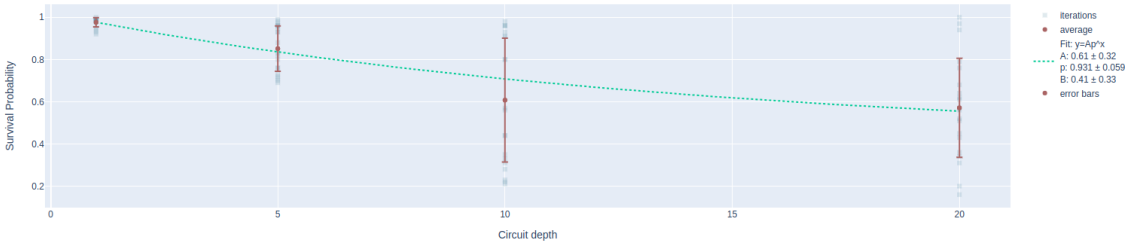


Figure 2.18: Output of the standard randomized benchmarking routine on qubit B2.

The DRAG scheme implements a single-qubit rotation by applying a shaped pulse with an envelope $\Omega(t)$ on one quadrature (typically along $\hat{\sigma_x}$) and a secondary pulse with an envelope proportional to the time derivative $\dot{\Omega}(t)$ on the orthogonal quadrature (typically $\hat{\sigma_y}$). For instance, a rotation about the x-axis is generated by the time-dependent Hamiltonian[8]:

$$\hat{H}_d(t) = \hbar\,\Omega_x(t)\sin(\omega_d t)\,\hat{\sigma}_x + \hbar\,\Omega_y(t)\sin(\omega_d t + \pi)\,\hat{\sigma}_y, \tag{2.14}$$

where the pulse shapes are defined as:

$$\Omega_x(t) = \Omega_0 e^{-t^2/(2\sigma^2)}, \tag{2.15}$$

$$\Omega_y(t) = \lambda/\eta \frac{d}{dt}\Omega_x(t), \tag{2.16}$$

where η represents the qubit anharmonicity and λ is a dimensionless scaling parameter. The efficacy of the DRAG correction arises from its ability to suppress virtual transitions that occur due to the spectral overlap of the primary pulse with off-resonant transitions in the multilevel qubit structure.

Theory predicts that setting $\lambda = 1$ minimizes the leakage to higher excited stats, while $\lambda = 0.5$ is optimal for correcting phase distortions induced during the gates [28], [32], [33]. In experimental implementations, the optimal value of λ may deviate from theoretical predictions due to pulse distortions, the limited bandwidth of the control electronics, and interactions with the readout resonator [34].

DRAG protocol

A straightforward experiment implemented in `Qibocal` to calibrate the λ parameter involves performing two separate measurements using DRAG pulses in the pulse sequence. In the first experiment, a sequence of DRAG pulses is applied in the order $Y_\pi X_{\frac{\pi}{2}}$ with both pulses parametrized by a given value of λ. For the second measurement, the procedure is the same but the pulse order is reversed $X_\pi Y_{\frac{\pi}{2}}$. These specific sequences are chosen because they ideally result in the same final quantum state—any discrepancy between the two indicates phase misalignment, in particular, these sequences exhibit opposite signs of phase errors, as explained in [35].

The post-processing consists of measuring the probability of the qubit being in the state $|1\rangle$ for each value of λ. A linear fit is performed for both pulse sequences, and the correct λ is the value where the two lines cross. This intersection corresponds to the λ value that minimizes phase errors and ensures the qubit operates with minimal distortion.

An example of the expected output for the DRAG routine is shown in Figure (2.19)

In `Qibocal` is implemented also another routine that can be used to determine the correct value of the DRAG parameter β. This method employs a pulse sequence of the form

$$[X_\pi - X_{-\pi}]^N, \tag{2.17}$$

where the repeated application of positive and negative π-rotations amplifies coherent errors arising from imperfect pulse shaping. This approach is particularly sensitive to distortions introduced by an incorrectly tuned quadrature derivative component, which the DRAG technique is specifically designed to compensate for. By varying β, which scales the amplitude of the derivative pulse on the quadrature channel, one observes oscillations in the qubit's return probability to the ground state $|0\rangle$, caused by constructive and destructive interference of the accumulated errors. The measured ground state population is recorded as a function of β and fit to a cosine curve. The optimal value

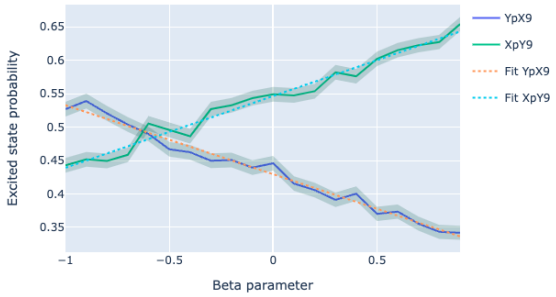


Figure 2.19: Expected output of the `drag_simple` routine. β is the symbol used in `Qibocal` for the λ parameter.

of β corresponds to the point where this oscillation is maximized—indicating minimal leakage and optimal cancellation of phase errors. This approach to the calibration of the β parameter was first introduced in [36]. A possible output of the routine is shown in Figure (2.20).

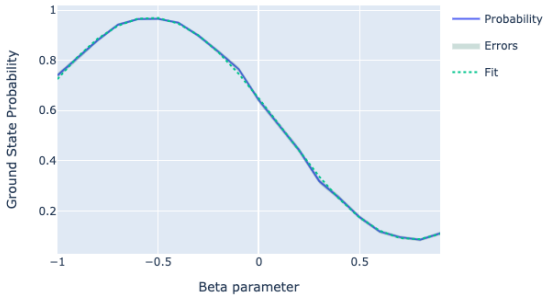


Figure 2.20: Expected output of the `drag_tuning` routine.

2.3 Calibration results

In Table (2.1), are summarized the results obtained for the calibration of line D of the `qw11q` chip. The readout and assignment fidelities were obtained by performing a classification exèperiment on each qubit with 5000 shots, while $T1$ and $T2$ measurements were carried out using the respective protocols, each with 2048 shots.
The standard randomized benchmarking was performed at four different circuit depths: $n = 1, 5, 10, 20$; for each depth the circuit was repeated 20 times with 100 shots per repetition. The number of shots in the randomized benchmarking experiment was deliberately limited, as this routine is inherently very time-consuming.

Qubit	**Readout Fidelity**	**Assignment Fidelity**	**T1** [μs]	**T2** [μs]	**Gate infidelity** ($\cdot 10^{-3}$)
D1	0.876	0.938	26.4 ± 0.4	13.0 ± 2.0	27.0 ± 21.0
D2	0.945	0.973	14.9 ± 0.1	18.0 ± 10.0	20.5 ± 6.2
D3	0.905	0.952	23.6 ± 0.3	28.0 ± 12.0	7.0 ± 18
D4	0.929	0.964	20.6 ± 0.2	38.0 ± 5.2	4.4 ± 4.8
B2	0.902	0.951	17.5 ± 0.2	27.5 ± 5.2	18.0 ± 16

Table 2.1: Single qubit gates calibration results. D line was the first one I tried to calibrate.
For completeness in the table, I report also the values for B2 which were calibrated lately for the purpose of showing calibration results in this chapter.

The values of the assignment fidelity and readout fidelity are overall satisfactory, indicating reliable state discrimination during measurement. Similarly, the relaxation time T_1 of the qubit shows acceptable values.

On the other hand, the coherence time T_2 is generally low, especially when considering that its theoretical upper bound is $2T_1$. This indicates the presence of significant pure dephasing mechanisms, which are limiting the coherence independently of energy relaxation processes. A reduced T_2 can have important consequences for subsequent experiments, particularly impacting the performance of randomized benchmarking protocols. Specifically, a short coherence time may degrade the fidelity of gate operations, leading to an overestimation of the average error per gate.

Undoubtedly, individuals with greater experience in the calibration of superconducting qubits are able to achieve significantly better figures of merit than those reported in table (2.1). For comparison, the qubits on the chip used in this work are designed to achieve coherence times in the range of $T_1 = 29 - 39\mu$s and $T_2 = 58 - 78\mu$s, which are consistent with typical targets for high-performance superconducting qubits. In terms of fidelity, state-of-the-art calibration techniques currently achieve readout fidelities between 95% and 99.5% [16], and single-qubit gate fidelities in the range of 99.5% to 99.99%. The satisfactory, yet overall limited, results obtained from this initial calibration can be partially attributed to a non-optimal calibration procedure and partially to limitations of the experimental setup, which could be further investigated.

Chapter 3

RB fidelity optimization

The calibration procedure described in Section (2.2) is typically time-consuming and demands significant experimental expertise, particularly when aiming for state-of-the-art performance. In the work presented in the previous chapter, the calibration process began from a pre-existing, even if suboptimal, configuration. The goal was to enhance the quality of single-qubit gates using the available `Qibocal` protocols.

Even under these overall favorable initial conditions, the effort required to refine gate performance showed the limitations of manual calibration approaches. As quantum processors scale in both qubit count and architectural complexity, calibration becomes increasingly difficult. Frequent recalibrations are necessary to mitigate the effects of parameter drift and environmental fluctuations, especially in superconducting qubit platforms [16].

Furthermore, as gate fidelities approach the fault-tolerance threshold, accurate control of gates becomes critical for most practical quantum computing applications. In this context, it is useful to introduce, in addition to native single-qubit gates, such as $R_X(\pi)$ pulses or virtual R_Z rotations, the Clifford gates. While native gates are implemented with a single pulse and can be directly calibrated, Clifford gates are typically composed of multiple native gates.

Clifford gates are particularly important in quantum computing because they form a finite subgroup of unitary operations that map Pauli operators onto themselves under conjugation. This property makes them a useful tool to characterize gate performance, their structured algebra and invariance properties make them particularly suitable for randomized benchmarking protocols [37]. In such protocols, Clifford gates enable the depolarization of coherent errors and allow for SPAM-robust estimation of average gate fidelity. Standard randomized benchmarking (RB) protocols, as used in this chapter, measure the average fidelity of a single-qubit Clifford gate. While this quantity does not directly reflect the fidelity of individual native gates, it remains a meaningful and widely used figure of merit.

In this chapter, we present an initial attempt to address both the challenges of calibration complexity and residual gate errors. The goal is to test calibration procedures that are not only effective but also repeatable and accessible to non-expert users. Building on the approach proposed in [38], which demonstrated that optimizing the sequence fidelity at a fixed length of RB can improve gate performance, we explored an automatic calibration improvement procedure based on average Clifford gate fidelity obtained from standard RB evaluation as an optimization target. In our implementation, the optimization is performed with respect to the $R_X(\pi)$ pulse which is a native gate in `Qibolab` (see Section (4.1.1)). The optimization is performed by adjusting its amplitude, frequency, and possibly, the multiplicative factor of the second quadrature component of the DRAG, pulse parameters.

The focus of this first study is on improving the average Clifford gate fidelity for single qubits. This choice was made considering the important role of these operations in quantum circuits and the relative simplicity of their control compared to multi-qubit gates. Optimizing Clifford gate fidelity provides a practical testbed for validating the effectiveness of closed-loop optimization strategies before generalizing them to more complex gates. The results presented in this chapter illustrate the performance of different optimization strategies applied to gate fidelity enhancement under realistic experimental conditions.

3.1 Randomized Benchmarking

A strong limitation to the application of quantum computing technologies in different fields is the accumulation of errors due to the loss of coherence as more quantum gates are applied in sequence. One approach to characterize gate performance is quantum process tomography, which provides a full description of the quantum process under investigation. However, this method becomes impractical for systems with more than a few qubits, as its time complexity scales exponentially with the system size [39], and the results are highly sensitive to state preparation and measurement (SPAM) errors.

To overcome these limitations, RB was developed and is now widely used to estimate the average error rate for a set of quantum gates. The main idea is that the cumulative error resulting from the combined action of random unitary gate sequences, which are drawn uniformly from a group of unitaries according to the Haar measure [40], behaves like a depolarizing channel [41]. This effectively removes the dependence on the specific structure of the noise and leads to a simple exponential decay model from which the average fidelity can be extracted.

Later improvements showed that the procedure could be further simplified by restricting the unitaries to the Clifford group and removing the requirement for sequences to be strictly self-inverting [37]. In the standard RB protocol implementation in `Qibocal` , sequences of random Clifford gates $C_1, C_2, ..., C_m$ are followed by a final inversion gate C_{m+1} which ideally returns the system to its initial state. In real devices, the measured survival probability provides an estimate of the average fidelity over the set of applied Clifford gates.

The standard RB procedure consists of the following steps:

1. Initialize the system in the ground state $|0\rangle$
2. For each sequence length m, build a sequence of m random Clifford gates $C_1, C_2, ..., C_m$
3. Determine the inverse gate $C_{m+1} = (C_m \circ ... \circ C_1)^{-1}$
4. Measure $C_{m+1} \circ C_m \circ ... \circ C_1 |0\rangle$

This process is repeated for different sequence lengths and random sequences.

In an ideal noiseless system, one would obtain:

$$C_{m+1} \circ C_m \circ ... \circ C_1 |0\rangle = (C_m \circ ... \circ C_1)^{-1} \circ (C_m \circ ... \circ C_1) |0\rangle = |0\rangle . \tag{3.1}$$

However, in real systems Equation (3.1) does not hold; instead randomization with Clifford gates behave as a depolarizing channel (1.79) with depolarization probability d.
The observed survival probability as a function of sequence length m follows the exponential decay model

$$F(m) = Ap^m + B, \tag{3.2}$$

where $1-p$ quantifies the depolarization rate, and A, B captures SPAM contributions but not the initial state preparation error.
The depolarization probability d is related to the average Clifford gate fidelity F for a system with n qubits through

$$F = 1 - \frac{d}{2^n - 1}, \tag{3.3}$$

from which the average error per Clifford gate $\varepsilon_{Clifford}$ is obtained as

$$\varepsilon_{Clifford} = 1 - F, \tag{3.4}$$

leading to the final expression

$$\varepsilon_{Clifford} = \frac{d}{2^n - 1} = \frac{1-p}{1-2^{-n}}. \tag{3.5}$$

This shows how the average error per Clifford gate is directly determined by the exponential decay rate p extracted from the RB protocol.

3.2 RB evaluation and optimization

3.2.1 RB evaluation parameters

In the results presented in the following, we employed the RB routine implemented in `Qibocal` , an example of which is shown in Section (2.2.7). This routine was used consistently across all tests, with the following set of parameters: we used 1000 unique random Clifford sequences (`num_of_sequences = 1000`), ensuring robust statistical reliability across different realizations of gate noise. Each sequence was evaluated at increasing circuit depths up to a maximum of 1000 Clifford gates (`max_circuit_depth = 1000`), with the depths spaced by increments of 10 (`delta_clifford = 10`), resulting in benchmarking points at depths of $1, 10, 20, \ldots, 1000$. For each depth and random sequence, we generated only one instance (`n_avg = 1`), meaning that each sequence was distinct and not repeated with the same gate pattern. Consequently, statistical averaging was achieved across the ensemble of random sequences at each depth rather than by repeating individual circuits. [1] All the results presented in the following were obtained running the protocols on qubit `D1`.

3.2.2 Parameters for the optimization

The optimization procedures described in this work aim to enhance the average fidelity of a Clifford gate by minimizing its infidelity, as determined through RB experiments. The optimization is carried out over three control parameters: the amplitude and frequency of the microwave pulse for the $R_X(\pi)$ gate implementation and the DRAG correction parameter β, which modulates the second quadrature of the pulse shape. This closed-loop optimization approach follows a structure similar to the Optimal Randomized Benchmarking for Immediate Tune-up (ORBIT) protocol introduced in [38], wherein the gate performance is iteratively improved based on benchmarking results.

Prior to the application of the optimization algorithm, a fine-tuning sequence is executed to obtain reliable starting values for each parameter. This preliminary step involves a sequence of calibration routines implemented within the `Qibocal` framework. The qubit frequency is first refined using a Ramsey experiment (see Section (2.2.5)), subsequently a flipping sequence (see Section (2.2.5)) is used to adjust the microwave pulse amplitude to ensure that the applied drive results in a precise π-rotation. Finally, the optimal value of the DRAG parameter β is estimated through a dedicated DRAG calibration routine (see Section (2.2.8)). After each of these routines, the platform configuration is updated accordingly in the `platform.json` file, which stores the relevant parameters for the experimental setup [2].

These calibrated values are then used to initialize the optimization algorithm, as illustrated in the code reported in Listing (3.1).

Listing 3.1: Code to set up the RX gate optimization experiment.

```
with Executor.open(
    "myexec",
    path=executor_path,
    platform=platform,
    targets=[target],
    update=True,
    force=True,
) as e:

    e.platform.settings.nshots = 2000
    drag_output = e.drag_tuning(beta_start=-4, beta_end=4, beta_step=0.5)
    ramsey_output = e.ramsey(
        delay_between_pulses_end=1000,
```

[1] All experiments and results presented in this chapter were performed using version 0.1 of `Qibocal` . This means that the parameters used to execute the `rb_ondevice` routine, as described above, refer specifically to version 0.1. At the time of writing, version 0.2 of `Qibocal` has been released and is actively maintained. As a result, the available parameters for running the RB routine may have changed and might no longer correspond exactly to those described in this text.

[2] In `Qibolab` the `qibolab.Platform` object holds all the information required to execute programs in a real QPU. It is comprised of different objects that contain information about the native gates and the lab's instrumentation. The QPU parameters can be saved in (and loaded from) the `parameters.json` file.

```
            delay_between_pulses_start=10,
            delay_between_pulses_step=10,
            detuning=3000000,
            relaxation_time=200000,
        )
        flipping_output = e.flipping(
            nflips_max=50,
            delta_amplitude=0.0005,
            nflips_step=1,
        )

        beta_best = drag_output.results.betas[target]
        ampl_RX = e.platform.qubits[target].native_gates.RX.amplitude
        freq_RX = e.platform.qubits[target].native_gates.RX.frequency

        init_guess = np.array([ampl_RX, freq_RX, beta_best])

        lower_bounds = np.array([-0.5, freq_RX - 4e6, beta_best - 0.25])
        upper_bounds = np.array([0.5, freq_RX + 4e6, beta_best + 0.25])
        bounds = Bounds(lower_bounds, upper_bounds)

        opt_results, optimization_history = rb_optimization(
            e, target, method, init_guess, bounds
        )

    report(e.path, e.history)
```

In addition to defining the initial parameter values, explicit boundaries were also set for the optimization. This choice was motivated by the need to constrain the search space to physically meaningful regions, avoid unstable pulse configurations, and ensure compatibility with the dynamic range and resolution of the control electronics. For the amplitude, the range was chosen to match the full extent accessible by the hardware. The frequency parameter, expressed in Hz, was allowed to vary within a window of 4 MHz centered on the optimal resonance frequency obtained from the Ramsey experiment, ensuring that only physically meaningful detunings are explored. For the DRAG parameter β, the search range was set to 0.25 around the value returned by the initial DRAG calibration; this interval corresponds to the resolution used in the scan performed during the `drag_tuning` routine, thus preserving consistency with earlier characterization steps.

In a subsequent phase of this study, a comparative analysis was performed by repeating the optimization while keeping the pulse shape fixed to a Gaussian envelope, effectively removing β as an optimization variable.

3.3 SciPy optimization methods

3.3.1 Nelder-Mead

The initial optimization attempts were carried out using standard algorithms available in the `SciPy` library [42]. As a first approach, a gradient-free optimization method was selected to reduce sensitivity to measurement noise. Specifically, we employed the Nelder-Mead algorithm, whose application to the RB fidelity has already been reported in the literature [38].

Algorithm information

The Nelder-Mead optimization method, originally introduced by Nelder and Mead in 1965 [43], is a widely used numerical optimization technique for unconstrained problems in multidimensional spaces.
This derivative-free method is operates using simplex, which is a polytope of $n+1$ vertices in a n-dimensional space. The algorithm iteratively updates the simplex by replacing its worst-performing vertex with a new candidate point, thereby guiding the search towards an optimal solution. If the goal is to minimize a given function $f(\mathbf{x})$ where $\mathbf{x} \in \mathbb{R}^n$ the algorithms proceeds with the following steps:

1. If not otherwise initialized, $n+1$ points are sampled for building the initial simplex
2. Order the test points according to their values at vertices: $f(\mathbf{x}_1) \leq f(\mathbf{x}_2) \leq \cdots \leq f(\mathbf{x}_{n+1})$ and check whether the algorithm should terminate.
3. Calculate $\mathbf{x}_0$, the centroid of all points except $\mathbf{x}_{n+1}$.
4. Reflection: Compute the reflected point $\mathbf{x}_r = \mathbf{x}_0 + \alpha(\mathbf{x}_0 - \mathbf{x}_{n+1})$ with $\alpha > 0$. If $\mathbf{x}_r$ satisfies $f(\mathbf{x}_1) \leq f(\mathbf{x}_r) < f(\mathbf{x}_n)$, then a new simplex is obtained by replacing the worst-performing point $\mathbf{x}_{n+1}$ with $\mathbf{x}_r$ and then go to step 1.
5. Expansion: If $\mathbf{x_r}$ is the current best point, meaning that $f(\mathbf{x}_r) < f(\mathbf{x}_1)$, then the expanded point is computed: $\mathbf{x}_e = \mathbf{x}_0 + \gamma(\mathbf{x}_r - \mathbf{x}_0)$ with $\gamma > 1$. If $\mathbf{x}_e$ satisfies $f(\mathbf{x}_e) < f(\mathbf{x}_r)$, then a new simplex is obtained by replacing $\mathbf{x}_{n+1}$ with the expanded point $\mathbf{x}_e$ and then go to step 1.
   If instead $f(\mathbf{x}_e) \geq f(\mathbf{x}_r)$, the new simplex is obtained by replacing $\mathbf{x}_{n+1}$ with $\mathbf{x}_r$, and then go to step 1.
6. Contraction: In this case is certain that $f(\mathbf{x}_r) \geq f(\mathbf{x}_n)$ then:
   - If $f(\mathbf{x}_r) < f(\mathbf{x}_{n+1})$: compute the contracted point $\mathbf{x}_c = \mathbf{x}_0 + \rho(\mathbf{x}_r - \mathbf{x}_0)$ with $0 < \rho \leq 0.5$. If $\mathbf{x}_c$ satisfies $f(\mathbf{x}_c) < f(\mathbf{x}_r)$, then a new simplex is obtained by replacing $\mathbf{x}_{n+1}$ with $\mathbf{x}_c$ and go to step 1.
     Otherwise, go to step 6.
   - If $f(\mathbf{x}_r) \geq f(\mathbf{x}_{n+1})$: compute the contracted point $\mathbf{x}_c = \mathbf{x}_0 + \rho(\mathbf{x}_{n+1} - \mathbf{x}_0)$ with $0 < \rho \leq 0.5$. If $\mathbf{x}_c$ satisfies $f(\mathbf{x}_c) < f(\mathbf{x}_{n+1})$, then a new simplex is constructed with $\mathbf{x}_c$ and go to step 1.
     Otherwise, go to step 6.
7. Shrinkage: Replace all points except the best, $\mathbf{x}_1$, with $\mathbf{x}_i = \sigma(\mathbf{x}_i - \mathbf{x}_1), 0 < \sigma \leq 0.5$

The algorithm terminates when the standard deviation of the function values of the current simplex falls below a user-initialized tolerance. When the cycle stops the point of the simplex associated with the lower function value is returned as the proposed optimum

The values of the parameters α, γ, ρ and σ were left to default of `SciPy`: $\alpha = 1, \gamma = 2, \rho = 0.5, \sigma = 0.5$.

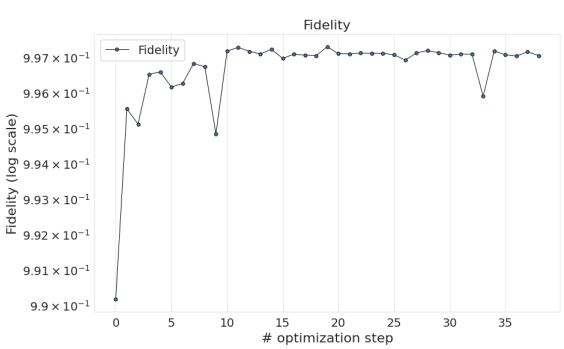
Results

In all applications of the Nelder-Mead optimization algorithm presented in this work, a maximum number of function evaluations was imposed to prevent excessively long experimental runtimes. The primary limitation arises from the cost function itself, which corresponds to the average Clifford gate fidelity estimated via RB. By using the parameters reported in Section (3.2.1), each RB sequence evaluation requires a minimum of 20 seconds, subsequently the evaluation of the cost function is particularly resource-intensive in terms of computation time. To mitigate these constraints, both a cap on the total number of cost function evaluations and a maximum number of optimization steps were defined. It is important to note that each iteration of the NM algorithm typically involves the evaluation of multiple points in parameter space: at least four in the case of optimization over three parameters, and at least three when optimizing over two parameters.

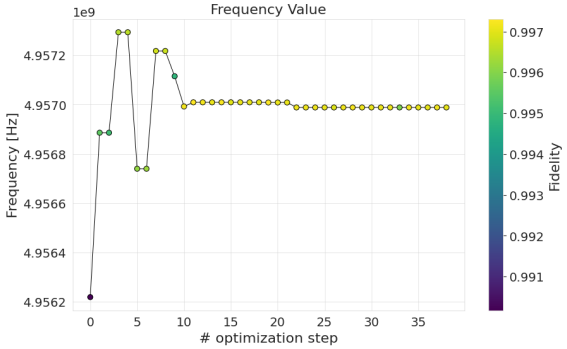
In the first run of the algorithm, the maximum number of iterations was limited at 40 such that no more than 160 cost function evaluations were performed. Additionally, a convergence tolerance of $1 \cdot 10^{-4}$ was set (`tol=1e-4` [3]), applying stopping criteria to both parameter updates and cost function changes.

The results of this initial optimization attempt are shown in Figure (3.1). As illustrated in Figure (3.1a), a clear improvement in gate fidelity was achieved, increasing from an initial value of 99.02% to a final value of 99.71%. Despite this apparent improvement, the optimizer did not converge within the specified tolerance. After forty steps, the stopping criterion was not satisfied, and the optimization was formally marked as unsuccessful. Nevertheless, the progression of the fidelity suggests a stable improvement trend, indicating that the algorithm was moving toward a local optimum even if convergence, as defined by the stopping conditions, was not strictly achieved.

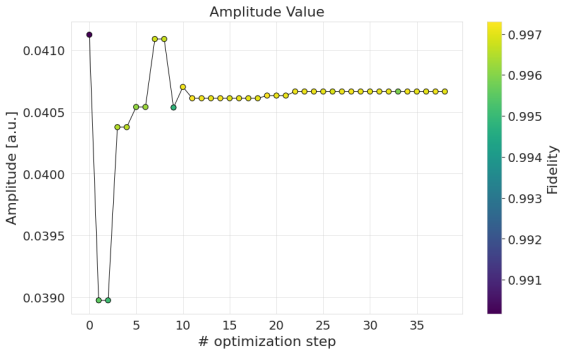
[3] In `scipy.optimize.minimize`, this parameter sets both the tolerance on the optimization variables, referred to as `xtol`, and the tolerance on the cost function value `ftol`.



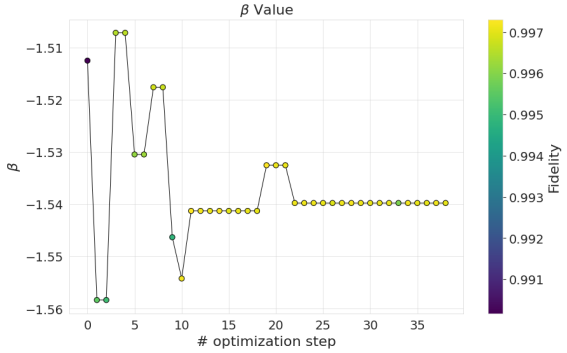
(a) Plot of the average Clifford gate fidelity as a function of the function evaluations.



(b) Plot of the frequency of the $R_X(\pi)$ gate corresponding to different optimization steps.



(c) Plot of the amplitude of the $R_X(\pi)$ gate corresponding to different optimization steps.



(d) Plot of the β multiplying coefficient for the second quadrature component of the DRAG pulse.

Figure 3.1: Plots of the fidelity and optimization parameters as a function of the number of optimization step.

The values of the parameters being optimized are reported in Figures (3.1b), (3.1c), and (3.1d). From these plots, it is again evident that the optimization is converging as the parameters stabilize around specific values: `amplitude` $= 0.04$, `frequency` $= 4.957$ GHz, and `beta` $= -1.54$.

However, a possible limitation to the performance of this optimization approach may lie in the absence of an explicitly defined initial simplex. Setting only the initial parameter values, without specifying the full simplex, can lead the algorithm to explore the parameter space inefficiently. For this reason, further runs of the same method were carried out, this time also initializing the simplex explicitly.

Furthermore, considering that the allowed variation range for the β parameter was restricted to only ± 0.25 around the value determined by the `drag_tuning` routine, quite a small interval, we explored the possibility of improving the efficiency of the optimization by reducing the number of free parameters. Specifically, additional optimization runs were performed in which only the amplitude and frequency of the $R_X(\pi)$ pulse were optimized, while β was kept fixed at 0.

This choice was motivated by the working assumption that, at least in this initial phase, leakage errors were not the dominant source of infidelity. Since the DRAG correction primarily addresses leakage to higher energy levels, we thought that β would have a limited effect on the overall gate fidelity compared to the more significant impact of accurately calibrating the native gate's amplitude and frequency. As such, the focus was placed on parameters expected to yield the greatest improvements in fidelity while maintaining a limited run time.

Some optimization runs were performed with this revised configuration, in this case, the initial simplex was constructed based on preliminary fine-tuning routines. Specifically, we considered the uncertainties on amplitude and frequency obtained respectively from the flipping and Ramsey procedures. These values were scaled by a factor of 1.5 to define the characteristic step sizes; using these values, a triangular simplex was generated in the amplitude-frequency parameter space, centered around the initial guess, and offset along each axis to ensure sufficient coverage of the local landscape.

The results of these runs are reported in the following sections and are labeled as `init_simp_1`, `init_simp_2`, and `init_simp_3`.

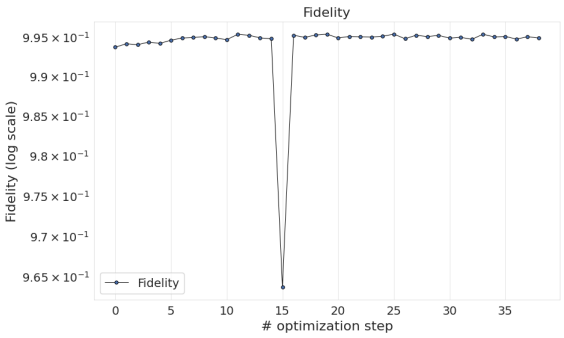
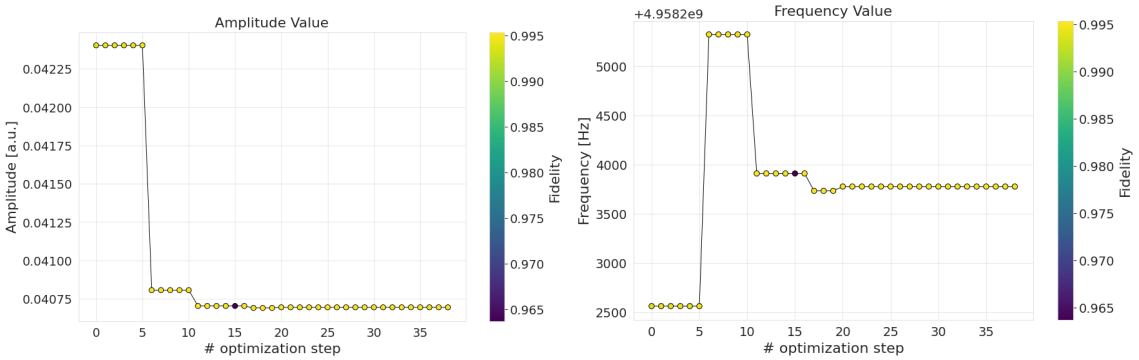


Figure 3.2: Plot of the average Clifford gate fidelity as a function of the optimization steps for the first round optimization - `init_simp_1`.



(a) Plot of the amplitude of the $R_X(\pi)$ gate corresponding to different optimization steps.

(b) Plot of the frequency of the $R_X(\pi)$ gate corresponding to different optimization steps.

Figure 3.3: Plots of the optimization parameters as a function of the number of optimization steps for the first round optimization - `init_simp_1`.

From the plots showing the outcome of the first optimization run, it can be observed that the fidelity increases over the initial iterations and then stabilizes around a value of approximately 99.45% after about fifteen steps. An aspect that is particularly noteworthy is the noticeable drop in fidelity observed at the sixteenth iteration, where the fidelity falls below the 97% threshold (Figure (3.2)), despite the amplitude and frequency of the $R_X(\pi)$ pulse showing no significant changes compared to the previous optimization steps (see Figure (3.3)). Similar behavior is observed in other plots illustrating the optimization progress for this algorithm (see Figure (3.4)), which may point to possible issues in measurement or data acquisition, or to imperfect control over the qubit state.

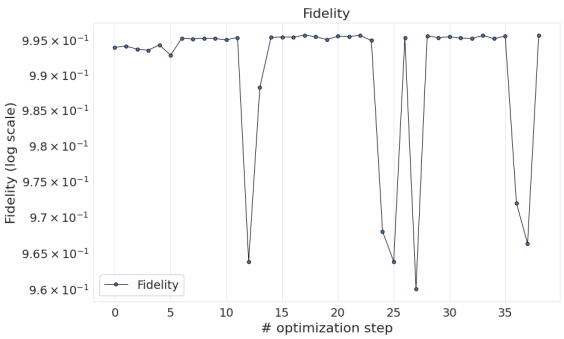
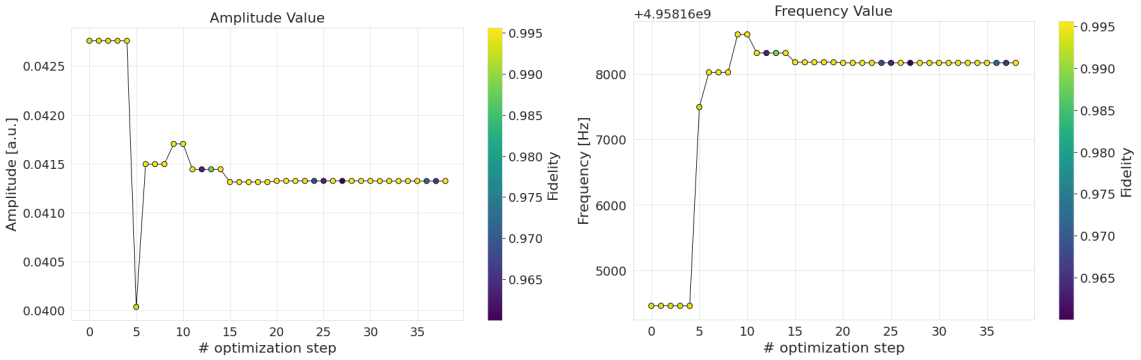


Figure 3.4: Plot of the average Clifford gate fidelity as a function of the optimization steps for the second round optimization - `init_simp_2`.



(a) Plot of the amplitude of the $R_X(\pi)$ gate corresponding to different optimization steps.

(b) Plot of the frequency of the $R_X(\pi)$ gate corresponding to different optimization steps.

Figure 3.5: Plots of the optimization parameters as a function of the number of optimization steps for the first round optimization - `init_simp_2`.

In particular, regarding the optimization labeled as `init_simp_2`, the most important observation is again the presence of large variations in fidelity despite nearly fixed parameters. That is, for comparable values of amplitude and frequency, significantly different fidelity outcomes are observed. This is especially evident when examining the optimization steps corresponding to the pronounced fidelity dips in Figure (3.4). By comparing those same steps with the parameter evolution shown in Figure (3.5), it becomes evident that neither the amplitude nor the frequency of the $R_X(\pi)$ pulse changes significantly. It should be noted, as will be discussed in more detail in the conclusions, that this behavior may ultimately limit the effectiveness of the optimization process in improving the average Clifford gate fidelity.

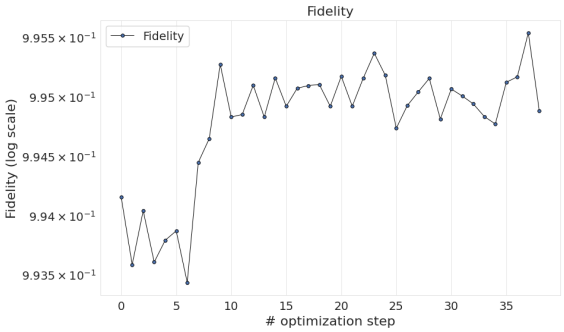
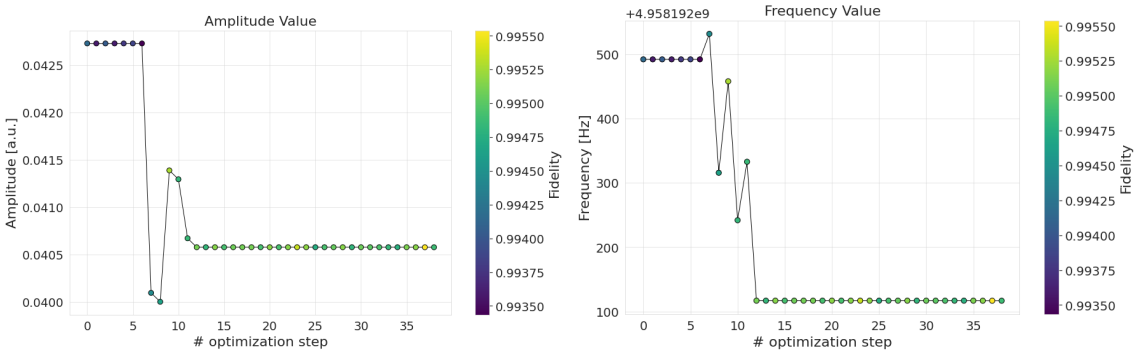


Figure 3.6: Plot of the average Clifford gate fidelity as a function of the optimization steps for the first round optimization - `init_simp_3`.

In the third and final case, the lack of convergence is immediately evident from the fidelity plot shown in Figure (3.6). Upon closer inspection, it was observed that the frequency parameter quickly settles at the lower bound of its defined range. This initially led to the hypothesis that the preceding Ramsey-based fine-tuning step used to initialize the optimization had failed. However, this assumption was disproven by the report automatically generated for each operation performed with `Qibocal` . The corresponding Ramsey experiment, which confirms successful execution, is shown in Figure (3.8). These findings lead us to conclude that in this case, the Nelder Mead algorithm simply failed to converge with the same stability observed in the previous optimization runs.

A summary of the results obtained using the Nelder Mead algorithm under different configurations is presented in the top rows of Table (3.1) while a more detailed summary is reported in Table (3.4) and Table (3.5), at the end of the present chapter.

Overall, it is evident that including the β parameter for DRAG correction in the optimization led to higher average Clifford gate fidelities. For this reason, β was included again in the later optimization runs.



(a) Plot of the amplitude of the $R_X(\pi)$ gate corresponding to different optimization steps.

(b) Plot of the frequency of the $R_X(\pi)$ gate corresponding to different optimization steps.

Figure 3.7: Plots of the optimization parameters as a function of the number of optimization steps for the first round optimization - `init_simp_3`.

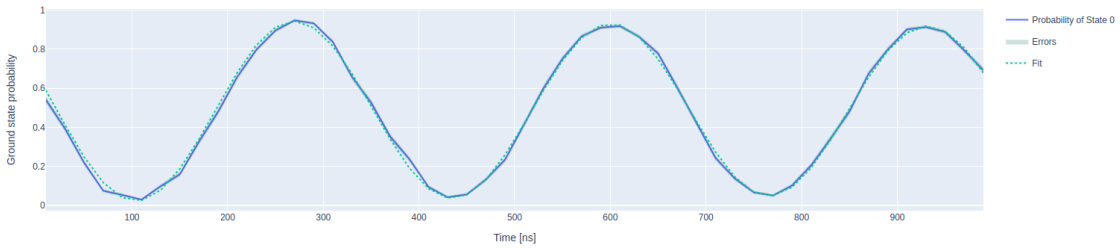


Figure 3.8: Ramsey experiment performed to fine-tune the $R_X(\pi)$ gate frequency before the optimization process.

3.3.2 SLSQP

Algorithm description

To explore possible improvements in performance, we tried using a gradient-based algorithm. In particular, we used the Sequential Least Squares Programming (SLSQP) method available in the `SciPy` library.

The Sequential Least Squares Programming (SLSQP) method, originally developed by Dieter Kraft in 1988 [44], is a gradient-based algorithm for solving constrained nonlinear optimization problems. The algorithm solves a nonlinear constrained problem of the form:

$$\begin{aligned} &\min_{x \in \mathbb{R}^n} && f(x) \\ &\text{subject to} && c_i(x) = 0, \quad i = 1, \ldots, m \\ &&& d_j(x) \geq 0, \quad j = 1, \ldots, p \\ &&& \mathbf{x}^{(L)} \leq \mathbf{x} \leq \mathbf{x}^{(U)} \end{aligned}$$

It belongs to the class of Sequential Quadratic Programming (SQP) methods, which iteratively solve a sequence of quadratic programming subproblems that locally approximate the original nonlinear problem.

1. Linearization: At iteration k, the nonlinear constraints are approximated by their first-order Taylor expansion around the current point $\mathbf{x}_k$. This transforms the nonlinear constraints into a linear system locally.

2. Quadratic subproblem construction: The objective is approximated by a quadratic model of the Lagrangian function:

$$\mathcal{L}(\mathbf{x}, \lambda, \mu) = f(\mathbf{x}) - \sum_{i=1}^{m} \lambda_i c_i(\mathbf{x}) - \sum_{j=1}^{p} \mu_j d_j(\mathbf{x}), \tag{3.6}$$

where λ_i and μ_j are Lagrange multipliers. The Hessian of the Lagrangian is not computed explicitly but approximated using a BFGS-like quasi-Newton update.

3. QP Subproblem solution: A quadratic program is solved to determine a search direction $\mathbf{p}_k$, subject to the linearized constraints and bounds. The subproblem minimizes the quadratic model subject to these constraints.

4. Constrained line search: A line search is performed along $\mathbf{p}_k$ using a merit function that balances reduction in the objective with the feasibility of constraints. This helps ensure global convergence.

5. Update: The iterate is updated via $\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{p}_k$, where α_k is the step size determined by the line search. The quasi-Newton approximation of the Hessian is also updated.

6. Termination: The algorithm terminates when the norm of the projected gradient and constraint violations fall below a specified tolerance.

If analytic derivatives for the objective or constraints are not provided, they are approximated using forward finite differences. In such cases, each gradient estimation requires $n + 1$ function evaluations per iteration, where n is the number of variables. The values of internal parameters, such as merit function balancing terms and quasi-Newton updates, are managed internally and are not user-configurable through `SciPy`.

Results

For the `SLSQP` optimization run, the same constraints and parameter bounds used in the Nelder-Mead optimization were applied: a maximum of 40 optimization steps and parameter ranges set by the initial fine-tuning routine carried out before the fidelity optimization.

As shown in Figure (3.9a), the optimization procedure appears to converge, a result that is also confirmed by the object returned by the `SciPy` optimizer, where the `success` assumes has the `true` value. This indicates that the algorithm successfully reached a local minimum within the specified tolerance of $1 \cdot 10^{-4}$ in less the 15 optimization steps. On the other hand, it is important to consider that SLSQP is a gradient-based method and, as such, may lack robustness in the presence of a noisy or irregular cost landscape. This makes it susceptible to becoming trapped in local minima, especially if the optimization is not ideal.

In this case, an interesting observation is that, as expected, a sharp drop in fidelity corresponds to a frequency value that deviates significantly from the optimal one for the $R_X(\pi)$ gate. This behavior is consistent with the assumption that frequency and amplitude are the two dominant parameters in determining the quality of the $R_X(\pi)$ gate and as a consequence the average fidelity of Clifford gates.

However, as can be seen in Figure (3.9a), the convergence behavior is not particularly stable: this is also due to the fact that the SLSQP method may not be robust enough to be reliably used as a systematic approach for optimizing average Clifford gate fidelity. In particular, it becomes difficult to define a minimum number of steps beyond which a certain fidelity level can be guaranteed.

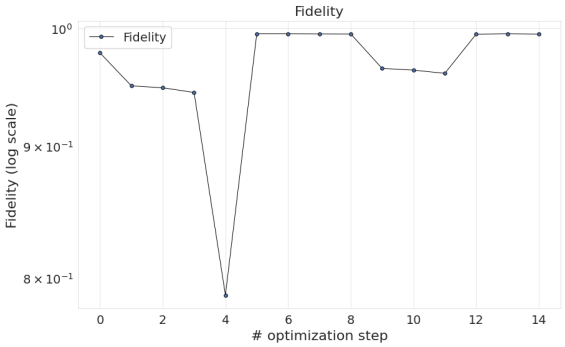
Furthermore, from a computational standpoint, this method is relatively inefficient. Since the gradient must be estimated numerically, each step involves multiple evaluations of the cost function, significantly increasing the total number of function calls required (see Table (3.1)).

3.4 CMA-ES

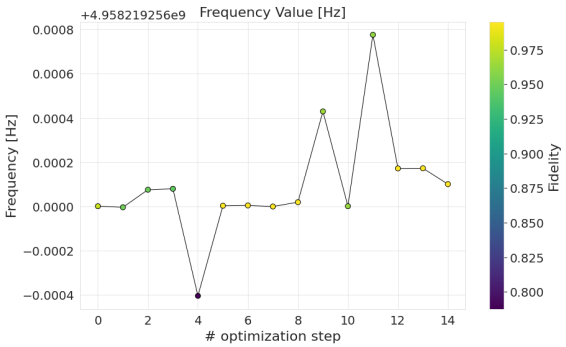
3.4.1 Algorithm description

Covariance Matrix Adaptation Evolution Strategy (`CMA-ES` [45]), is a population-based evolutionary algorithm designed for optimizing complex, non-convex, and high-dimensional functions.
It belongs to the broader class of Evolution Strategies (ES), a subset of Evolutionary Algorithms (EAs)(see [46]), and is particularly effective for black-box optimization where gradient information is unavailable.

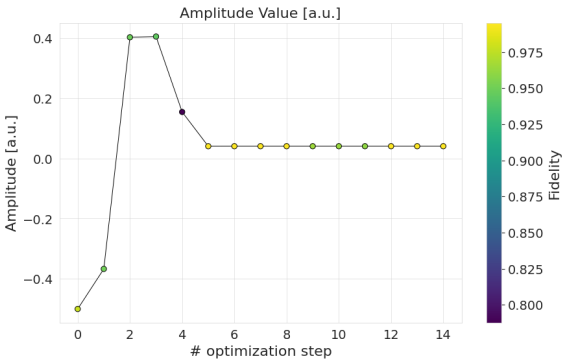
Evolution Strategies (ES) are a class of optimization methods that employ self-adaptive mechanisms to explore the search space efficiently. Unlike classical optimization techniques that rely on gradient descent, ES leverages stochastic sampling to navigate rugged and multimodal landscapes.



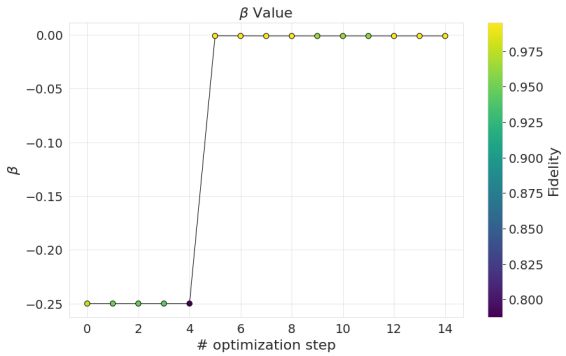
(a) Plot of the average Clifford gate fidelity as a function of the optimization steps.



(b) Plot of the frequency of the $R_X(\pi)$ gate corresponding to different optimization steps.



(c) Plot of the amplitude of the $R_X(\pi)$ gate corresponding to different optimization steps.



(d) Plot of the β multiplying coefficient for the second quadrature component of the DRAG pulse.

Figure 3.9: Plots of the fidelity and optimization parameters as a function of the number of optimization steps for the SLSQP optimization method.

In this context, `CMA-ES` is an adaptive stochastic search method that iteratively refines a probability distribution over the search space. Unlike traditional Genetic Algorithms (GAs), which rely on crossover and mutation operators, `CMA-ES` employs a multivariate normal distribution to generate candidate solutions. The method adaptively updates the distribution's mean and covariance matrix based on the fitness of sampled points.

The fundamental idea behind `CMA-ES` is the use of a multivariate Gaussian distribution to model promising search directions. Let μ_t denote the mean of the distribution at iteration t, and Σ_t the covariance matrix. Then, a new population of λ candidate solutions $\mathbf{x}_i^{(t+1)} \sim \mu_t + \sigma_t \mathcal{N}(\mu_t, \sigma_t^2, \Sigma_t)$, where σ_t is a step size controlling the exploration.

The `CMA-ES` algorithm follows the following steps:

1. If not otherwise specified, the initial parameters are set: mean vector $\mu_\mathbf{0}$, covariance matrix Σ_0[4], step size σ_0, population size λ
2. Generate λ new candidate solutions $\mathbf{x}_i$ according to a multivariate normal distribution.
3. Evaluate the objective function $f(\mathbf{x}_i)$ for each candidate solution.
4. Sort the new candidate solutions based on fitness: $f(\mathbf{x}_0) \leq ... \leq f(\mathbf{x}_\lambda)$.
5. Update the mean vector μ with the $m = \lfloor \lambda/2 \rfloor$ top performing solutions:

$$\mu \leftarrow \sum_{i=0}^{m} \mathbf{w}_i \mathbf{x}_i, \tag{3.7}$$

   where $\mathbf{w}_i$ are internally defined weights.
6. Update the isotropic and anisotropic evolution path $\mathbf{p}_\sigma$, $\mathbf{p}_c$ [5].

[4] $\Sigma_0 = \mathbb{I}$ for isotropic search

[5] For details on the update process of the evolution paths see [45].

Method	Success	Infidelity	Iterations	RB Evaluations	Duration [s]
Nelder-Mead	False	0.002732	40	93	2892
init_simplex_1	False	0.004593	40	99	3096
init_simplex_2	False	0.004297	40	100	3128
init_simplex_3	False	0.004507	40	106	3190
SLSQP	True	0.004861	15	180	3873

Table 3.1: This table provides a short summary of the results obtained using different optimization methods available in `SciPy`.
The *Success* column indicates whether the algorithm converged successfully. A value of True denotes that convergence was achieved within the specified tolerance of $1 \cdot 10^{-4}$ and before reaching the maximum number of iterations; otherwise, the value is False.
The reported value of the objective function corresponds to the average Clifford gate infidelity obtained at the end of optimization via Randomized Benchmarking (1-fidelity), which serves as the cost function minimized by the algorithm. The *Duration* column refers to the total runtime of the script, including the initial fine-tuning stage preceding the optimization.

7. Update the covariance matrix:

$$C \leftarrow (1 - c_1 - c_\mu)C + c_1 \mathbf{p}_c \mathbf{p}_c^T + c_\mu \sum_{i=1}^{\mu} w_i \mathbf{y}_i \mathbf{y}_i^T, \tag{3.8}$$

   where c_1 and c_μ are learning rates and $\mathbf{y}_i$ represents the deviation of the i-th candidate solution from the mean **mu**.

8. Update the step size using a cumulative path evolution mechanism

$$\sigma \leftarrow \sigma \cdot \exp\left(\frac{c_\sigma}{d_\sigma}\left(\|\mathbf{p}_\sigma\| - E\|\mathcal{N}(0, I)\|\right)\right), \tag{3.9}$$

   where c_σ is the learning rate for step-size adaptation, d_σ is a damping factor $\|\mathbf{p}_\sigma\|$ is the length of the evolution path and $E\|\mathcal{N}(0, I)\|$ is the expected length of a standard normally distributed random vector.

We decided to test `CMA-ES` for the RB optimization as it is particularly well-suited for noisy or non-smooth objective functions. The stochastic nature of `CMA-ES` allows it to remain robust against such perturbations, as it does not rely on local gradient information and instead, samples across a distribution to infer promising search directions.

Unless manually specified, `CMA-ES` employs internal heuristics to determine several of its hyperparameters. For instance, the population size λ is typically set according to the dimensionality of the problem:

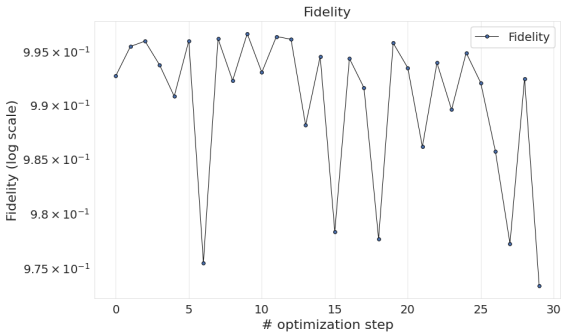
$$\lambda = 4 + 3\lfloor \log n \rfloor \tag{3.10}$$

where n is the problem dimension [47]. For full details on default parameter settings and adaptation rules, see [45].

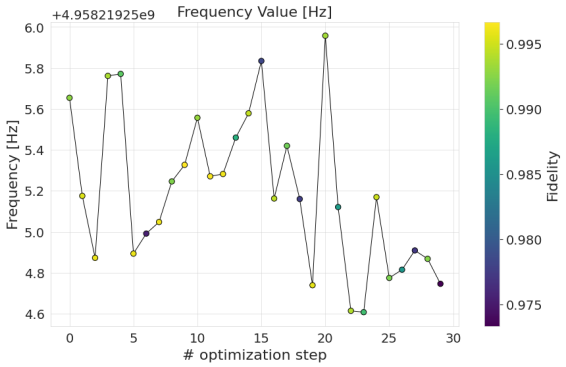
3.4.2 Results

In the `CMA-ES` optimization run, parameter settings were chosen to be as consistent as possible with those used in previous optimization algorithms. Specifically, the initial values and the parameter bounds were defined as shown in Listing (3.1), based on the results of the preliminary fine-tuning procedure. The maximum number of generations was set to 30, slightly lower than the iteration limits used in the `SciPy`-based methods. This choice was motivated by two main considerations. First, we expected a performance comparable to that of the previously tested algorithms, and that consequently, 30 optimization steps (or generations) would be enough to reach an improved fidelity value. Second, we expected the algorithm to require longer execution times due to the larger number of cost functions evaluation: the population size was not manually set but allowed to default to the internal rule of the `CMA-ES` implementation, as in Equation (3.10). In this

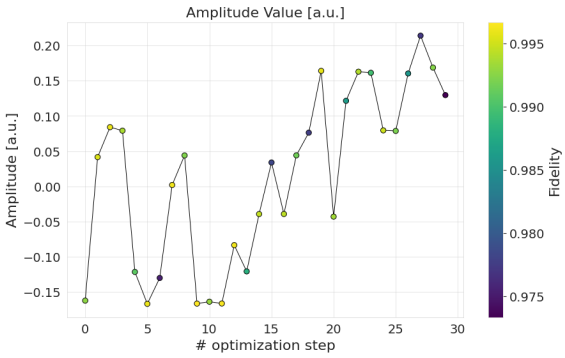
case, with three parameters, pulse amplitude, frequency, and DRAG correction parameter β, the population consisted of 7 candidate solutions per generation. Given a population of 7 individuals and a total of 30 generations, the optimization required 210 evaluations of the objective function to complete.



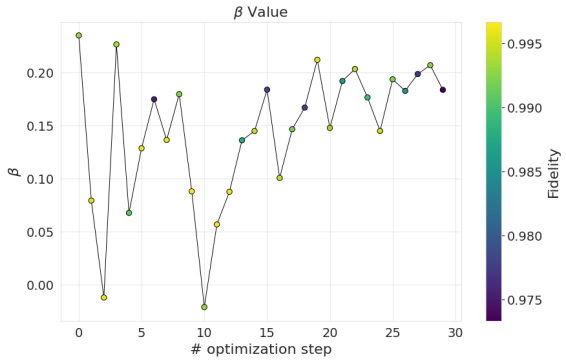
(a) Plot of the average Clifford gate fidelity as a function of the optimization steps.



(b) Plot of the frequency of the $R_X(\pi)$ gate corresponding to different optimization steps.



(c) Plot of the amplitude of the $R_X(\pi)$ gate corresponding to different optimization steps.



(d) Plot of the β multiplying coefficient for the second quadrature component of the DRAG pulse.

Figure 3.10: Plots of the fidelity and optimization parameters as a function of the number of optimization steps for the CMA optimization method.

The results of the fidelity optimization using the `CMA-ES` algorithm are presented in Figure (3.10). Each point in the plots corresponds to the best-performing individual within the population at each of the 30 optimization steps (generations). As shown, the algorithm does not achieve full convergence within the allowed number of generations. This outcome is likely influenced by the relatively small population size used. Indeed, previous studies in the literature have demonstrated that larger populations can significantly improve the performance per function evaluation [48], obviously at the cost of increased computational time.

The choice to limit both the number of generations and the population size was made deliberately to maintain a manageable total runtime. Increasing either of these parameters would have led to a substantial rise in computational cost, as `CMA-ES` evaluates the objective function λ times per generation where λ is the population size.

Another factor that negatively influenced the algorithm convergence is the choice of the initial step size parameter, $\sigma = 0.25$. In `CMA-ES` σ controls the standard deviation of the multivariate normal distribution used to generate the first population, thereby determining how broadly the algorithm explores the parameter space around the initial guess. A smaller σ would have concentrated the search in a narrower region, possibly leading to faster convergence to a local optimum. However, this value of $\sigma = 0.25$ was chosen with the goal of exploring a wider portion of the parameter space than in previous `SciPy`-based optimizations, in order to investigate the possible presence of higher-fidelity regions that had not yet been sampled. An evaluation of the computational cost of running the `CMA-ES` algorithm is reported in Table (3.2).

3.5 Optuna

3.5.1 Algorithm description

The Tree-Structured Parzen Estimator (TPE) algorithm, as implemented in the `Optuna` optimization framework [49], is a model-based, derivative-free optimization method. `Optuna` belongs to the class of Sequential Model-Based Optimization (SMBO) strategies [50], which iteratively refine a surrogate model of the objective function based on past evaluations and select new points in parameter space by balancing exploration and exploitation [51].

The TPE method differs from classical Gaussian Process-based Bayesian optimization by using non-parametric density estimation to guide the search through parameter space. Instead of directly modeling the objective function $f(\mathbf{x})$, TPE models the conditional probability distribution of the parameters $\mathbf{x}$ given the value of the objective function.

Given a set of past observations $\{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$, where $y_i = f(\mathbf{x}_i)$, the algorithm proceeds as follows:

1. Split the observations: Define a threshold value y^*, typically chosen as a quantile of the observed objective values. The dataset is then split into two subsets:
   - $\mathcal{L} = \{\mathbf{x}_i : y_i < y^*\}$, the set of promising configurations
   - $\mathcal{G} = \{\mathbf{x}_i : y_i \geq y^*\}$, the set of less promising configurations
2. Estimate densities: Construct two kernel density estimators (KDEs) to approximate the conditional densities:
$$l(\mathbf{x}) \approx p(\mathbf{x} \mid y < y^*) \quad \text{and} \quad g(\mathbf{x}) \approx p(\mathbf{x} \mid y \geq y^*).$$
3. Candidate sampling: Draw a number of candidate parameter configurations $\mathbf{x}$ from the distribution $l(\mathbf{x})$, which is biased toward more promising regions of the search space.
4. Acquisition function: For each sampled candidate $\mathbf{x}$, compute the ratio
$$\mathrm{EI}(\mathbf{x}) = \frac{l(\mathbf{x})}{g(\mathbf{x})},$$
which serves as an acquisition function analogous to expected improvement. The candidate with the highest ratio is selected for evaluation.
5. Evaluate and update: Evaluate the objective function $y = f(\mathbf{x})$ at the selected configuration and append the result to the history. Return to step 1.

This process continues until a stopping criterion is met, such as a maximum number of trials or convergence of the best-observed value.

Differently from other TPE implementations, `Optuna` supports automatic pruning designed to terminate unpromising trials early, thus reducing the computational cost. In common applications such as neural network training, pruning is enabled by reporting intermediate objective values via `trial.report(...)` and comparing them to statistics of previously completed trials. By default, `Optuna` employs the `MedianPruner`, which terminates a trial if its intermediate value is worse than the median of completed trials at the same step.

In the present work, the cost function is defined as the infidelity extracted from a complete RB routine and the optimization problem involves a low-dimensional parameter space, comprising only three continuous parameters; given these characteristics, pruning was not applied.

3.5.2 Results

As a first attempt at applying the optimization algorithm implemented in the `Optuna` library, we decided to perform a total of 30 optimization steps, or trials as referred to in the library itself. This choice was made to maintain consistency with the number of steps used in previous optimization methods. As in the earlier cases, initial parameter values and search boundaries were provided to the algorithm and derived from the fine-tuning routines executed prior to the optimization.

The results obtained with this first optimization attempt are reported in Figure (3.11) - (3.14).

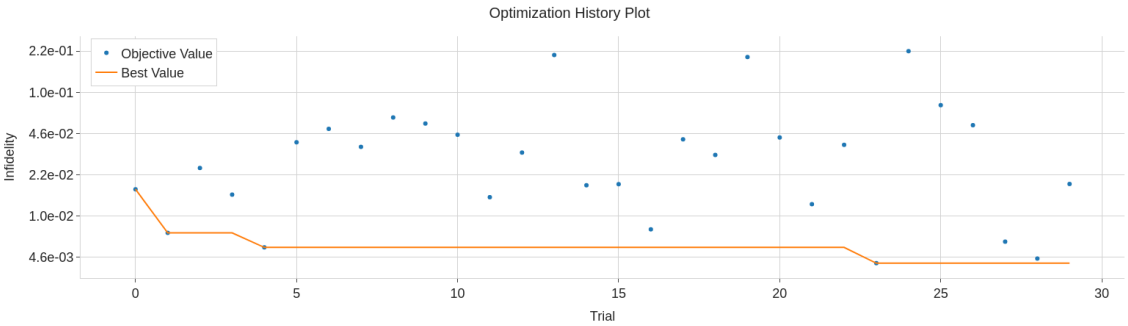


Figure 3.11: Plot of the optimization history using `Optuna` for 30 trials. Infidelity as a function of trials.

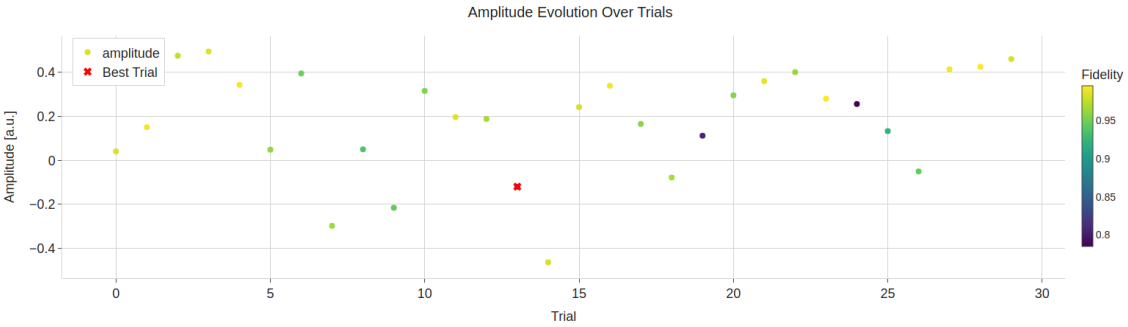


Figure 3.12: Plot of the $R_X(\pi)$ amplitude as a function of the number of trials. The color map indicates the Clifford gate average fidelity. The red cross denotes the $R_X(\pi)$ amplitude corresponding to the highest observed fidelity.

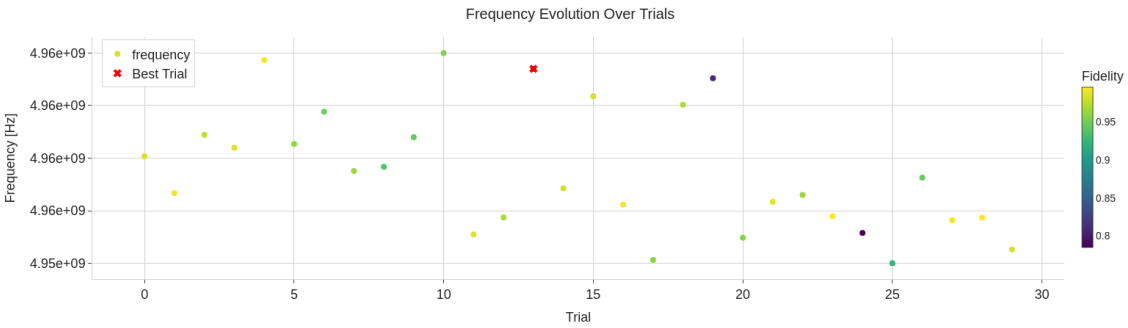


Figure 3.13: Plot of the $R_X(\pi)$ frequency as a function of the number of trials. The color map indicates the Clifford gate average fidelity. The red cross denotes the $R_X(\pi)$ frequency corresponding to the highest observed fidelity

The first observation from the optimization plots is that no clear convergence is achieved. This behavior is not unexpected and can be attributed to the nature of the `Optuna` optimization framework, which is not designed to guarantee convergence in the traditional sense. Unlike gradient-based methods or some evolutionary strategies, `Optuna` is more exploratory by design and may not settle into a stable optimum within a fixed number of trials.

Instead, `Optuna` proves valuable as a tool for systematically exploring the parameter space. In doing so, it may identify previously unvisited regions that yield better performance. In this specific run, the highest fidelity was achieved at the 23rd trial, reaching an average Clifford gate fidelity of 99.58%. A result that is comparable to the best fidelities obtained using the other optimization strategies tested.

However, a clear advantage of using `Optuna` is its efficiency. The entire optimization process, comprising only 30 function evaluations, was completed in approximately 12 minutes; see Table (3.2) for the exact runtime in seconds. Due to this relatively low computational cost, it becomes feasible to explore a larger number of trials to assess whether higher fidelities can be achieved or

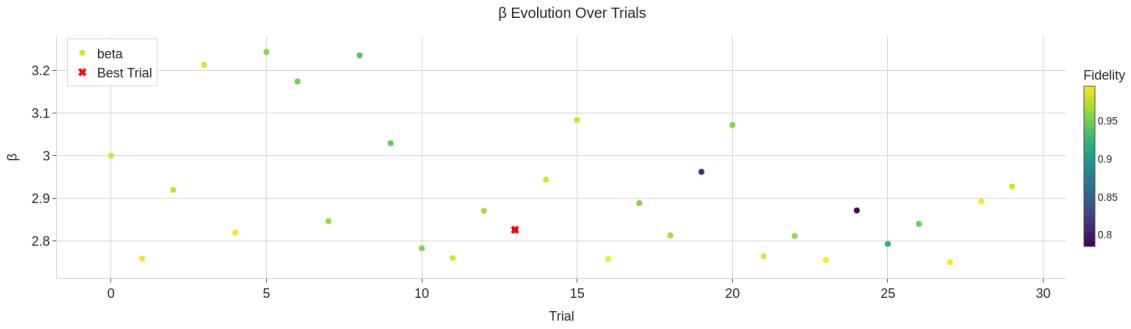


Figure 3.14: Plot of the value of the β coefficient for the second quadrature of the $R_X(\pi)$ pulse as a function of the number of trials. The color map indicates the Clifford gate average fidelity. The red cross denotes the β value corresponding to the highest observed fidelity

whether the algorithm can exhibit more stable convergence patterns.

Based on this reasoning, a second optimization run was performed using 100 trials. The evolution of the fidelity during this extended optimization is shown in Figure (3.15). Contrary to initial expectations that a longer optimization would yield better results, it was observed that the lowest cost function value (i.e., highest fidelity) was reached within the first 20 steps. In particular, the best average Clifford gate fidelity of 99.64% was obtained at the 13^{th} trial.

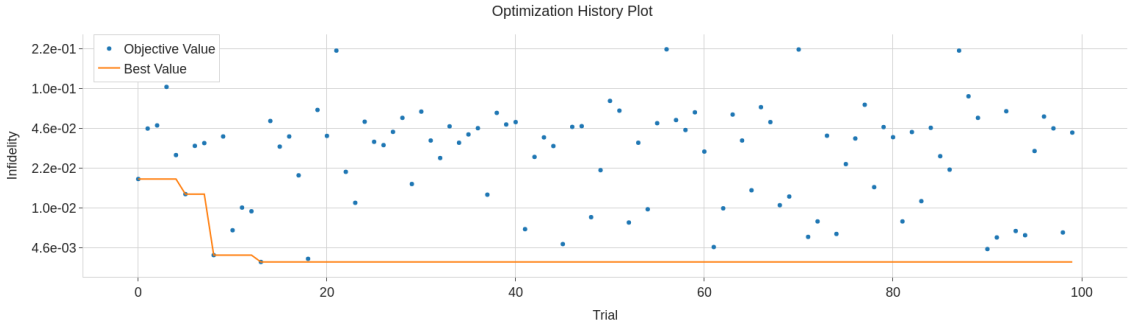


Figure 3.15: Plot of the optimization history using `Optuna` for 100 trials. Infidelity as a function of trials.

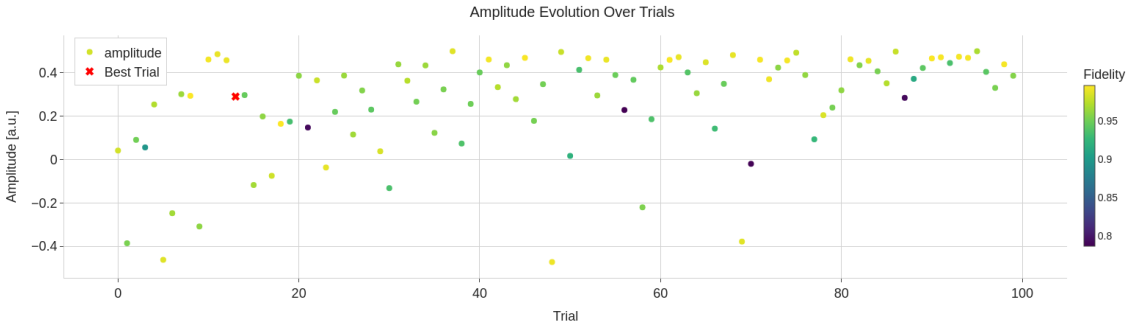


Figure 3.16: Plot of the $R_X(\pi)$ amplitude as a function of the number of trials. The color map indicates the Clifford gate average fidelity. The red cross denotes the $R_X(\pi)$ amplitude corresponding to the highest observed fidelity.

An analysis of the plots and the numerical values obtained for the $R_X(\pi)$ pulse parameters shows that the highest Clifford gate average fidelity corresponds to distinct values of these parameters. This effect is particularly evident for the amplitude and β parameters. While the amplitude range remains fixed, being constrained by hardware limitations, the range used to search for the optimal β varies significantly across the two optimization runs.

Although β is not the most critical parameter in determining the fidelity outcome, as highlighted

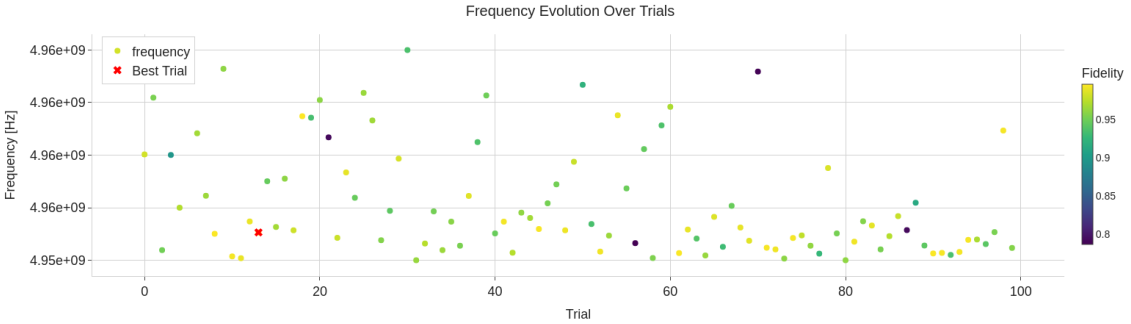


Figure 3.17: Plot of the $R_X(\pi)$ frequency as a function of the number of trials. The color map indicates the Clifford gate average fidelity. The red cross denotes the $R_X(\pi)$ frequency corresponding to the highest observed fidelity.

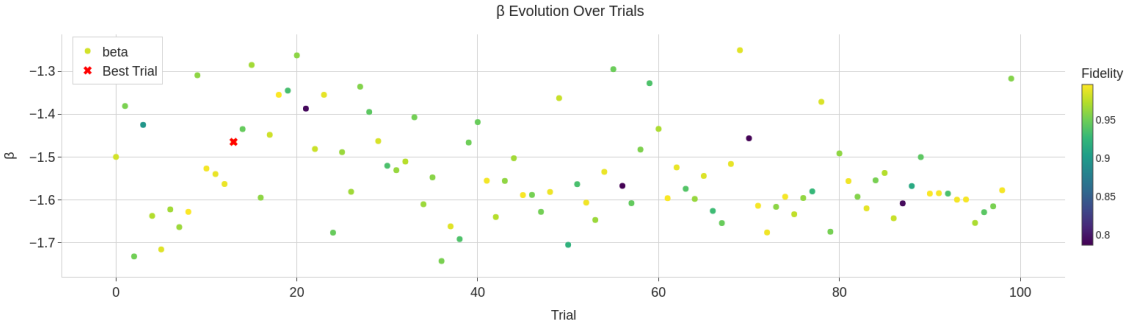


Figure 3.18: Plot of the value of the β coefficient for the second quadrature of the $R_X(\pi)$ pulse as a function of the number of trials. The color map indicates the Clifford gate average fidelity. The red cross denotes the β value corresponding to the highest observed fidelity

also by the Optuna analysis, which indicates that β accounts on average for approximately 9% of the fidelity improvement, its proper calibration does have a measurable impact on the shape of the control pulse and, therefore, on the overall gate performance.

The reason for the significant variation in the allowed search ranges for β was then traced back to a failure in the fine-tuning routine responsible for calibrating the DRAG parameter. The failed fitting routines are shown in Figure (3.19). As a result of this issue, the optimization workflow was modified to include control on the χ^2 value obtained from the fitting procedure to monitor potential failures in the pre-optimization stage.

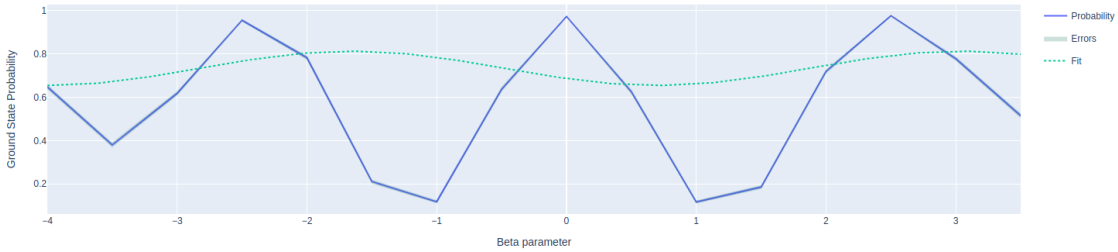
Given the results obtained in the two previous optimization runs using `Optuna`, we proceeded with a third round of optimization, this time ensuring control over the execution of the fine-tuning routines that precede the optimization itself. In consideration of the relatively short time required by `Optuna` in the previous attempt, approximately 40 minutes for 100 trials, we decided to significantly increase the number of optimization trials. Specifically, we set an upper limit of 1000 trials, to improve parameter space exploration and again potentially achieve better convergence.

The optimization history is shown in Figure (3.20). At first, the results appear promising: the algorithm seems to achieve an average Clifford gate fidelity of 99.99%. However a closer examination of the outputs corresponding to all RB routines with reported average Clifford gate infidelities below $1.5 \cdot 10^{-3}$, it becomes evident that these results are actually the consequence of failed exponential fits on the RB output.

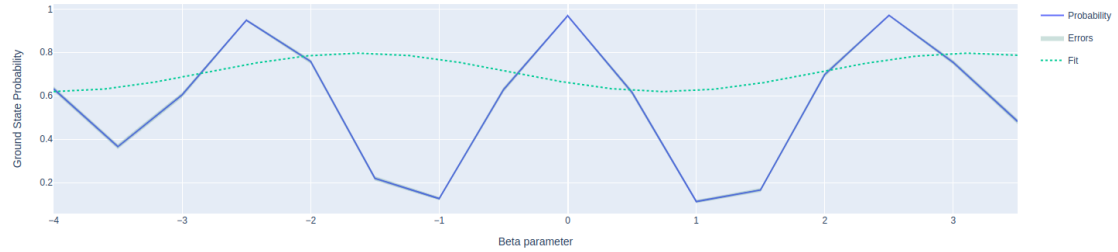
Figure (3.21) presents a few representative examples of such cases, specifically showing the RB plots corresponding to the four lowest infidelities obtained during the optimization.

At this point, the analysis was repeated after discarding all data points associated with a cost function value lower than $1.5 \cdot 10^{-3}$. The revised optimization history is shown in Figure (3.22). In this analysis, the best fidelity achieved was 99.85% much lower than the previously reported 99.98%, but still a satisfactory result. This outcome corresponds to an improvement in fidelity of approximately 0.37% compared to the initial configuration.

The evolution of the optimized parameters over the trials of the `Optuna` run is shown in Figures (3.23) - (3.25). Once again, the most interesting parameter to analyze is the DRAG coefficient β,



(a) Plot of the output of the `drag_tuning` routine performed before running the 30 steps optimization with `Optuna`.



(b) Plot of the output of the `drag_tuning` routine performed before running the 100 steps optimization with `Optuna`.

Figure 3.19: Plots failed fine-tuning routines for estimating the β parameter for the DRAG pulse shape. From the plot is clear that the fitting procedure failed, a possible improvement to allow for a better fit could have been the choice of a smaller step size to scan the possible β values.

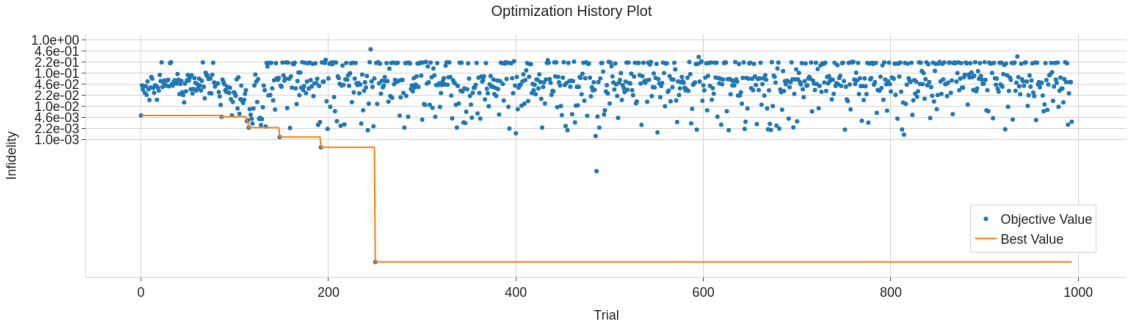


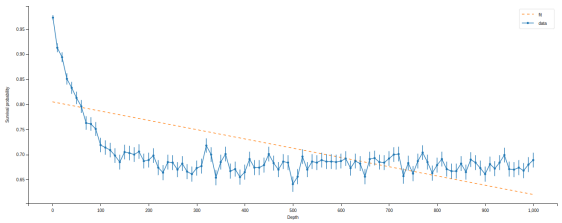
Figure 3.20: Plot of the optimization history using `Optuna` for 1000 trials. Infidelity as a function of trials.

whose sampled values are clearly centered around zero. Notably, the optimal value found for β is approximately $-510 \cdot 10^{-6}$, a value so close to zero that it effectively corresponds to a purely Gaussian pulse shape.

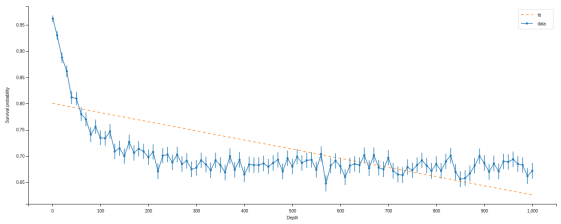
As for the amplitude and frequency parameters, it is reassuring to observe that the best-performing values lie within the most densely sampled regions of the parameter space. This supports the conclusion that the optimization process executed by `Optuna` behaved as expected, effectively concentrating the search around the most promising areas. Despite these observations, the system does not appear to be particularly stable, as small variations in the parameter values lead to significant fluctuations in the resulting fidelity. Overall, the results presented so far suggest that the optimization process carried out using `Optuna` does not exhibit clear convergence.

Based on the results obtained from the previous optimization run, particularly concerning the parameter β, a second optimization and measurement sequence was performed using the same configuration but varying only the amplitude and frequency of the $R_X(\pi)$ gate. The results of this new optimization run are presented in Figure (3.26). In this case, the minimum of the cost function was reached at the 865^{th} step, corresponding to an average Clifford gate fidelity of 99.85%.

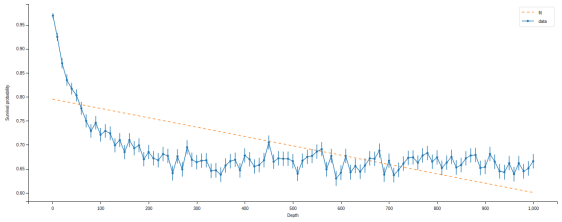
This result is comparable to that obtained in the previous run, and it suggests two main conclusions. First, under the current fidelity regime, optimization over the parameter β does not



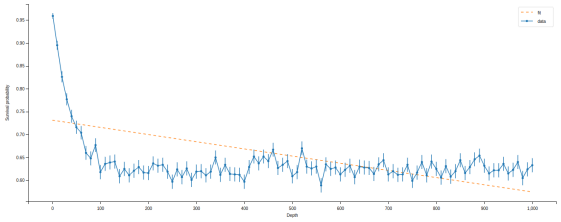
(a) Plot of the output of RB protocol for trial 250.



(b) Plot of the output of RB protocol for trial 486.



(c) Plot of the output of RB protocol for trial 192.



(d) Plot of the output of RB protocol for trial 148.

Figure 3.21: Plots of the failed RB fit for different `Optuna` trials.

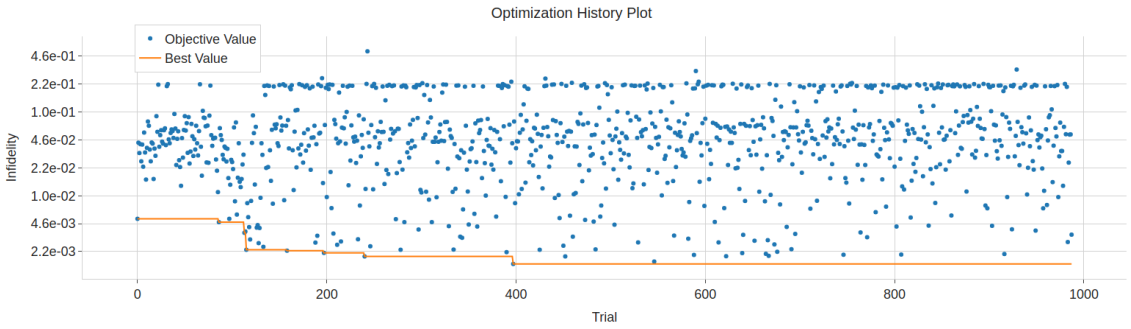


Figure 3.22: Plot of the optimization history using `Optuna` for 1000 trials and excluding the data points associated to RB routines where the fitting procedure failed. Infidelity as a function of trials.

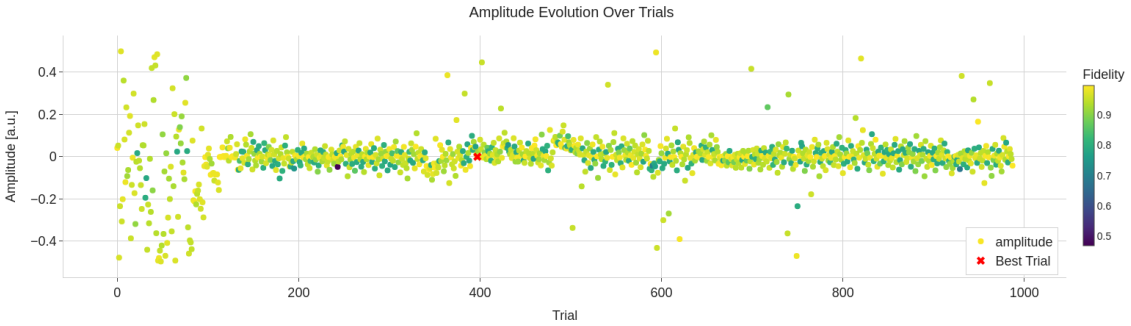


Figure 3.23: Plot of the $R_X(\pi)$ amplitude as a function of the number of trials. The colormap indicates the Clifford gate average fidelity. The red cross denotes the $R_X(\pi)$ amplitude corresponding to the highest observed fidelity.

appear to play a significant role in improving performance. Second, it is likely that we have reached a saturation point in the cost landscape, where further improvements to the average Clifford gate fidelity cannot be achieved by tuning only the three parameters considered so far. This indicates that additional parameters or alternative control strategies may be required to surpass the current fidelity limit.

The table below summarizes the infidelity values obtained at the final iteration for each of the

Figure 3.24: Plot of the $R_X(\pi)$ frequency as a function of the number of trials. The colormap indicates the Clifford gate average fidelity. The red cross denotes the $R_X(\pi)$ frequency corresponding to the highest observed fidelity.

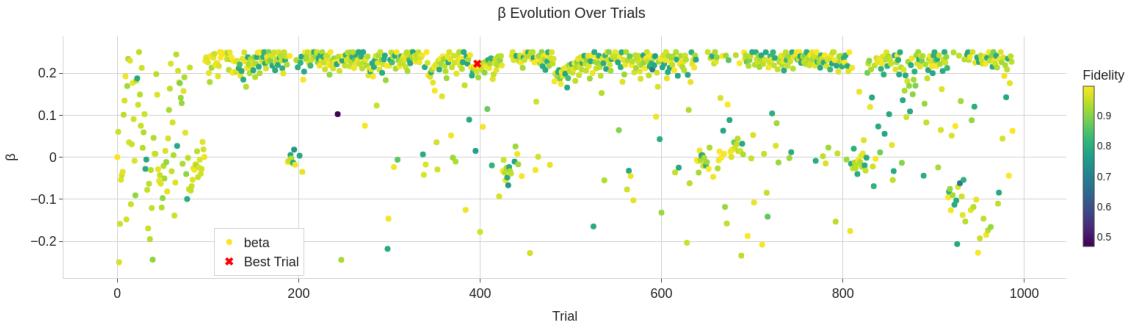


Figure 3.25: Plot of the value of the β coefficient for the second quadrature of the $R_X(\pi)$ pulse as a function of the number of trials. The colormap indicates the Clifford gate average fidelity. The red cross denotes the β value corresponding to the highest observed fidelity

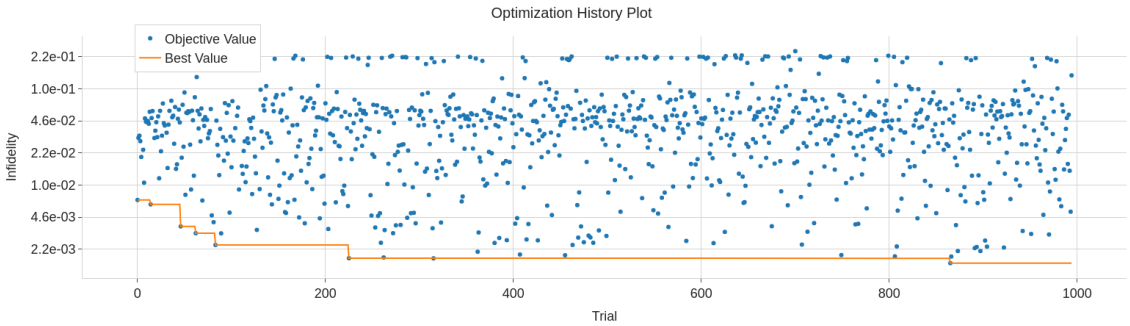


Figure 3.26: Plot of the optimization history using `Optuna` for 1000 trials. Infidelity as a function of trials.

optimization algorithms tested in this study. It also reports the number of optimization steps and function evaluations performed, as well as the total runtime required to complete each optimization.

3.6 RB optimization conclusions

As outlined at the beginning of this chapter, the primary goal was to evaluate whether an automated optimization procedure, executed directly on quantum hardware, could reliably improve the average fidelity of single-qubit Clifford gates, specifically the $R_X(\pi)$ gate. The desired outcome was a routine that could be executed periodically, improving gate fidelity in a reproducible and time-efficient manner.

The results obtained from various optimization strategies, Nelder-Mead, `CMA-ES`, and `Optuna`, demonstrate that while meaningful improvements in fidelity are achievable, the current implementation does not yet meet the robustness and efficiency requirements for routine deployment. The

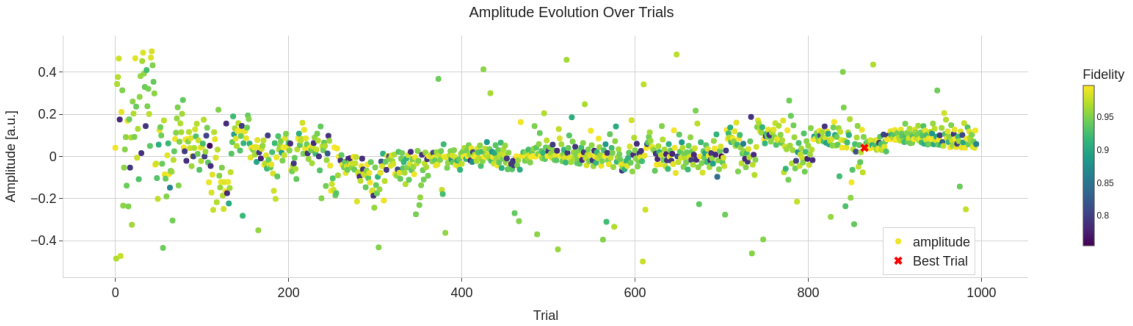


Figure 3.27: Plot of the $R_X(\pi)$ amplitude as a function of the number of trials. The colormap indicates the Clifford gate average fidelity. The red cross denotes the $R_X(\pi)$ amplitude corresponding to the highest observed fidelity.

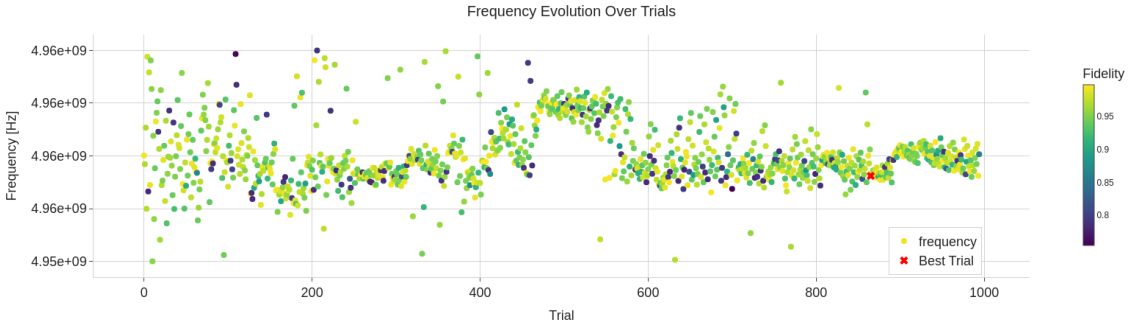
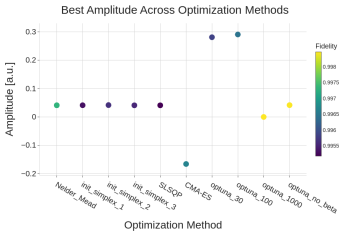


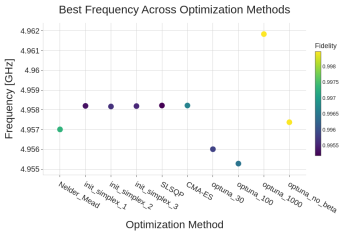
Figure 3.28: Plot of the $R_X(\pi)$ frequency as a function of the number of trials. The colormap indicates the Clifford gate average fidelity. The red cross denotes the $R_X(\pi)$ frequency corresponding to the highest observed fidelity.

primary bottleneck remains the cost of the Randomized Benchmarking (RB) routine, which is necessary for fidelity evaluation. Because each optimization requires multiple evaluations of this cost function, the total execution time quickly becomes a limiting factor.

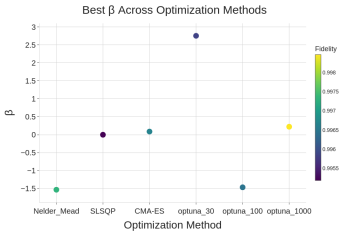
Additionally, we found that convergence is not guaranteed. In several cases, especially those involving `Optuna` or `CMA-ES` with a broader parameter search, the optimization did not yield stable or reproducible results. Even when high fidelities were reached (up to 99.85%), the fidelity showed large sensitivity to small changes in parameter values, casting doubt on the practical stability of the identified configurations. This is particularly evident from the results shown in Table (3.3) and illustrated in Figure (3.29).



(a) Amplitude values corresponding to highest fidelity for each optimization method.



(b) Frequency values corresponding to highest fidelity for each optimization method.



(c) Values of the β parameter corresponding to highest fidelity for each optimization method. The cases in which the β parameter was not updated during the optimization are not reported.

Figure 3.29: The plots show the best parameter values for the different optimization methods tested.

Two directions for future improvement emerge from this work. First, the RB evaluation process

Method	RB Value	Iterations	RB Evaluations	Duration [s]
CMA-ES	0.003346	30	210	4 536
Optuna 30 steps	0.018178	30	30	763
Optuna 100 steps	0.042908	100	100	2 257
Optuna 1000 steps	0.003424	1000	1000	21 811
Optuna no β	0.137907	1000	1000	21 820

Table 3.2: This table provides a short summary of the results obtained using the `CMA-ES` and `Optuna` algorithm.
In particular, regarding the `Optuna` optimization algorithm the number of optimization iterations and of Randomized Benchmarking optimization is the same as the number of function evaluations coincides with the number of trials and no pruning was performed. The reported value of the objective function corresponds to the average Clifford gate infidelity obtained at the end of optimization via Randomized Benchmarking (1-fidelity), which serves as the cost function minimized by the algorithm. The *Duration* column refers to the total runtime of the script, including the initial fine-tuning stage preceding the optimization.

Analysis Name	Fidelity Best	Amplitude [a.u.]	Frequency Best [$\times 10^9$ Hz]	β Best
Nelder_Mead	0.99731	0.04063	4.95701	-1.53253
init_simplex_1	0.99533	0.04069	4.95820	-
init_simplex_2	0.99564	0.04131	4.95817	-
init_simplex_3	0.99554	0.04058	4.95819	-
SLSQP	0.99518	0.04058	4.95822	-0.00115
CMA-ES	0.99665	-0.16634	4.95822	0.08802
optuna_30	0.99583	0.28059	4.95601	2.75519
optuna_100	0.99645	0.29026	4.955279	-1.46504
optuna_1000	0.99847	-0.00051	4.961836	0.2222
optuna no β	0.99846	0.04104	4.957374	-

Table 3.3: The table reports the best fidelity, amplitude, frequency, and β from each optimization method allowing for an easy comparison.

itself could be optimized. In the present study, sequences of depth 1000 were used, and for each depth, 1000 Clifford circuits were sampled. This approach, while reliable, is computationally expensive. A detailed study, possibly involving hyperparameter optimization techniques, could help determine whether shorter sequences or fewer samples still yield meaningful fidelity estimates, potentially reducing total runtime and making on-chip calibration more feasible.

Second, the performance of the optimization algorithms themselves could be refined. For example, in the case of `CMA-ES`, adjusting the population size or reducing the initial value of σ could improve convergence, although at the cost of reduced parameter space exploration or increased computational load. These modifications may be particularly relevant in a more stable calibration regime, where parameter values show limited drift over time.

In conclusion, while there is still room for improvement in both the fidelity estimation routines and the design of the optimization algorithm, the results presented here demonstrate that automated, hardware-level gate fidelity optimization is indeed feasible and yields meaningful improvements. Despite the fact that many of the routines require over 30 minutes to complete, a clear reduction in the cost function in the average Clifford gate infidelity was consistently observed.

Importantly, in the absence of a dedicated procedure targeting the average Clifford gate fidelity, the only alternative would be to rely on manual tuning by an operator. This would involve repeatedly guessing suitable values for amplitude, frequency, and β DRAG parameter in order to improve performance, time-consuming and inefficient approach. As quantum processors continue to scale up in qubit count, manual calibration becomes increasingly impractical. These findings therefore support the development of automated optimization protocols as a necessary and scalable path forward for high-fidelity quantum control.

Analysis Name	**Initial cost**	**Best cost**	**Best index**	**Amplitude best [a.u.]**	**Frequency best [$\times 10^9$ Hz]**	**β best**	**Best cost improv. [%]**	**Initial fidelity**	**Best fidelity**	**Best fidelity improv[%]**
Nelder_Mead	0.00983	0.00269	19	0.04063	4.95701	-1.53253	72.59	0.99017	0.99731	0.72
init_simplex_1	0.00633	0.00467	25	0.04069	4.95820	-	26.23	0.99367	0.99533	0.17
init_simplex_2	0.00607	0.00436	17	0.04131	4.95817	-	28.27	0.99393	0.99564	0.17
init_simplex_3	0.00585	0.00446	37	0.04058	4.95819	-	23.73	0.99415	0.99554	0.14
SLSQP	0.02172	0.00482	13	0.04058	4.95822	-0.00115	77.79	0.97828	0.99518	1.73
CMA-ES	0.00728	0.00335	9	-0.16634	4.95822	0.08802	54.04	0.99272	0.99665	0.40
optuna_30	0.016471	0.00416	23	0.28059	4.95601	2.75519	74.70	0.98352	0.99583	1.25
optuna_100	0.017535	0.00354	13	0.29026	4.955279	-1.46504	79.77	0.98246	0.99645	1.42
optuna_1000	0.005307	0.00152	405	-0.00051	4.961836	0.2222	71.18	0.99469	0.99847	0.37
optuna no β	0.006983	0.00153	865	0.04104	4.957374	-	77.96	0.99301	0.99846	0.54

Table 3.4: This table reports the results obtained from the optimization of the average Clifford gate fidelity as a function of the pulse frequency, amplitude, and the parameter β, which scales the second quadrature component in the case of a DRAG pulse. Specifically, it shows the initial and optimal values of the cost function, as well as the corresponding parameter values at the optimum. The table also includes the initial fidelity, the percentage decrease in the cost function, and the percentage improvement in fidelity achieved after optimization.

Analysis Name	**Initial cost**	**Final cost**	**Amplitude final [a.u.]**	**Frequency final [$\times 10^9$ Hz]**	**β final**	**Final cost improv. [%]**	**Initial fidelity**	**Final fidelity**	**Final fidelity improv. [%]**
Nelder_Mead	0.00983	0.00295	0.04067	4.95699	-1.53976	70.02	0.99017	0.99705	0.69
init_simplex_1	0.00633	0.00517	0.04069	4.95820	-	18.41	0.99367	0.99483	0.12
init_simplex_2	0.00607	0.00437	0.04132	4.95817	-	28.10	0.99393	0.99563	0.17
init_simplex_3	0.00585	0.00512	0.04058	4.95819	-	12.46	0.99415	0.99488	0.07
SLSQP	0.02172	0.00507	0.04058	4.95822	-0.00115	76.64	0.97828	0.99493	1.70
CMA-ES	0.00728	0.003346	0.12956	4.95822	0.18364	-265.93	0.99272	0.97336	-1.95
optuna_30	0.016471	0.018178	0.46076	4.95479	2.92767	-10.36	0.98352	0.98182	-0.17
optuna_100	0.017535	0.042908	0.386175	4.95469	-1.31692	-144.68	0.98246	0.95709	-2.58
optuna_1000	0.005307	0.003424	-0.042718	4.95794	0.0626649	35.48	0.99469	0.99657	0.18
optuna no β	0.006983	0.137907	0.058214	4.95818	-	-1874.74	0.99301	0.86209	-13.18

Table 3.5: This table reports the results obtained from the optimization of the average Clifford gate fidelity as a function of the pulse frequency, amplitude, and the parameter β, which scales the second quadrature component in the case of a DRAG pulse. Specifically, it shows the initial and final values of the cost function, as well as the corresponding parameter values at the final step. The table also includes the initial fidelity, the percentage decrease in the cost function, and the percentage improvement in fidelity achieved at the end of the optimization process.

Chapter 4

Pulses analysis and tuning

Having concluded that the closed-loop optimization protocol we tested would not significantly improve our circuit performance, we shifted our focus towards the improvement and implementation of individual protocols to increase the accuracy of qubit operations.
In this chapter, we present the results of two additions to the `Qibocal` software. The first is the inclusion of an $RX90$ gate as a native gate, which can enhance the performance of protocols requiring qubit rotations of $\frac{\pi}{2}$. The second is the implementation of the cryoscope, a routine first described in [1], which is useful for correcting distortions in the magnetic flux pulse applied to the SQUID.

4.1 RX90 calibration

As discussed in Section (2.2), it is possible to calibrate system parameters and perform fine-tuning routines to accurately determine the amplitude and frequency of the drive pulse required to transition the qubit from the $|0\rangle$ state to the $|1\rangle$ state, which corresponds to the $R_X(\pi)$ rotation on the Bloch sphere.

However, executing more complex quantum circuits and algorithms requires the ability to perform a broader set of quantum operations. In gate-based quantum computing, and in particular for superconducting hardware platforms, this is achieved by composing more general unitary operations from a limited set of pre-defined, hardware-native quantum gates. These native gates, or simply natives, are elementary operations that are directly implementable and physically calibrated on the quantum processor.

The choice and quality of these native gates are extremely important as all higher-level gates will be decomposed into sequences of natives, and any calibration errors in the latter can accumulate and propagate through a circuit, degrading the overall fidelity.

4.1.1 `Qibolab` native gates

Native gates are those gates that can be directly implemented at the hardware level, in contrast to abstract logical gates, which must be transpiled into sequences of hardware-supported primitives. In the case of `Qibolab` [1], the native gate set consists of $R_X(\pi)$ gate, calibrated by performing the Rabi experiments described in Section (2.2.3), the measurement gate and the virtual-Z (VZ) gate which is not implemented via a physical pulse but rather through a dynamic adjustment of the phase of subsequent control pulses. This native gate set is sufficient for universal single-qubit control.

Indeed, it can be shown that any arbitrary single-qubit unitary operation $U(\gamma, \theta, \phi) \in SU(2)$ can be expressed, up to a global phase, as a sequence of rotations around the z and x axes of the Bloch sphere:

$$U(\gamma, \theta, \phi) = R_Z(\gamma) R_X(\theta) R_Z(\phi). \tag{4.1}$$

This result, known as Euler's decomposition, ensures that all single-qubit operations can be realized using a combination of R_X and R_Z rotations. Note that, in practice, this final R_Z rotation does not need to be realized as a physical pulse. Instead, it can be implemented virtually by

[1] At least for version 0.1 of `Qibolab`

adjusting the phase reference of subsequent pulses, a technique known as the virtual-Z gate [52].

4.1.2 Differences in pi-half pulse calibration

Since many routines and protocols in quantum computing rely on $R_X(\pi/2)$ rotations, we decided to explore the possibility of including the $R_X(\pi/2)$ gate in the native set of `Qibolab` . The main motivation for this addition is that, until now, we had assumed the amplitude or duration of the $R_X(\pi/2)$ gate was exactly half that of the $R_X(\pi)$ pulse. This assumption implies a perfectly linear response of the physical system of the qubit to the drive pulse; an idealization that may not be satisfied in practice due to nonlinearities in the system or imperfections in the pulse generation and transmission.

For each of the calibration experiments concerning the rotation pulses, we provide examples of plots to show how the relevant physical quantities vary depending on whether we are calibrating the $R_X(\pi)$ or the $R_X(\pi/2)$ gate.
In addition to the plots, we provide a summary table where we report the measured amplitude (or duration) of the microwave pulse, along with the associated uncertainty. In each table, we include also the standardized difference between the amplitude (or duration) values of the $RX90$ pulse obtained with different methods. The standardized difference is defined as

$$Z = \frac{v_{RX/2} - v_{RX90}}{\sqrt{\sigma^2_{RX90} + \sigma^2_{RX/2}}}, \tag{4.2}$$

where $v_{RX/2}$ is the value of the physical quantity measured by halving the value corresponding to the $R_X(\pi)$ pulse, v_{RX90} is the value of the physical quantity obtained from a direct calibration of the $R_X(\pi/2)$ gate. $\sigma^2_{RX/2}$ and σ^2_{RX90} are the respective measurement errors. The standardized difference provides a normalized measure of the discrepancy between the two values by incorporating their respective uncertainties. A small standardized difference indicates strong agreement within experimental error, while larger values suggest meaningful deviations that may point to systematic effects or non-linearities in the system response.

Rabi amplitude

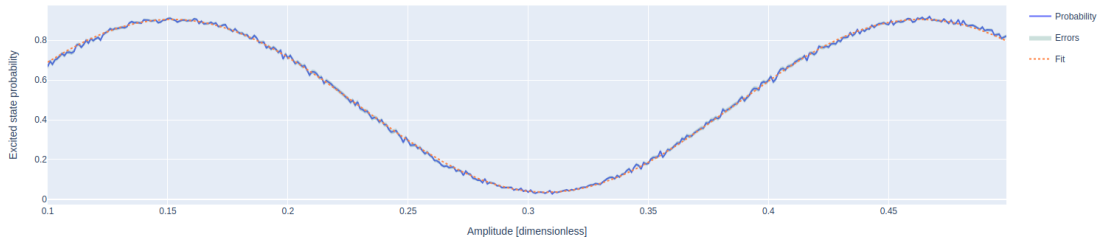
Table (4.1) reports the results of the Rabi experiments performed to calibrate the amplitude of the pulses associated with the $R_X(\pi)$ and $R_X(\pi/2)$ gates. For the $R_X(\pi/2)$ gate, both the amplitude derived by halving the result of the $R_X(\pi)$ calibration and that obtained from a direct calibration are presented for comparison.
The measurements were conducted sequentially: first all the $R_X(\pi)$ gates were calibrated, followed immediately by the $R_X(\pi/2)$ gates. Apart from the `rx90` boolean parameter, which allows the user to select which gate to calibrate, all other user-configurable parameters were kept constant between experiments

An initial observation from Table (4.1) is that the standardized difference is always positive. This follows directly from its definition in Equation (4.2): the amplitude derived by halving the calibrated $R_X(\pi)$ amplitude is, in all cases, larger than the value obtained through direct calibration of the $R_X(\pi/2)$ gate. Moreover, the magnitude of the standardized difference is between $10 - 10^2$, indicating a significant deviation between the two calibration approaches. This suggests that, despite the theoretical expectation of linear scaling between the pulse amplitudes of the $R_X(\pi)$ and $R_X(\pi/2)$ gates, in practice, the system exhibits non-ideal behavior likely stemming from hardware nonlinearities or limitations in the pulse shaping mechanism.

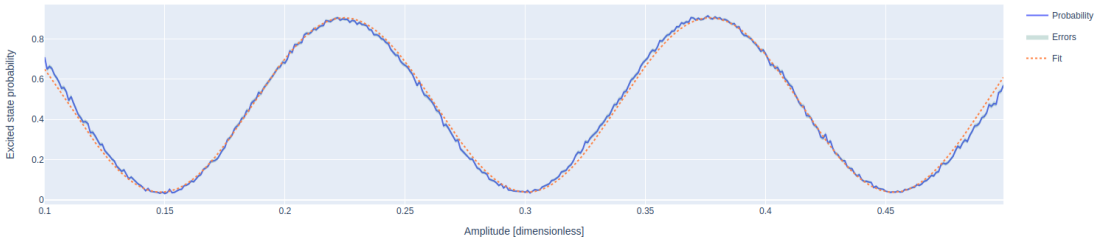
Rabi length

The pulse duration measurements for the $R_X(\pi)$ and $R_X(\pi/2)$ gates were carried out on the Soprano-D chip by QuantWare [53], when necessary we will refer to this chip as `qw5q`. The reason for this change is that these measurements were performed after the scheduled cryostat maintenance, and at the time `qw11q` was not yet calibrated. The `qw5q` chip is a five-qubit chip denoted as `0`, `1`, `2`, `3` and `4`.

The sequence of measurements mirrored that used for the amplitude calibration: $R_X(\pi)$ was calibrated first, followed by $R_X(\pi/2)$, with all experimental settings kept fixed except for the `rx90`



(a) Output of the `rabi_amplitude` protocol to calibrate the $R_X(\pi)$ gate for qubit `B4`.



(b) Output of the `rabi_amplitude` protocol to calibrate the $R_X(\pi/2)$ gate for qubit `B4`.

Figure 4.1: Comparison between the results of the amplitude calibration for the $R_X(\pi)$ and $R_X(\pi/2)$ gates.

flag. An example of the output of the calibration protocol for pulse duration, for both the $R_X(\pi)$ and $R_X(\pi/2)$ gates, is shown in Figures (4.2a) and (4.2b).

A summary of the quantitative results obtained from these calibrations is presented in Table (4.2). Again, both the inferred value (obtained by halving the $R_X(\pi)$ duration) and the directly calibrated value for the $R_X(\pi/2)$ gate are shown.
As in the case of the amplitude calibrations, the standardized difference values are generally positive, indicating that the duration obtained by halving the calibrated $R_X(\pi)$ value tends to be higher than the value obtained from the direct $R_X(\pi/2)$ calibration. The only exception to this trend occurs for qubit `5`, for which the standardized difference is negative. However, unlike the amplitude results discussed previously, the absolute values of the standardized difference are closer to unity or below, suggesting that the discrepancy between the two methods is less pronounced. This reduction in $|Z|$ is due to two factors: first the smaller absolute difference between the duration values obtained via direct and indirect calibration, and second the larger relative uncertainties associated with the duration estimates compared to those for amplitude.

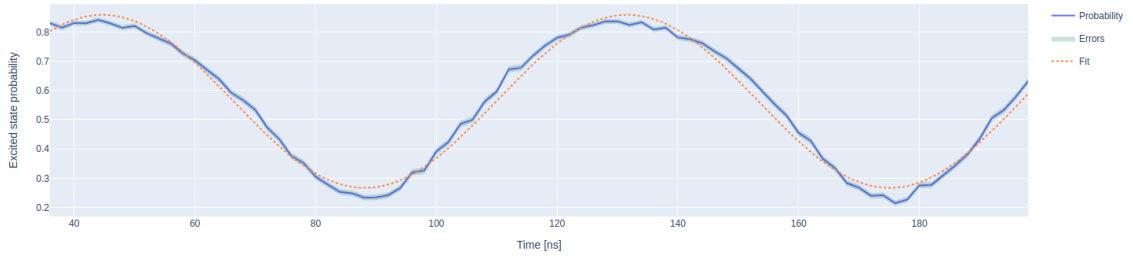
Although, these observations indicate a more consistent agreement between direct and indirect duration calibrations, non-negligible deviations still exist and may need further investigation, particularly in the context of high-fidelity gate execution.

Rabi amplitude frequency

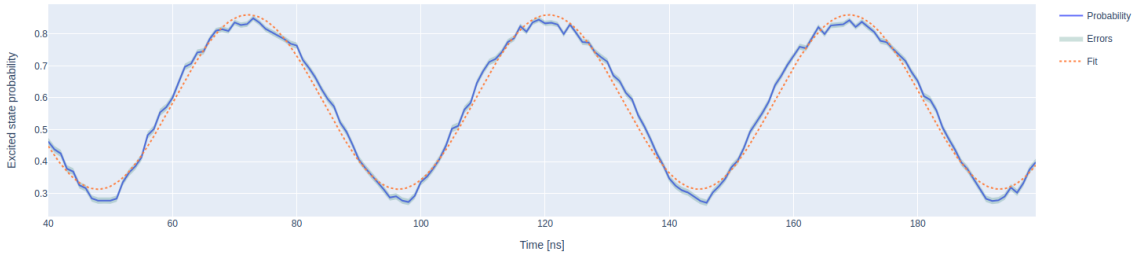
In Figure (4.3) we show the results of the Rabi amplitude-frequency calibration experiments, which aim to determine the correct amplitude and frequency for either the $R_X(\pi)$ or $R_X(\pi/2)$ gate, with a fixed pulse duration of 40 ns. As in previous cases, the scan ranges for amplitude and frequency were held constant between the two experiments; the only difference was the target gate for calibration.

Typically, in such experiments, one observes a single peak corresponding to the maximum probability of the qubit being in the excited state $|1\rangle$. However, when we extend the scan ranges or, as in this case, we use the same scan intervals for a gate that ideally requires roughly half the amplitude, the resulting pattern deviates from the usual single-peak expectation. Instead, we observe a more complex structure, consistent with the model described by Equation (2.7). The ideal interference pattern for the system is shown in Figure (4.4). The amplitude and frequency values used for the plot in Figure (4.4) are not in scale with the real system.

The figures discussed here are primarily useful from a qualitative standpoint. They illustrate how, within the previously effective amplitude and frequency ranges, where a single interference

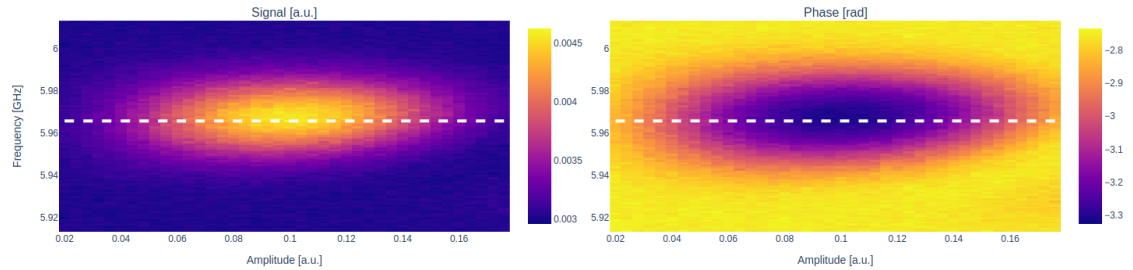


(a) Output of the `rabi_length` protocol to calibrate the $R_X(\pi)$ gate for qubit `3`.

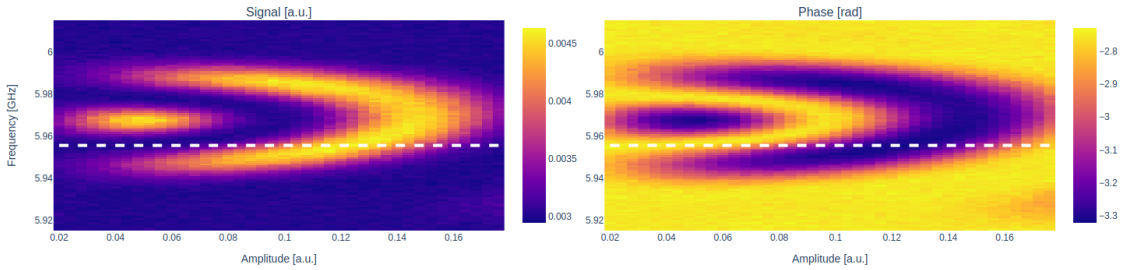


(b) Output of the `rabi_length` protocol to calibrate the $R_X(\pi/2)$ gate for qubit `3`.

Figure 4.2: Comparison between the results of the duration calibration for the $R_X(\pi)$ and $R_X(\pi/2)$ gates.



(a) Output of the `rabi_amplitude_frequency_signal` protocol to calibrate the $R_X(\pi)$ gate for qubit `B2`.



(b) Output of the `rabi_amplitude_frequency_signal` protocol to calibrate the $R_X(\pi/2)$ gate for qubit `B2`.

Figure 4.3: Comparison between the results of the amplitude and frequency calibration for the $R_X(\pi)$ and $R_X(\pi/2)$ gates.

peak was typically observed, we now observe a more complex interference pattern. However, with such an interference pattern, the standard peak detection algorithm fails to correctly identify the first peak, as we can understand from the misalignment of the dashed line in Figure (4.3b).

To allow for a quantitative comparison between the $R_X(\pi/2)$ amplitude obtained through direct calibration and the value inferred from the $R_X(\pi)$ calibration, additional measurements were carried out using modified scan ranges for amplitude and frequency. For the calibration of the $R_X(\pi)$ gate, the frequency was scanned over a 60 MHz range centered around the qubit frequency, with a step size of 1 MHz. The amplitude was scanned in the interval from 0.15 to 0.35 with a step of 0.01.

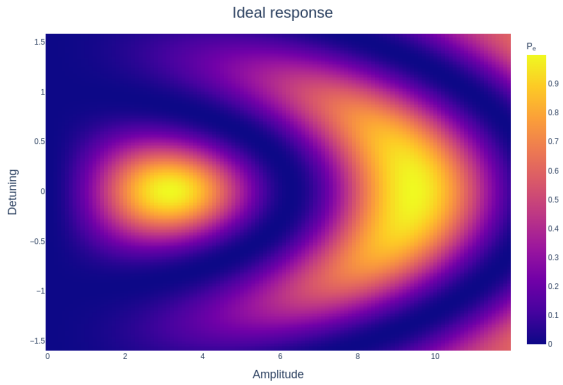
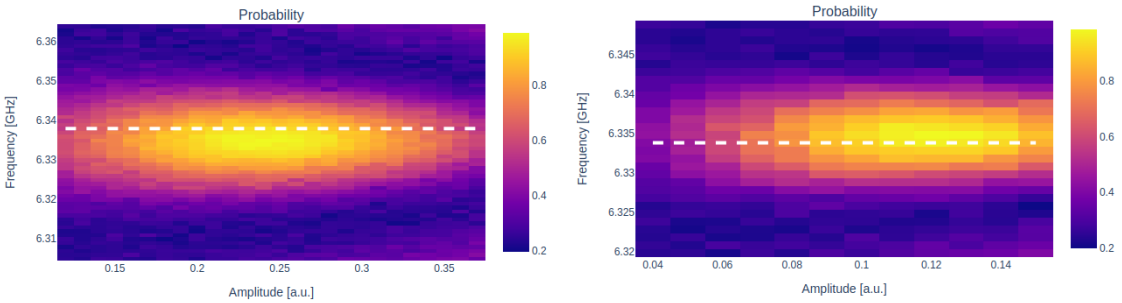


Figure 4.4: Ideal Rabi oscillation pattern for single qubit varying amplitude and frequency.

For the calibration of the $R_X(\pi/)$ gate, the frequency was scanned over a 30 MHz range centered around the qubit frequency, with a step size of 1 MHz. The amplitude was scanned in the interval from 0.04 to 0.16 using a step of 0.01.
The results of these measurements are summarized in Table (4.3), and an example of the corresponding interference patterns is shown in Figure (4.5).

As in the previous cases, the standardized difference values are consistently positive, reflecting that the amplitude obtained by halving the value calibrated for the $R_X(\pi)$ gate is always, even though slightly, greater than the one obtained through a direct calibration of the $R_X(\pi/2)$ gate. However, n Table (4.3) it is worth noting that, unlike the results from the other calibration protocols, the magnitude of the standardized difference in this case is consistently less than one. This indicates a reasonable agreement between the two estimation methods when experimental uncertainty is taken into account.

It is important to emphasize, however, that this agreement is largely a consequence of the large measurement uncertainty associated with the amplitude values (the average relative error is of the order of 80%).



(a) Output of the `rabi_amplitude_frequency` protocol to calibrate the $R_X(\pi)$ gate for qubit `3`.

(b) Output of the `rabi_amplitude_frequency` protocol to calibrate the $R_X(\pi/2)$ gate for qubit `3`.

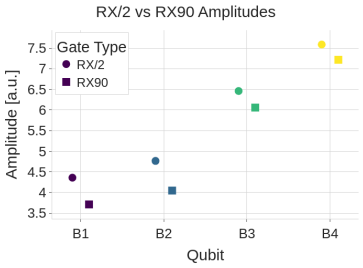
Figure 4.5: Comparison between the results of the amplitude and frequency calibration for the $R_X(\pi)$ and $R_X(\pi/2)$ gates.

In Figure (4.6) we show three plots, each comparing the pulse parameters obtained for implementing an $R_X(\pi/2)$ gate using different calibration strategies, across the various qubits for which the $RX90$ calibration was performed.

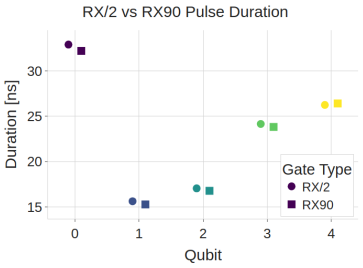
An interesting observation, previously mentioned but made more evident when examining the plots, is that both the amplitude and duration estimates for the $R_X(\pi/2)$ gate when obtained by halving the corresponding values calibrated for the $R_X(\pi)$ gate, tend to be slightly overestimated compared to those obtained via direct calibration of the $R_X(\pi/2)$ gate. This discrepancy is particularly evident for measurements conducted on the `qw11q` chip, as also confirmed by the values of the standardized difference reported in Table (4.1). In contrast, for the `qw5q` chip, the disagreement is noticeably smaller; in particular he standardized difference for amplitude values obtained via the `rabi_amplitude_frequency` routine remains below unity in absolute value. Nevertheless, it is important to emphasize that these latter amplitude values are also associated with larger relative

uncertainties and were obtained using a routine that is not the standard one for precise amplitude calibration, but rather intended to provide a rough estimate of the parameter region to explore.

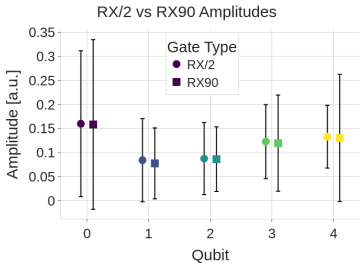
Overall, these results suggest that when circuit execution involves native $R_X(\pi/2)$ gates it may be advantageous to perform a dedicated calibration of the $R_X(\pi/2)$ gate, rather than relying solely on indirect estimates derived from $R_X(\pi)$ gate calibrations. This is especially true in scenarios where high gate fidelity is critical or when hardware-specific nonlinearities cannot be neglected.



(a) Amplitude of the pulse for the $R_X(\pi/2)$ for different qubits obtained by halving the $R_X(\pi)$ amplitude (circle) or by calibrating the $R_X(\pi/2)$ gate with the `rabi_amplitude` protocol.



(b) Duration of the pulse for the $R_X(\pi/2)$ for different qubits obtained by halving the $R_X(\pi)$ amplitude (circle) or by calibrating the $R_X(\pi/2)$ gate with the `rabi_length` protocol.



(c) Amplitude of the pulse for the $R_X(\pi/2)$ for different qubits obtained by halving the $R_X(\pi)$ amplitude (circle) or by calibrating the $R_X(\pi/2)$ gate with the `rabi_amplitude_frequency` protocol.

Figure 4.6: The plots show the amplitude (duration) values of the $R_X\pi/2$ gate for different qubits obtained from direct calibration or from the calibration of the $R_X(\pi)$ gate.

4.1.3 RX90 as native gate

Considering both these results and the fact that we wanted to avoid introducing breaking changes to the library, we did not replace the $R_X(\pi)$ gate with the $R_X(\pi/2)$ in the natives set. Instead, we opted to add the $R_X(\pi/2)$ gate to the set of native gates in `Qibolab` (see (4.2)). To support this change we updated the `native.py` module, which provides the data containers for holding the pulse parameters required for implementing every native gate.

Additionally, we modified some of the calibration experiments of the `Qibocal` library presented in Section (2.2) to support the calibration of this new gate. In particular, we adapted the code used for the various implementations of the Rabi oscillation measurement protocol to include the $R_X(\pi/2)$ gate.

Moreover, we also modified the function that constructs single-qubit rotations so that if the `rx90` flag is set to `True`, meaning the $RX90$ gate has been calibrated, the rotations are built starting from an $R_X(\pi/2)$ pulse (see refsnippet:rotation.adapt). In this case, the rotation angle is implemented by adjusting the pulse amplitude accordingly, and if the required amplitude exceeds 0.5, two consecutive pulses are applied to achieve the desired rotation.

Listing 4.1: Arbitrary qubit rotation implementation in `Qibolab` .

```
def rotation(
    seq: PulseSequence,
    theta: float = np.pi,
    phi: float = 0.0,
    rx90: bool = False,
) -> PulseSequence:
    """Create a sequence for single-qubit rotation.

    ‘‘theta‘‘ will be the angle of the rotation, while ‘‘phi‘‘ the angle that the
        rotation axis forms with x axis.
    If ‘‘rx90‘‘ is True the rotation will be performed starting from an RX90 pulse
        doubling its amplitude,
    if the amplitude is greater than 0.5 two pulses are played.
    """
```

```
    theta, phi = _normalize_angles(theta, phi)
    ch, pulse = seq[0]
    assert isinstance(pulse, Pulse)
    amplitude = pulse.amplitude * theta / np.pi
    if rx90:
        if amplitude > 0.5:
            p = replace(pulse, amplitude=amplitude, relative_phase=phi)
            return PulseSequence([(ch, p), (ch, p.model_copy())])
        else:
            return PulseSequence(
                [(ch, replace(pulse, amplitude=2 * amplitude, relative_phase=phi))]
            )
    return PulseSequence(
        [(ch, replace(pulse, amplitude=amplitude, relative_phase=phi))]
    )
```

Listing 4.2: Updated natives class in `Qibolab` .

```
class SingleQubitNatives(NativeContainer):
    """Container with the native single-qubit gates acting on a specific
    qubit."""

    RX: Optional[Native] = None
    """Pulse to drive the qubit from state 0 to state 1."""
    RX90: Optional[Native] = None
    """Pulse to drive the qubit from state 0 to state +."""
    RX12: Optional[Native] = None
    """Pulse to drive to qubit from state 1 to state 2."""
    MZ: Optional[Native] = None
    """Measurement pulse."""
    CP: Optional[Native] = None
    """Pulse to activate coupler."""

    def R(self, theta: float = np.pi, phi: float = 0.0) -> PulseSequence:
        """Create a sequence for single-qubit rotation.

        ``theta`` will be the angle of the rotation, while ``phi`` the angle that the
            rotation axis forms with x axis.
        """
        if self.RX90 is not None:
            return rotation(self.RX90, theta, phi, rx90=True)
        assert self.RX is not None
        return rotation(self.RX, theta, phi)
```

Qubit	RX		RX / 2		RX90		Standardized Difference
	Amplitude 10^{-2}[a.u.]	Errors 10^{-2}[a.u.]	Amplitude 10^{-2}[a.u.]	Errors 10^{-2}[a.u.]	Amplitude 10^{-2}[a.u.]	Errors 10^{-2}[a.u.]	
B1	8.72	0.001	4.36	0.005	3.711	0.006	83.095
B2	9.433	0.008	4.765	0.004	4.047	0.004	126.926
B3	12.92	0.1	6.46	0.003	6.058	0.005	68.942
B4	15.17	0.1	7.585	0.005	7.216	0.005	52.184

Table 4.1: Amplitude values for the $R_X(\pi)$, half $R_X(\pi)$ and $R_X(\pi/2)$ pulses as measured with the `rabi_amplitude` protocol with a fixed duration of 40 ns.

Qubit	RX		RX / 2		RX90		Standardized Difference
	Duration [ns]	Errors [ns]	Duration [ns]	Errors [ns]	Duration [ns]	Duration [ns]	
0	65.8	0.5	32.9	0.25	32.2	0.1	2.599
1	31.26	0.09	15.63	0.045	15.28	0.02	7.107
2	32.1	0.07	17.05	0.035	16.78	0.02	6.698
3	48.3	0.2	24.15	0.1	23.82	0.05	2.952
4	52.5	0.3	26.25	0.1	26.41	0.06	-1.372

Table 4.2: Duration values for the $R_X(\pi)$, half $R_X(\pi)$ and $R_X(\pi/2)$ pulses as measured with the `rabi_length` protocol with a fixed amplitude of 0.2.

Qubit	RX		RX / 2		RX90		Standardized Difference
	Amplitude [a.u.]	Errors [a.u.]	Amplitude [a.u.]	Errors [a.u.]	Amplitude [a.u.]	Errors [a.u.]	
0	0.3202	0.3028	0.1601	0.1514	0.1585	0.1763	0.007
1	0.1683	0.1725	0.08445	0.08625	0.0777	0.0737	0.059
2	0.1752	0.1438	0.0876	0.0749	0.0865	0.0671	0.011
3	0.2457	0.1536	0.12285	0.0768	0.1197	0.0998	0.025
4	0.2662	0.1305	0.1331	0.06525	0.1305	0.1321	0.017

Table 4.3: Amplitude values for the $R_X(\pi)$, half $R_X(\pi)$ and $R_X(\pi/2)$ pulses as measured with the `rabi_amplitude_frequency` protocol with a fixed duration of 40 ns.

4.2 Flux pulse correction

As explained in section (1.3), accurate dynamical control of qubit frequency is of key importance to realize single- and two-qubit gates. One of the on-chip control variables that is used on the QunatumWare chip is the magnetic flux through a SQUID loop, the signal for magnetic flux control originates from an arbitrary waveform generator (AWG) which operates at room temperature. As the signal propagates through various electrical components along the control line leading to the chip it undergoes linear dynamical distortions. If not properly compensated, these distortions can degrade gate performance, impacting experiment fidelity and repeatability.

The protocol that we describe in this section was first introduced in [1], the goal is to reconstruct the magnetic flux signal that biases the SQUID and then determine predistortions that need to be applied so that the qubit receives the flux pulse as intended by the experimenter.

4.2.1 Preliminary analyses

In [1] is proposed a technique to characterize flux-pulse distortions induced by components inside the dilution refrigerator by directly measuring the qubit state. In this routine, we send the qubit a pulse sequence where a flux pulse, of varying duration τ, is embedded between two $\frac{\pi}{2}$ pulses which are always separated by a fixed interval T_{sep}.
The first $\frac{\pi}{2}$ pulse rotates the qubit of $\frac{\pi}{2}$ around the y axis of the Bloch sphere, changing its state from $|0\rangle$ to $\frac{|0\rangle+|1\rangle}{\sqrt{2}}$.

When a flux pulse $\Phi_{Q,\tau}(t)$ of duration τ is sent to the qubit[2] after the first $\frac{\pi}{2}$ pulse, the qubits evolve to the state $\frac{|0\rangle+e^{i\varphi_\tau}|1\rangle}{\sqrt{2}}$ with relative quantum phase

$$\frac{\varphi_\tau}{2\pi} = \int_0^{T_{sep}} \Delta f_Q(\Phi_{Q,\tau(t)})\mathrm{d}t = \int_0^{\tau} \Delta f_Q(\Phi_{Q,\tau(t)})\mathrm{d}t + \int_\tau^{T_{sep}} \Delta f_Q(\Phi_{Q,\tau(t)})\mathrm{d}t; \tag{4.3}$$

where in the second step we separated the contributions from flux response up to τ and the turn-off transient [3].

The experiment is then completed with a $\frac{\pi}{2}$ rotation around the y- or x-axis of the Bloch sphere to measure respectively the $\langle X\rangle$ or $\langle Y\rangle$ components of the Bloch vector when applying the measurement gate MZ. From the measurement of $\langle X\rangle$ and $\langle Y\rangle$ we can extract the relative phase φ_τ. For detailed calculations on how $\langle X\rangle$ and $\langle Y\rangle$ are measured and φ_τ extracted see Appendix (C).

Then we can estimate $\Phi_Q(t)$ in the interval $[\tau, \tau+\Delta\tau]$ with the steps described in the following. From the measurement of $\varphi_{\tau+\Delta\tau}$ and φ_τ we can compute $\overline{\Delta f_R}$:

$$\overline{\Delta f_R} = \frac{\varphi_{\tau+\Delta\tau} - \varphi_\tau}{2\pi\Delta\tau} \tag{4.4}$$

$$= \frac{1}{\Delta\tau}\left(\int_0^{\tau+\Delta\tau} \Delta f_Q(\Phi_{Q,\tau+\Delta\tau}(t))dt + \int_{\tau+\Delta\tau}^{T_{sep}} \Delta f_Q(\Phi_{Q,\tau+\Delta\tau}(t))dt\right) \tag{4.5}$$

$$- \frac{1}{\Delta\tau}\left(\int_0^{\tau} \Delta f_Q(\Phi_{Q,\tau}(t))dt - \int_\tau^{T_{sep}} \Delta f_Q(\Phi_{Q,\tau}(t))dt\right) \tag{4.6}$$

$$= \frac{1}{\Delta\tau}\left(\int_\tau^{\tau+\Delta\tau} \Delta f_Q(\Phi_{Q,\tau+\Delta\tau})dt + \int_{\tau+\Delta\tau}^{T_{sep}} \Delta f_Q(\Phi_{Q,\tau+\Delta\tau}(t))dt - \int_\tau^{T_{sep}} \Delta f_Q(\Phi_{Q,\tau}(t))dt\right) \tag{4.7}$$

$$= \frac{1}{\Delta\tau}\int_\tau^{\tau+\Delta\tau} \Delta f_Q(\Phi_{Q,\tau+\Delta\tau})dt + \varepsilon \tag{4.8}$$

with

$$\varepsilon = \frac{1}{\Delta\tau}\left(\int_{\tau+\Delta\tau}^{T_{sep}} \Delta f_Q(\Phi_{Q,\tau+\Delta\tau}(t))dt - \int_\tau^{T_{sep}} \Delta f_Q(\Phi_{Q,\tau}(t))dt\right).$$

[2] To send a $\Phi_{Q,\tau}(t)$ flux pulse we are actually sending a $V_{\mathrm{in},\tau}(t)$ DAC pulse
[3] In Appendix (B) we calculate the relative phase φ_τ for a more general shape of the flux pulse.

The phase contribution from the turn-off transients is minimal due to the sharp return to the first-order flux-insensitive sweet spot of the nearly quadratic $\Delta f_Q(\Phi_Q)$; numerical simulations suggest that $|\varepsilon|/\Delta f_R$ remains within the range of approximately 10^{-2} to 10^{-3} for typical dynamical distortions in commonly used electronic components [54] [55], for this reason it will be neglected.

At this point, we can obtain the reconstructed flux pulse $\Phi_R(t)$ inverting Equation (1.33).

In the following sections we will describe how we used this protocol to reconstruct the voltage-to-flux step response, and then how to determine and apply pre-distortion corrections to the control waveforms.

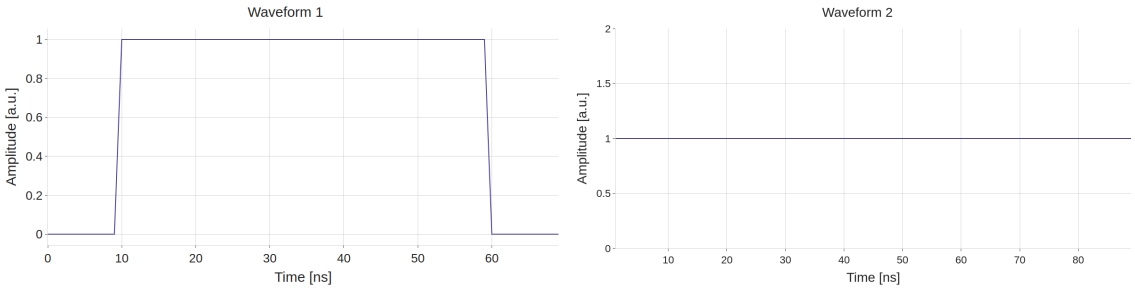
Experimental parameters

In our implementation, we fixed T_{sep} to 100 ns longer than the maximum duration of the flux pulse; this approach avoids the need for fine timing calibration, as mentioned also in Rol et al [1].

In the initial analysis conducted to develop the cryoscope routine for the flux pulse, we used a waveform defined as a combination of two step-functions. Specifically, the waveform remains at zero for the first 10 ns, rises to a nominal unit amplitude for the subsequent 50 ns, and returns to zero for the final 10 ns.
The flux pulse amplitude used for this first acquisition and the corresponding preliminary analyses was 0.1, which, as observed using the `flux_amplitude_frequency` routine, induces a detuning of approximately 0.01 GHz in qubit `D1`.

Given the relatively small detuning, we decided to perform an additional acquisition using a flux pulse with a higher amplitude to obtain more reliable results. For this second acquisition, a `flux_pulse_amplitude` of 0.5 was used, corresponding to a detuning of approximately 0.5 GHz. In this case, the waveform was also modified: a single step pulse of unit amplitude and 90 ns duration was applied, with no initial or final padding.

In the following, we will refer to the two waveforms as waveform_1 and waveform_2. The shapes of both waveforms are shown in Figure (4.7)



(a) Plot of the ideal flux pulse for waveform_1. (b) Plot of the ideal flux pulse for waveform_2.

Figure 4.7: Plot of the ideal waveform of the flux pulse.

The decision to perform tests with the second, longer waveform, omitting initial padding, was motivated by practical considerations related to the pulse reconstruction process. In particular, since the qubit's response is only affected during periods when a non-zero flux is applied, the system behavior before the pulse onset is less relevant for reconstruction purposes. Our primary focus is on characterizing the qubit's dynamics during the flux pulse and, potentially, in the transient period following its termination, to assess how long it takes for the qubit to return to its idle frequency after the pulse is switched off [4].

All experimental results and analyses presented herein are based on data acquired from qubit `D1` of chip `qw11q`, with OPX1000 by Quantum Machines [19] as acquisition device. The OPX1000 has a sampling rate of 1 GSample/s, corresponding to a temporal resolution of 1 ns which sets a lower limit $\Delta\tau \geq 1$ ns for the acquisition, this is also the $\Delta\tau$ value we used in all acquisitions.

[4] Although we did not focus on this aspect, this is relevant in practical applications, as it allows for shorter idle times between pulses and thus enables a higher number of gate operations within a given time window.

Phase reconstruction

The first step in analyzing the cryoscope data involves isolating the oscillatory components of the signals corresponding to the $\langle X \rangle$ and $\langle Y \rangle$ measurements. These oscillations reflect the coherent evolution of the qubit under the influence of the applied flux pulse.

What is experimentally measured for each observable is the probability $p_1(t)$ of finding the qubit in the excited state $|1\rangle$ as a function of time. From this probability, we reconstruct the temporal evolution of the z-component of the Bloch vector using the relation:

$$z(t) = 1 - p_1(t). \tag{4.9}$$

The reconstructed $z(t)$ signal is then fitted with an exponentially decaying sinusoid, analogous to the fitting procedure used in Ramsey experiments. This fitting is performed independently on the traces obtained from the $\langle X(t) \rangle$ and $\langle Y(t) \rangle$ measurements, allowing us to estimate the characteristic oscillation frequencies associated with each case.

At this point, to access the phase evolution, we reconstruct the complex signal $s(t) = \langle X(t) \rangle + i\langle Y(t) \rangle$. To do this, we use the following relations:

$$\langle X \rangle = 2p_1^X - 1 \tag{4.10}$$

$$\langle Y \rangle = 1 - 2p_1^Y \tag{4.11}$$

where $p_1^X(t)$ and $p_1^Y(t)$ denote the measured probabilities of finding the qubit in state $|1\rangle$ from the $\langle X \rangle$ and $\langle Y \rangle$ measurements, respectively. The derivations of these relations is reported in Appendix (C).

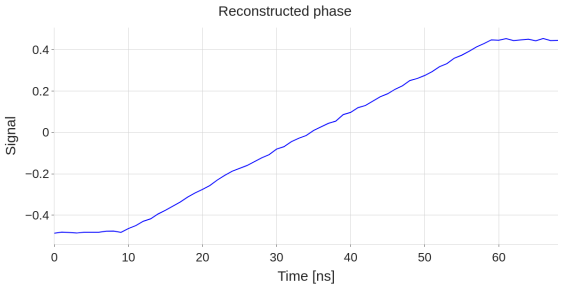
Once the characteristic oscillation frequency has been estimated, we demodulate the signal by this frequency: $s_{\text{demod}}(t) = e^{2\pi i f_{\text{demod}} t} \cdot s(t)$.
This operation shifts the signal's frequency content to near-zero removing the fast oscillations and leaving only the slower variations in phase.

This step is important because the raw signal may oscillate rapidly, making it difficult to directly extract the underlying phase dynamics. Demodulation transforms the signal into a smooth, slowly varying function of time, allowing for reliable computation of the phase and its derivative and, in turn, of the instantaneous detuning. Once demodulated, the time-dependent phase of the signal is extracted from the complex argument of the demodulated data. Without this preprocessing, the dominant carrier frequency could overshadow the relevant phase information.

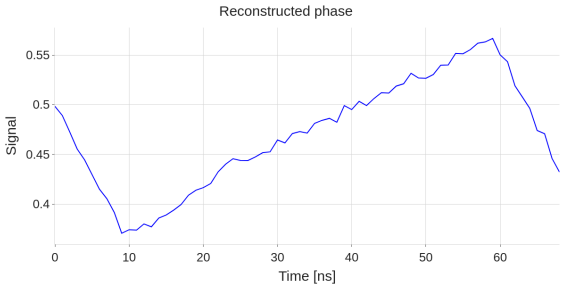
In the examples of phase reconstruction (4.8), looking in particular at Figure (4.8a), the first observation we can make is that the reconstructed qubit phase begins to vary immediately after $t = 10$ ns, while remaining essentially constant up to $t = 9$ ns. This seems to suggest that, at least within this time window, the flux pulse onset and the resulting qubit frequency shift happen quickly, with no evident latency. One possible explanation for the fast response observed at low flux amplitude $A = 0.1$, is that the flux pulse reaches its nominal value rapidly, making the expected exponential rise in frequency shift too fast to be resolved. Additionally, the small amplitude induces a weak detuning, leading to slow phase accumulation that may be more susceptible to background noise, potentially obscuring the signal.

A second observation is that the demodulated signal in Figure (4.8b) appears to exhibit increased phase oscillations compared to the non-demodulated case, potentially giving the impression that demodulation degrades the phase signal quality. However, a clearer picture emerges when considering a longer temporal evolution, as shown in Figures (4.8c) and (4.8d). In the non-demodulated case (Figure (4.8c)), the reconstructed phase signal is highly oscillatory, making it difficult to identify the qubit evolution. In contrast, the demodulated signal in Figure (4.8d) yields a much smoother phase trajectory, clearly revealing the onset and development of a frequency shift.

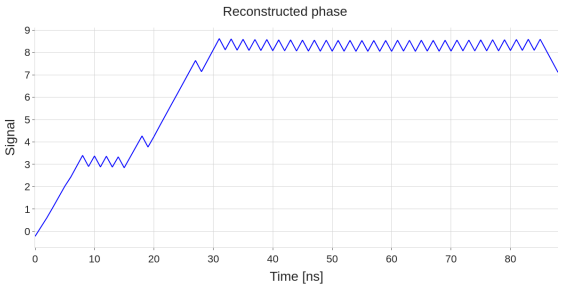
Notably, unlike the seemingly abrupt transition observed in the short time window (Figure (4.8b)), the long-timescale demodulated phase shows a gradual evolution, indicating that the flux pulse requires a finite rise time to reach its full amplitude. During this transient phase, the qubit frequency, and hence the rate of phase accumulation, increases progressively. This behavior is consistent with the physical expectation that the flux pulse ramps up smoothly, and it highlights the importance of demodulation in accurately recovering the continuous phase evolution of the qubit in response to the pulse.



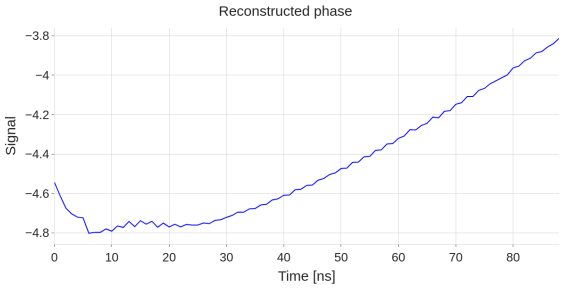
(a) Reconstruction of the phase evolution of the qubit under a flux pulse of shape waveform_1 with no demodulation applied on the acquired data.



(b) Reconstruction of the phase evolution of the qubit under a flux pulse of shape waveform_1 with demodulation applied on the acquired data.



(c) Reconstruction of the phase evolution of the qubit under a flux pulse of shape waveform_2 with no demodulation applied on the acquired data.



(d) Reconstruction of the phase evolution of the qubit under a flux pulse of shape waveform_1 with demodulation applied on the acquired data.

Figure 4.8: Examples of reconstruction of the phase evolution of a superconducting qubit under the action of a flux pulse with different waveforms.

Flux pulse reconstruction

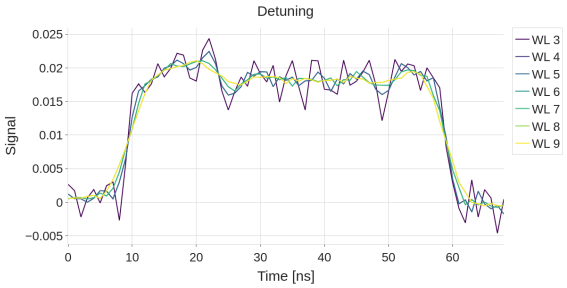
After reconstructing the phase signal from the demodulated complex data, we can estimate the instantaneous detuning of the qubit by computing the time derivative of this phase. As explained at the beginning of this Section, the rate at which the phase accumulates corresponds to the detuning (see Equation (4.4)), to obtain the detuning it's thus necessary to differentiate the phase signal.

Savitzky-Golay filter To perform this differentiation in a way that reduces the impact of noise, we used a Savitzky-Golay filter, which operates by fitting a polynomial to small segments of the data along the time axis. In this work, we used the `scipy.signal` implementation of the signal with the default settings where possible. In particular, the filter runs in `'interp'` mode, which doesn't pad or extend the signal at the edges; instead, it fits a polynomial to the last few data points and uses that to estimate the values near the boundaries.

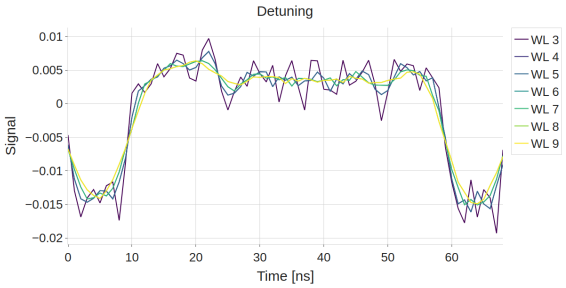
For our analysis, we use a polynomial order of 2 and explore how different values of `window_length` affect the final detuning signal. The choice of `window_length` is particularly important because the Savitzky-Golay filter effectively acts similarly to a moving average. This has some relevant consequences. First, if the window is too small, the filter does not average over enough points to effectively suppress high-frequency noise, making the derivative signal noisier. On the other hand, if the window is too large, it can over-smooth the signal and distort fast transitions. If the window is too large, the filter may incorporate data from the initial flat region into the fit, which can artificially stretch the signal's rising edge or introduce distortions at the boundaries. This can give the impression of a slower onset of the phase response than what is physically accurate, and such edge effects are particularly noticeable when the signal includes an initial segment of zero flux.

The effect of different `window_length` values on the reconstructed detuning is shown in Figure (4.9).

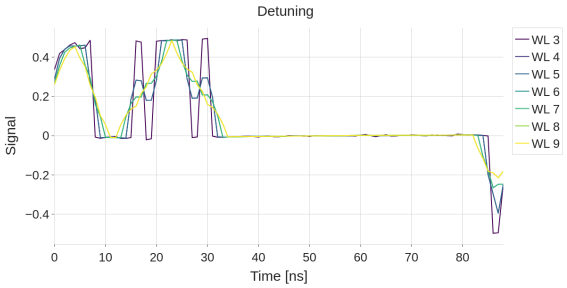
This tuning is especially important because the detuning is obtained by differentiating the phase signal, which amplifies any residual oscillations or imperfections. The issue becomes evident in the acquisition performed with `waveform_1` at an amplitude of 0.1, where the flux pulse is preceded by 10 ns of zero signal. In this case, two non-physical responses appear in the signal:



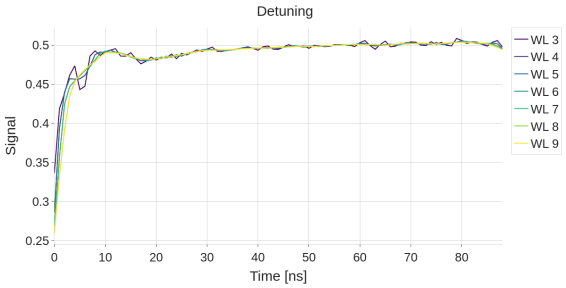
(a) Reconstruction of the detuning of the qubit under a flux pulse of shape waveform_1 without demodulation applied on the acquired data.



(b) Reconstruction of the detuning of the qubit under a flux pulse of shape waveform_1 with demodulation applied on the acquired data.



(c) Reconstruction of the detuning of the qubit under a flux pulse of shape waveform_2 without demodulation applied on the acquired data.



(d) Reconstruction of the detuning of the qubit under a flux pulse of shape waveform_2 with demodulation applied on the acquired data.

Figure 4.9: Examples of reconstruction of the detuning of a superconducting qubit under the action of a flux pulse with different waveforms.

first, the detuning appears to evolve even before the flux pulse is applied, giving the impression of a non-causal response; second, an exponential rise is observed in the reconstructed detuning that is not actually present in the qubit phase signal. These spurious effects are largely absent when the acquisition window is extended and the signal starts immediately with the pulse, but we nonetheless explored how different values of `window_length` affect the final result.

In particular, we tested values of `window_length` corresponding to a range between 3 and 10. The lower bound of 3 is set by the minimum required for a second-order polynomial fit, as the filter requires that `polyorder` must be less than `window_length`.

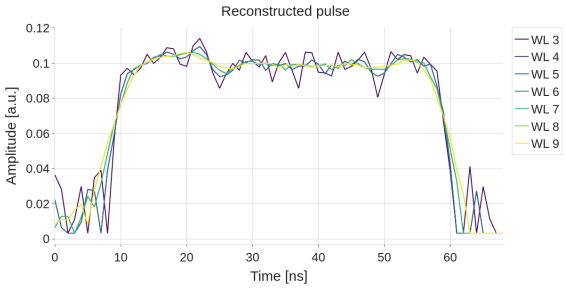
Figure (4.9) shows the reconstructed detuning under different conditions. In particular, panels (4.9a) and (4.9c) include results obtained without applying demodulation. For the first flux pulse, although demodulation introduces additional oscillations in the phase signal, it does not substantially affect the final detuning. The main discrepancy appears at the signal boundaries, where the ideally zero detuning, shows sharp, unphysical variations amplified by the effect of differentiation.

The situation is different for the second flux pulse. Without demodulation, the detuning reconstruction fails to reflect the expected physical behavior, as seen in Figure (4.9c). In contrast, when demodulation is applied, as shown in Figure (4.9d), the reconstructed detuning clearly follows the anticipated pattern: an exponential onset followed by a stable plateau, consistent with a qubit experiencing a constant-amplitude flux pulse.

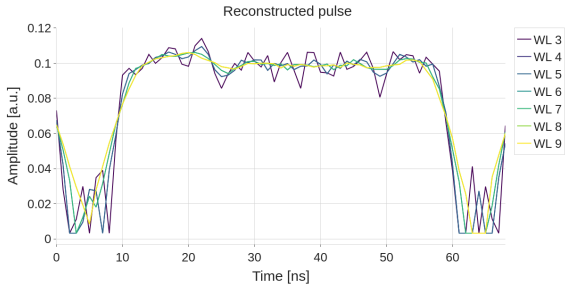
Finally, after reconstructing the qubit detuning, we invert the relationship between detuning and applied flux amplitude described by equation (1.33), to recover the actual flux amplitude as a function of time. Specifically, we assume a quadratic model of the form

$$f = c_1 A^2 + c_2 A + c_3, \tag{4.12}$$

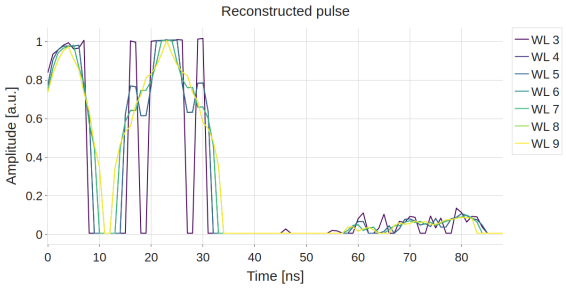
where f denotes the detuning, A is the applied flux amplitude, and c_1, c_2, and c_3 are coefficients obtained from the `flux_amplitude_frequency` routine (see Appendix (D) for more details). By inverting this relation, we reconstruct the time-dependent flux amplitude experienced by the qubit. The results of this reconstruction are shown in Figure (4.10). For completeness, we have included the plots related to the amplitude reconstruction performed for all previously discussed cases, both with and without demodulation.



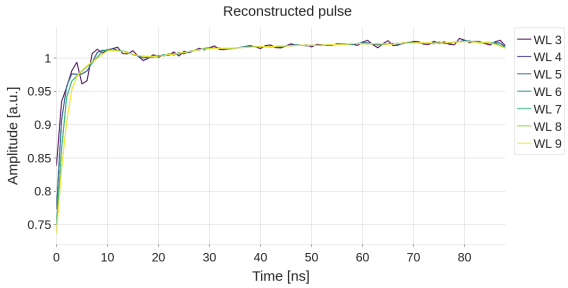
(a) Flux pulse reconstruction using the cryoscope technique.
The intended pulse has the shape of waveform_1 and no demodulation was originally applied.



(b) Flux pulse reconstruction using the cryoscope technique.
The intended pulse has the shape of waveform_1, demodulation was originally applied.



(c) Flux pulse reconstruction using the cryoscope technique.
The intended pulse has the shape of waveform_2 and no demodulation was originally applied.



(d) Flux pulse reconstruction using the cryoscope technique.
The intended pulse has the shape of waveform_2, demodulation was originally applied.

Figure 4.10: Examples of reconstruction of the flux amplitude.

Based on the results presented so far regarding the reconstruction of the flux pulse, we chose to adopt a default `window_length` corresponding to 7 points, similarly to what was done in [1].

IIR corrections

Our goal in implementing the Cryoscope protocol is to determine how to predistort the flux pulse in order to produce the desired detuning on the qubit. This predistortion is applied by filtering the waveform generated by the AWG using a combination of FIR and IIR digital filters. The goal of these filters is to compensate for the linear-dynamical distortions in the flux control line that would otherwise distort the shape of the pulse as it reaches the qubit.

Infinite Impulse Response (IIR) filters are digital filters characterized by the use of both past input samples and feedback from previous output values. The general form of an IIR filter in discrete time is given by the difference equation:

$$a_0 y[n] = \sum_{i=0}^{N} b_i x[n-i] - \sum_{i=1}^{M} a_i y[n-i], \tag{4.13}$$

where $x[n]$ is the input signal, $y[n]$ is the output signal, b_i are the feedforward coefficients, a_i are the feedback coefficients, N and M are the orders of the feedforward and feedback components, respectively.

This recursive structure allows IIR filters to approximate systems with exponential or resonant dynamics, which makes them especially suitable for correcting physical distortions like exponential overshoot or undershoot.

Indeed, the first step in our signal correction procedure is to characterize and compensate for the exponential overshoot or undershoot observed in the system's step response. These distortions can be interpreted as resulting from the system's impulse response $h(t)$ which when convolved with an ideal step input $V_{in}(t) = V_0 \cdot u(t)$, produces the measured output:

$$s(t) = (h * V_{in})(t), \tag{4.14}$$

where $u(t)$ is the Heaviside step function and the system is assumed to be causal ($h(t) = 0$ for $t < 0$).

In practice, we observe that the system's step response closely follows an exponential form, which we model as:

$$s(t) = g(1 - Ae^{-t/\tau}) \cdot u(t), \tag{4.15}$$

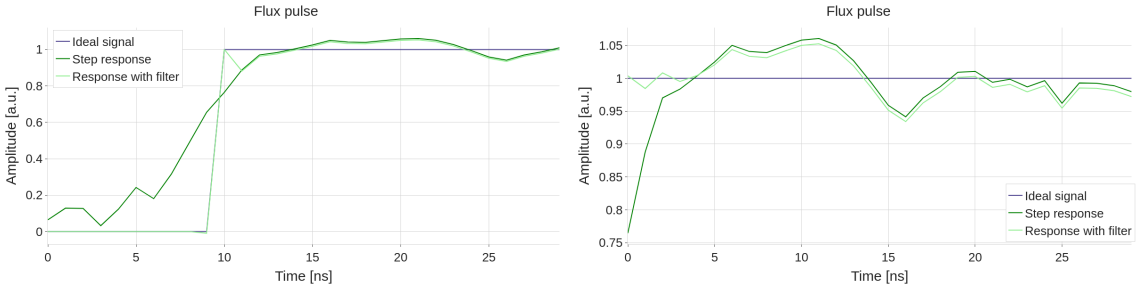
where A is the exponential amplitude characterizing the overshoot or undershoot, τ is the charachteristic time constant and g is a gain factor.

To compensate for this distortion, we construct a digital IIR filter designed to recover an ideal step-like response from the measured signal. This is done by modeling the observed step response with an inverted exponential correction function. The corrected signal is then defined as:

$$s_{\text{corr}}(t) = \frac{s(t)}{g(1 - Ae^{-t/\tau})}, \tag{4.16}$$

where $s(t)$ is the measured response. The parameters g, A, and τ are estimated by performing a least-squares minimization. Specifically, we minimize the squared difference between the corrected signal $s_{\text{corr}}(t)$ and the ideal target signal using `least_squares` method of the `scipy.optimize` library.

Figures (4.11) and (4.12) show the results of applying the exponential filter, obtained through the procedure described above, to different flux pulses.



(a) Comparison between the ideal, reconstructed response and corrected flux pulse for a step signal.

(b) Comparison between the ideal, reconstructed response, and corrected flux pulse for the step signal without the padding.

Figure 4.11: Comparisons between the ideal flux pulse (blue), the reconstructed flux pulse (dark green), flux pulse after the application of the correction (light green) for different pulse shapes. The exponential correction is applied as described in Equation .

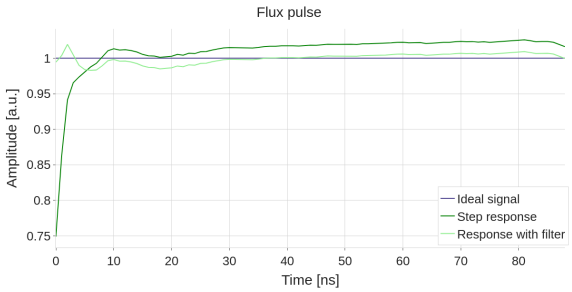


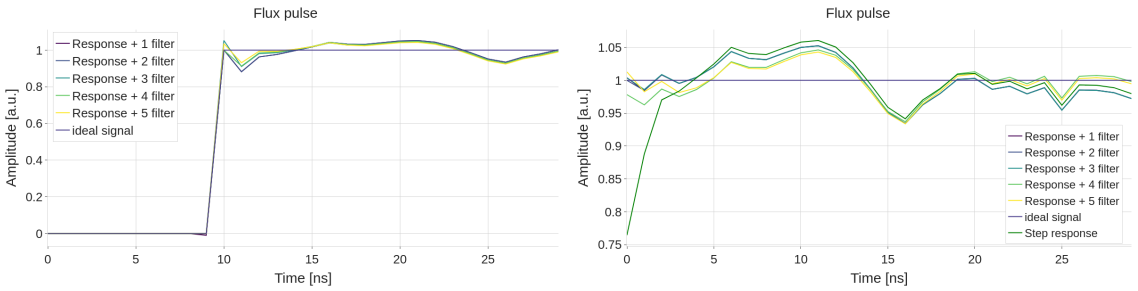
Figure 4.12: Comparison between the ideal flux pulse (blue), and corrected flux pulse for a rectangular pulse with no padding.

The first observation from these plots is that the IIR filter effectively compensates for the exponential rise observed in the step response. In particular, for the signals shown in Figure (4.11a), the Normalized Mean Square Error (NMSE) [5] decreases from 5.0185% to 0.1894% following the application of the correction. If we instead consider only the time interval during which the signal

[5]Normalized Mean Square Error (NMSE) is a figure of merit used to quantify the error between a processed signal $y[m, n]$ and a reference (ideal) signal $x[m, n]$. It extends the Mean Square Error (MSE), which measures the average squared difference between the signals, by normalizing it with the total power of the reference signal. This

is non-zero, as in Figure (4.11b), the NMSE decreases from 0.32% to 0.0902%.
In the acquisition performed using waveform_2, the signal exhibits significantly reduced distortion compared to the previous case. This results in an NMSE of 0.1282% for the uncorrected signal, which decreases to 0.0052% following the application of the exponential correction (see Figure (4.12)). However, particularly in Figure (4.12), it seems that the filter does not fully eliminate the overshoot. A possible explanation for this behavior is that the apparent exponential overshoot may originate from imperfect signal normalization, rather than from an actual residual distortion.

After verifying the effectiveness of a single exponential filter, we tried to use a combination of more exponentials to correct flux distortions occurring over time scales ranging from 30 to 200 ns, as was suggested in the original work by Rol et al. Based on this, we tested the application of multiple exponential corrections of the form given in Equation (4.11).



(a) Plot of the ideal, reconstructed, and corrected pulse with different numbers of exponential corrections for a step signal.

(b) Plot of the ideal, reconstructed, and corrected pulse with different numbers of exponential corrections for a step signal without the initial padding.

Figure 4.13: Comparisons between the ideal flux pulse (blue), the actual flux pulse (dark green), and the flux pulse after the application of a different number of exponential filters for different pulse shapes. The correction was applied iteratively, with each step using an exponential function of the form given in Equation ((4.11)).

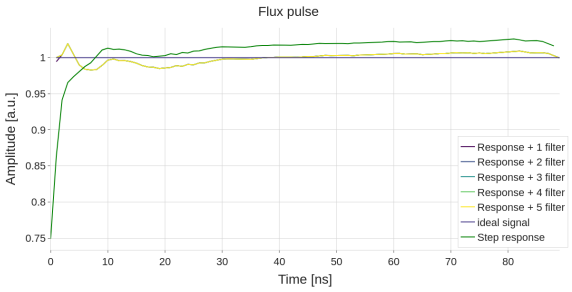


Figure 4.14: Plot of the ideal flux pulse (blue), reconstructed response (dark green), and corrected pulse with different numbers of exponential corrections for a rectangular pulse with no padding.

As shown in Figure (4.13), the application of successive exponential corrections yields a modest improvement in the reconstructed signal, particularly for the acquisition performed with waveform_1. In this case, the NMSE obtained at each iteration for the signal shown in Figure (4.13b) is, respectively, 0.0902%, 0.0901%, 0.0901%, 0.0663%, and 0.0596%. In contrast, for the second acquisition, the application of additional exponential filters does not produce a significant enhancement. After the first correction, the NMSE decreases from 0.1282% to 0.0052%, as previously noted, but remains essentially unchanged with further iterations. This difference is likely due to the distinct experimental conditions of the two acquisitions. In the first case, the reconstruction of the signal acquired with a `flux_pulse_amplitude` of 0.1 exhibits a more pronounced distortion, which the

normalization enables fair comparison across different datasets or signal amplitudes. The NMSE is defined as:

$$\text{NMSE} = \frac{\sum_{m=0}^{M-1}\sum_{n=0}^{N-1}|y[m,n]-x[m,n]|^2}{\sum_{m=0}^{M-1}\sum_{n=0}^{N-1}|x[m,n]|^2}$$

filters compensate for. On the other hand, in the second case, the reconstructed signal already appears much less distorted, reducing the benefit of using more than one exponential correction.

Before presenting the remaining results and continuing with the description of the procedure, it is useful to make the following observation. The tests on filter performance and the determination of corresponding taps using `waveform_1` primarily serve as a means to evaluate the effectiveness and limitations of the correction method. In this case, the exponential rise corrected by the filters does not reflect a physical property of the system, but rather an artifact introduced by the smoothing behavior of the Savitzky-Golay filter.

To obtain correction taps that address real physical distortions in the flux line, it is necessary to use acquisitions based on pulses without padding order to guarantee a more accurate reconstruction of the system response by avoiding non-physical effects.

Consequently, in the analyses that follow, we decided to focus exclusively on the non-zero portion of both flux signal. Even for the initial acquisitions that include a flat pre-pulse region, this segment has been excluded to ensure that the tests performed with the filters are more meaningful.

At this stage, the parameters for the exponential corrections have been determined. However, to apply these corrections in real-time, they must be discretized and implemented as digital filters within the qubit control electronics. Accordingly, the fitted parameters are used to calculate the coefficients of a first-order IIR filter as follows:

$$a_0 = 1, \tag{4.17}$$
$$a_1 = -(1-\alpha), \tag{4.18}$$
$$b_0 = 1 - k + k\alpha, \tag{4.19}$$
$$b_1 = -(1-k)(1-\alpha), \tag{4.20}$$

where

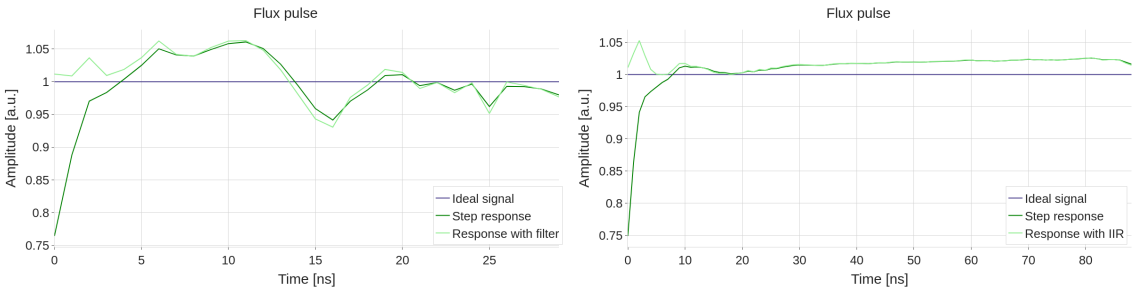
$$\alpha = 1 - \exp\left(-\frac{1}{f_s \cdot \tau(1+A)}\right), \tag{4.21}$$

and

$$k = \begin{cases} \frac{A}{(1+A)(1-\alpha)}, & \text{if } A < 0, \\ \frac{A}{1+A-\alpha}, & \text{if } A \geq 0, \end{cases} \tag{4.22}$$

with f_S being the sampling frequency of the digital hardware. Since this is an approximation of the exponential response, we expect the quality of the correction to be somewhat reduced compared to what is achieved by applying the continuous exponential filter.

To predict the effect of applying the real-time predistortion filters, we use the `lfilter` function from the `SciPy` library. The function takes three vectors as input: `lfilter(a,b,sig)` and applies the filter defined by the coefficient vectors `b` and `a` to the signal represented by the vector `sig`. The results of applying these IIR filters as discrete digital filters are shown in Figure (4.15).



(a) Plot of the ideal, reconstructed, and corrected pulse for a step signal without the initial padding and a 0.1 flux pulse amplitude.

(b) Plot of the ideal, reconstructed, and corrected pulse for a rectangular pulse with no padding and a 0.5 flux pulse amplitude.

Figure 4.15: Comparisons between the ideal flux pulse (blue), the actual flux pulse (dark green), flux pulse after the application of the discrete IIR corrections (light green) for different flux pulses.

As expected, the quality of the correction obtained using the IIR digital filter is lower compared to that achieved by applying the exponential correction via inversion (as in Equation (4.11)). This

is evidenced by the resulting NMSE: in the first case, the application of the IIR filter yields an NMSE of 0.1221%, compared to 0.0902% obtained with the previous method. In the second case, the NMSE decreases from 0.1282% to 0.0361% following the IIR correction. Although this represents a noticeable reduction in performance relative to the inversion-based approach, the resulting NMSE remains remarkably low.

FIR corrections

Once the feedback and feedforward taps for the IIR filters have been determined, it is necessary to implement a finite impulse response filter to correct for residual distortions on shorter timescales, less than 30 ns, which are not fully addressed by the IIR. A Finite Impulse Response (FIR) filter produces its output as a weighted sum of the current and a finite number of past input samples. It is mathematically defined by

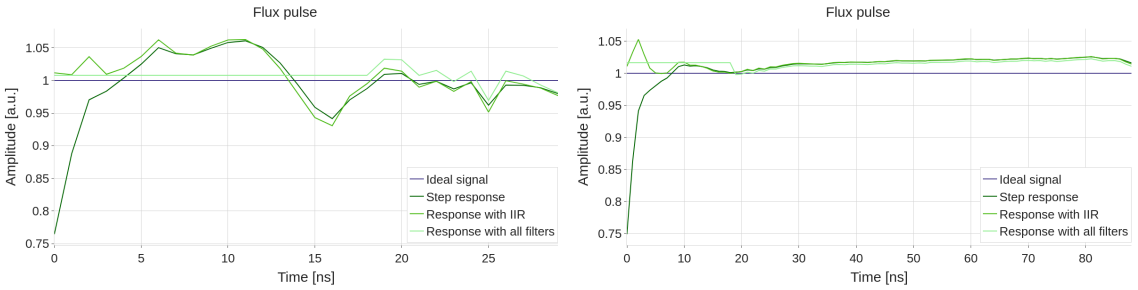
$$y[n] = \sum_{i=0}^{N} b_i x[n-i], \tag{4.23}$$

where $x[n]$ is the input signal and $y[n]$ is the output signal of the filter.

The filter coefficients are obtained by minimizing the average relative deviation between the filtered output (the result of applying the FIR filter to the signal already corrected by the IIR filter) and the ideal step response. This optimization is carried out using the `CMA-ES` algorithm (for further details on the `CMA-ES` algorithm, see Section (3.4.1)).

For our initial tests, we chose to use 20 taps.
Given that the OPX1000 controller for the qw11q operates at a sampling rate of 1 GSa/s, this corresponds to one tap per nanosecond. Accordingly, in these initial tests, we optimized 20 parameters to determine the 20 filter taps.

The results of applying the FIR filters are shown in Figure (4.16). Specifically, Figure (4.16a) displays the results of applying the FIR filter to the flux pulse generated using waveform_1, while Figure (4.16b) shows the corresponding results for the flux pulse generated with waveform_2. Both sets of filtered signals were obtained using the `lfilter` function from the `SciPy` library.



(a) Plot of the ideal, reconstructed, and corrected pulse for a step signal without the initial padding and a 0.1 flux pulse amplitude.

(b) Plot of the ideal, reconstructed, and corrected pulse for a rectangular pulse with no padding and a 0.5 flux pulse amplitude.

Figure 4.16: Comparisons between the ideal flux pulse (blue), the actual flux pulse (dark green), the flux pulse after the application of the discrete IIR corrections (green), and both IIR and FIR corrections (light green) for different flux pulses.

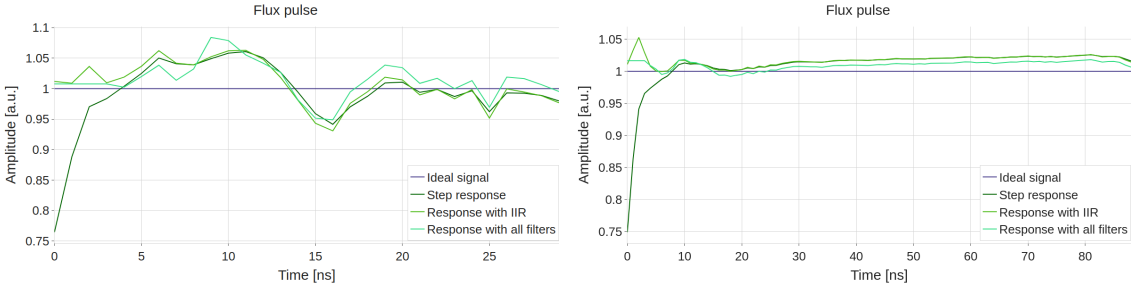
The first observation, consistent with the results shown in Figures (4.11), (4.12), and (4.15), is that there may be an issue with signal normalization that we are not addressing yet. This is evident because, in theory, the FIR filter coefficients are optimized to bring the filtered flux signal as close as possible to unity. However, the results show that even where the FIR filters should adjust the signal to exactly one, the value remains slightly above unity.

A second important consideration concerns the correction strategy. When the number of parameters optimized equals the number of FIR filter coefficients, and this coincides with the number of signal samples, there is effectively a one-to-one mapping between optimized parameters, filter coefficients, and signal points. This means that the optimization is essentially adjusting the signal point-by-point to match the desired target values. Although this procedure yields a close fit to the reference signal, its practical utility is limited, particularly because the signal used for optimization

is pre-recorded rather than acquired in real-time. As a result, future signals may exhibit different features, reducing the generalizability of the resulting filter.

To improve robustness, it is preferable to reduce the number of free parameters in the optimization. A finer-grained optimization, where individual parameters are assigned to each filter coefficient and time sample, can be restricted to the initial portion of the response (approximately the first 5 ns) where the distortions due to the IIR filters are most pronounced. Beyond this interval, a coarser parameterization can be adopted by grouping adjacent FIR coefficients under a single optimization parameter. This strategy reduces the dimensionality of the problem while promoting a more general filter profile, better suited for correcting distortions in signals that differ from the original reference.

As a practical example, we repeated the FIR coefficient optimization using 17 parameters with the `CMA-ES` algorithm. The first 4 parameters correspond directly to the first 4 FIR coefficients, while the remaining 13 parameters each control two coefficients, effectively determining 26 coefficients. Given the 1 GSa/s sampling rate, this approach optimizes the correction over a time window of approximately 30 ns. The results of this improved optimization are shown in Figure (4.17).



(a) Plot of the ideal, reconstructed, and corrected pulse for a step signal without the initial padding and a 0.1 flux pulse amplitude.

(b) Plot of the ideal, reconstructed, and corrected pulse for a rectangular pulse with no padding and a 0.5 flux pulse amplitude.

Figure 4.17: Comparisons between the ideal flux pulse (blue), the actual flux pulse (dark green), the flux pulse after the application of the discrete IIR corrections (green), and both IIR and FIR corrections (light green) for different flux pulses.

As expected, the application of FIR filters leads to a clear improvement in signal quality. For the traces shown in Figure (4.17a), the NMSE decreases from 0.1221% to 0.1083% in the first case, and from 0.0361% to 0.0139% in the second.

Output filters calculations

Using `Qibolab` the filters are provided to the control electronics via the `parameters.json` file, which contains the coefficients for the feedforward and feedback components according to the equation

$$y[n] = \sum_{m=1}^{M} a_m y[n-m] + \sum_{k=0}^{K} b_k x[n-k], \tag{4.24}$$

where $y[n]$ is the output signal, $x[n]$ is the input waveform, a_m are the feedback coefficients, and b_k are the feedforward coefficients.

In our case, we have a set of feedback coefficients determined through IIR correction and two sets of feedforward coefficients: the first obtained from the IIR-based correction, and the second from FIR-based correction on short timescales. To uniquely determine the coefficients to be passed to the electronics, it is necessary to derive a single set of feedforward coefficients and a single set of feedback coefficients by combining the two sets of feedforward filters through convolution.

Indeed, we can consider the input signal x to which we apply a first IIR filter, thus obtaining an output signal y such that

$$y[n] = \sum_{m=1}^{M} a[m] y[n-m] + \sum_{k=0}^{N} b[k] x[n-k]. \tag{4.25}$$

If we apply a second IIR filter to the y signal we obtain a z signal as output, that can be written as

$$z[n] = \sum_{m=1}^{M} a'[m]z[n-m] + \sum_{k=0}^{N} b'[k]y[n-k]. \tag{4.26}$$

Now we consider Equation (4.25) and rewrite it as

$$y[n] - \sum_{m=1}^{M} a[m]y[n-m] = \sum_{k=0}^{N} b[k]x[n-k], \tag{4.27}$$

then by applying a Z-transform, we obtain

$$Y(z)\left(1 - \sum_{m=1}^{M} a[m]z^{-m}\right) = X(z)\left(\sum_{k=0}^{N} b[k]z^{-k}\right) \tag{4.28}$$

so that

$$H_1(z) = \frac{Y(z)}{X(z)} = \frac{\sum_{k=0}^{N} b[k]z^{-k}}{1 - \sum_{m=1}^{M} a[m]z^{-m}} = \frac{B(z)}{A(z)} \tag{4.29}$$

$$\rightarrow Y(z) = H_1(z)X(z) = \frac{B(z)}{A(z)}X(z) \tag{4.30}$$

We can do the same also for Equation (4.26) and rewrite it as

$$z[n] = \sum_{m=1}^{M} a'[m]z[n-m] + \sum_{k=0}^{N} b'[k]y[n-k], \tag{4.31}$$

which, by applying the Z-transform becomes

$$Z(z)\left(1 - \sum_{m=1}^{M} a'[m]z^{-m}\right) = Y(z)\left(\sum_{k=0}^{N} b'[k]z^{-k}\right). \tag{4.32}$$

Again we can write the the transfer function

$$H_2(z) = \frac{Z(z)}{Y(z)} = \frac{\sum_{k=0}^{N} b'[k]z^{-k}}{1 - \sum_{m=1}^{M} a'[m]z^{-m}} = \frac{B'(z)}{A'(z)} \tag{4.33}$$

$$\rightarrow Z(z) = H_2(z)Y(z) = \frac{B'(z)}{A'(z)}Y(z) = \frac{B'(z)}{A'(z)}\frac{B(z)}{A(z)}X(z) \tag{4.34}$$

$$= \left(\frac{\sum_{k=0}^{N} b'[k]z^{-k}}{1 - \sum_{m=1}^{M'} a'[m]z^{-m}}\right)\left(\frac{\sum_{k=0}^{N} b[k]z^{-k}}{1 - \sum_{m=1}^{M} a[m]z^{-m}}\right)X(z) \tag{4.35}$$

From the transfer function we can obtain the expression for $Z(z)$ in terms of $X(z)$

$$\rightarrow Z(z)\left(1 - \sum_{m=1}^{M} a'[m]z^{-m}\right)\left(1 - \sum_{m=1}^{M} a[m]z^{-m}\right) = \left(\sum_{k=0}^{N} b'[k]z^{-k}\right)\left(\sum_{k=0}^{N} b[k]z^{-k}\right)X(z) \tag{4.36}$$

$$\rightarrow Z(z)\left(\sum_{m=0}^{M} a'[m]z^{-m}\right)\left(\sum_{m=0}^{M} a[m]z^{-m}\right) = \left(\sum_{k=0}^{N} b'[k]z^{-k}\right)\left(\sum_{k=0}^{N} b[k]z^{-k}\right)X(z) \tag{4.37}$$

where in the last step, the term 1 has been absorbed into the summation as the coefficient a_0. By expanding the products, we obtain:

$$\left(\sum_{m=0}^{M} a'[m]\right)\left(\sum_{m=0}^{M} a[m]\right) = \sum_{m=0}^{2M}\sum_{i=0}^{m} a'[i]a[m-i] = c[k], \qquad \text{with} \quad m = 0, \ldots, 2M, \tag{4.38}$$

$$\left(\sum_{k=0}^{N} b'[k]\right)\left(\sum_{k=0}^{N} b[k]\right) = \sum_{k=0}^{2N}\sum_{i=0}^{k} b[i]b[k-i] = d[k], \qquad \text{with} \quad k = 0, \ldots, 2N. \tag{4.39}$$

It is then possible to re-write equation (4.37) using the new expression for the filters

$$Z(z)\left(\sum_{m=0}^{2M} c[m]z^{-m}\right) = \left(\sum_{k}^{2N} d[k]z^{-k}\right) X(z) \tag{4.40}$$

$$\rightarrow Z(z)\left(1 - \sum_{m=1}^{2M} c[m]z^{-m}\right) = \left(\sum_{k}^{2N} d[k]z^{-k}\right) X(z) \tag{4.41}$$

then we apply the inverse-Z-transform and obtain

$$z[n] - \sum_{m=1}^{2M} c[m]z[n-m] = \sum_{k=0}^{2N} d[k]x[n-k] \tag{4.42}$$

$$z[n] = \sum_{m=1}^{2M} c[m]z[n-m] + \sum_{k=0}^{2N} d[k]x[n-k] \tag{4.43}$$

where the feedforward (feedback) taps of the final filters, are given by the convolution of the feedforward (feedback) taps of the two filters as shown in Equation (4.38) and Equation (4.39).

Results

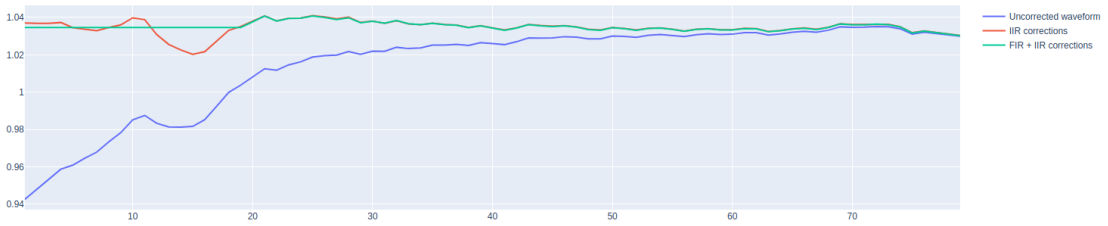


Figure 4.18: Output of the cryoscope routine on qubit `B3`.

Figure (4.18) shows the result of the cryoscope routine executed on qubit `B3`. As previously discussed, the application of a single IIR filter is sufficient to correct the exponential rise in the step response. However, on longer timescales, residual oscillations remain that are not addressed by either the IIR or FIR filters.

While we could attempt to mitigate these effects by combining multiple IIR and FIR filters, the remaining signal distortions appear to be oscillatory in nature and are likely due to ringing phenomena, transient oscillations that typically arise from impedance mismatches, or resonant components in the control line. These ringing effects are particularly evident in the acquisitions shown in Figure (4.19) and (4.20), and persist even after the application of corrective filtering, as confirmed by the reconstructed flux pulses displayed in Figures (4.21) and (4.22).

In particular, Figure (4.21) and Figure (4.22) show the reconstructed flux pulse when the digital filters determined with the cryoscope routine are applied, the measured NMSE for the two signal is equal to 0.0101% and 0.0135%; in both plots we can see that the oscillations decay in amplitude over time and appear to vanish after approximately 75 ns. Nonetheless, we expect that the application of these filters will significantly improve the control of the qubit frequency.

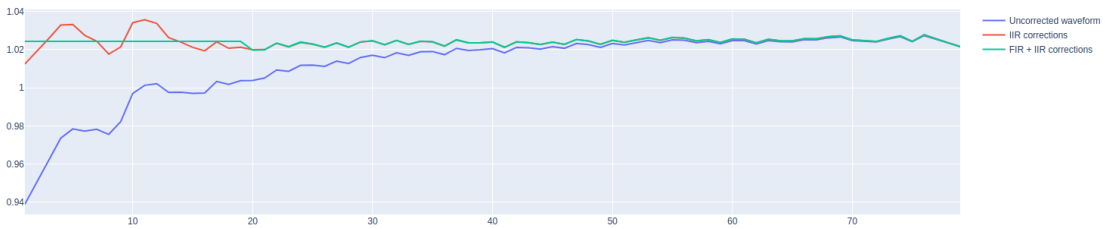


Figure 4.19: Output of the cryoscope routine on qubit `B2`.

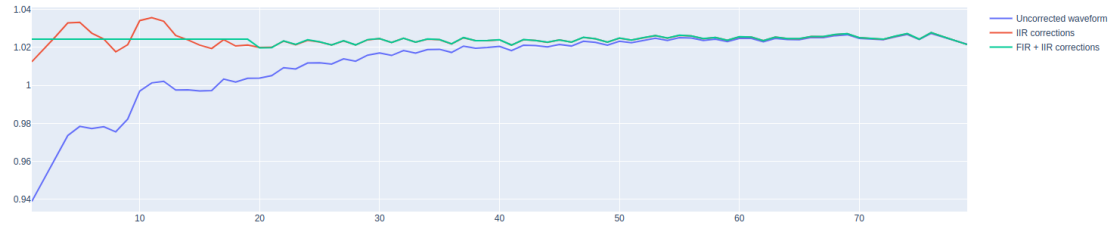


Figure 4.20: Output of the cryoscope routine on qubit `B4`.

Figure 4.21: Flux pulse reconstruction for qubit `B2`.

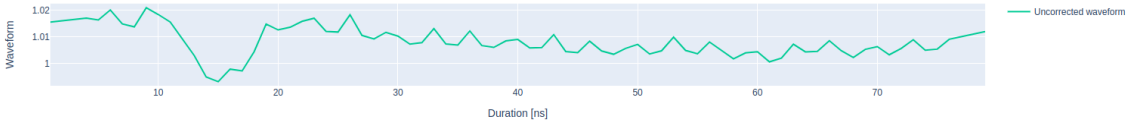


Figure 4.22: Flux pulse reconstruction for qubit `B4`.

Experiments in which flux signal distortions are particularly evident are those where a well-defined chevron pattern is expected. Deviations from the ideal chevron structure often indicate the presence of flux distortions in the control signal [55].

An example of such an experiment is the routine referred to as chevron in `Qibocal` . The name derives from the characteristic chevron-shaped interference pattern that serves as the expected output of the protocol. This calibration routine is used to characterize and tune two-qubit native gates, such as the CNOT or iSWAP, which are typically implemented using sequences of flux pulses, possibly applied to multiple qubits, combined with virtual Z-rotations. Both CNOT and iSWAP are entangling gates: the CNOT gate flips the state of the target qubit conditional on the control qubit being in the $|1\rangle$ state, while the iSWAP gate exchanges the quantum states $|10\rangle \leftrightarrow |01\rangle$, introducing a phase factor of i.

For example, we can consider the pulse sequence used to calibrate the iSWAP gate between a pair of qubits consists of a π pulse followed by a flux pulse of varying amplitude and duration, applied to the qubit with the highest frequency in the pair. The initial π pulse brings the qubit into the state $|1\rangle$, while the flux pulse detunes its frequency near resonance with the second qubit.

The iSWAP gate exploits the coherent exchange of excitations between two capacitively coupled qubits by tuning them into resonance via a flux pulse. When brought into resonance, the computational basis states $|10\rangle$ and $|01\rangle$ become energetically near-degenerate and interact via the coupling, resulting in a level repulsion characteristic of an avoided crossing. In this regime the population oscillates between the two states with a frequency determined by the coupling strength g; the expected population oscillation pattern follows:

$$p_e(t, \Delta) = \frac{\Delta^2}{\Delta^2 + 4g^2} + \frac{4g^2}{\Delta^2 + 4g^2} \cos^2\left(\frac{\sqrt{\Delta^2 + 4g^2}}{2} t\right) \tag{4.44}$$

where $\Delta = \omega_1 - \omega_2$ is the detuning between the two qubits, and g is the coupling constant. This probability distribution precisely describes what is commonly referred to in the literature as the chevron pattern.

In `Qibocal`, this routine is used to calibrate native two-qubit gates. For instance, by adjusting the duration t of the flux pulse such that $t = \pi/2g$ the system undergoes a complete excitation exchange, effectively implementing an iSWAP gate up to local phase corrections. If, instead, the goal is to calibrate a CZ gate, the pulse sequence remains the same, with the only difference being the inclusion of an initial π-pulse applied to the lower-frequency qubit. This ensures that both qubits are initially prepared in the $|1\rangle$ state.

The ideal chevron pattern is shown in Figure (4.23), the figure is meant to give a qualitative idea of how the chevron plot should look like, the scale is different from the real system.

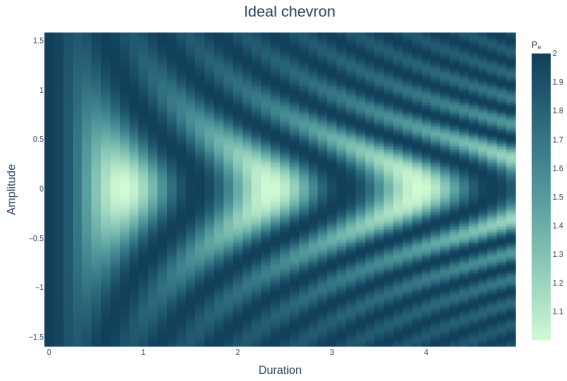


Figure 4.23: Ideal chevron pattern.

Figure (4.24) and Figure (4.25) show the actual chevron pattern obtained from the calibration of a CZ gate on our hardware without applying any digital filter.

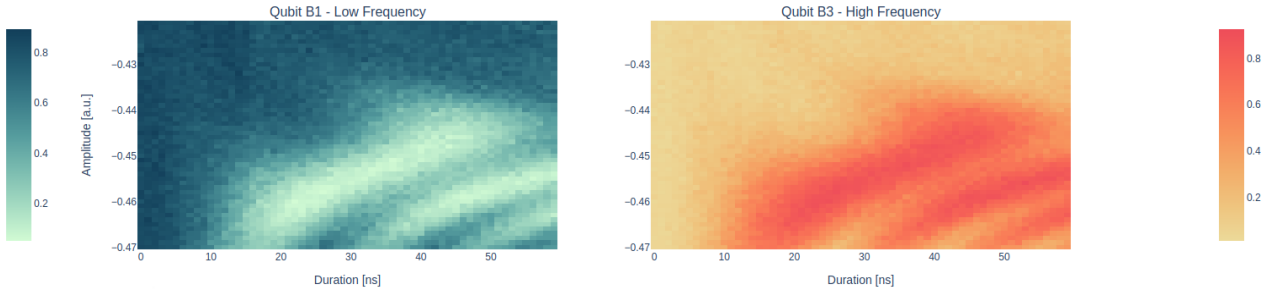


Figure 4.24: Chevron pattern obtained from the calibration of a CZ gate on qubits `B1` and `B3`.
No filters applied.

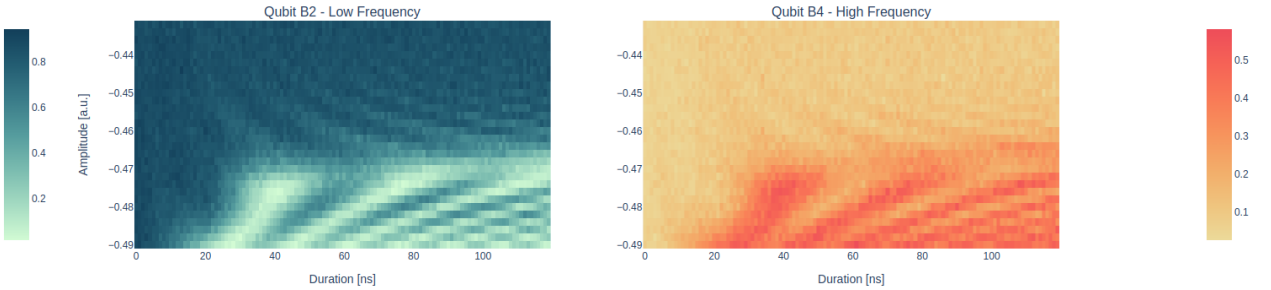


Figure 4.25: Chevron pattern obtained from the calibration of a CZ gate on qubits `B2` and `B4`.
No filters applied.

After the application of the filters determined by using the cryoscope protocol the chevron pattern that we obtained is shown in Figure (4.26) and Figure (4.27).

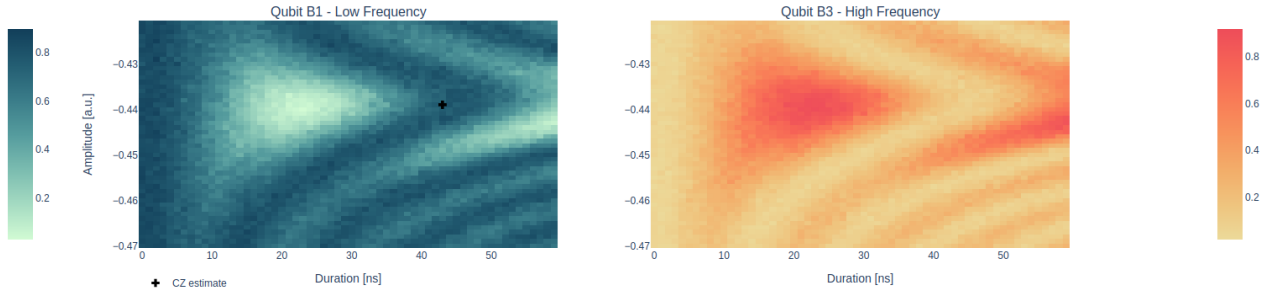


Figure 4.26: Chevron pattern obtained from the calibration of a CZ gate on qubits `B1` and `B3`.

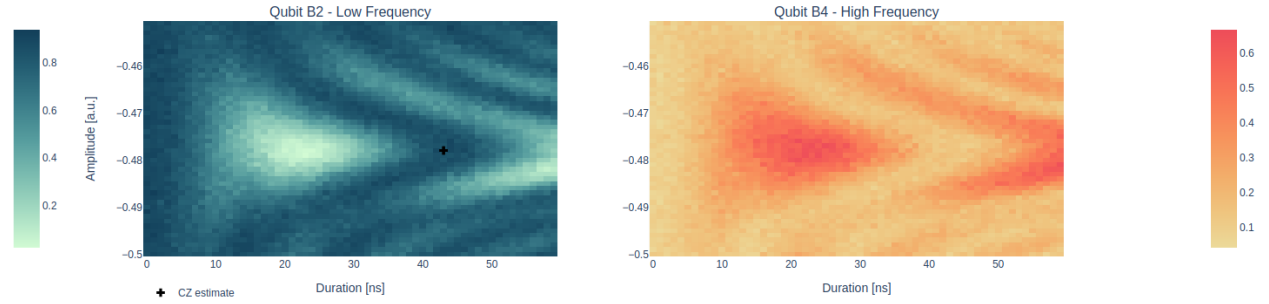


Figure 4.27: Chevron pattern obtained from the calibration of a CZ gate on qubits `B2` and `B4`.

4.2.2 Final protocol implementation

Based on the results presented in Subsection (4.2.1) - Results, we have implemented a new `Routine`, which is available starting from version v0.2 of `Qibocal` . As with all other routines in `Qibocal` , users can access it through a runcard[6]. An example of such runcard is shown in Listing (4.3).

The final implementation of the routine is grounded in the considerations and analyses presented in Subsection (4.2.1). In particular, we set the separation time T_{sep} to 100 ns longer than the maximum duration of the flux pulse, consistently with the choices taken in the preliminary analyses. The variation in pulse duration $\Delta\tau$, is defined by the user and determines the temporal resolution for sampling the time-dependent frequency shift. The lower bound of $\Delta\tau$ is constrained by the sampling capabilities of the acquisition hardware.

Based on outcomes and considerations of the preliminary analyses, we chose to use a standard rectangular pulse that begins at time $t = 0$ ns and maintains a constant, non-zero amplitude. The duration of each flux pulse is determined by the parameters defined by the user in the runcard. Specifically, users specify the minimum (`duration_min`) and maximum (`duration_max`) durations of the pulse, as well as the increment step (`duration_step`), which corresponds to $\Delta\tau$.

The amplitude of the flux pulse, specified via the `flux_pulse_amplitude` parameter, is also configurable in the runcard. Although user-defined, this amplitude should ideally induce a detuning from the qubit's sweet spot comparable to what is used in standard flux-pulsed gate operations, thereby ensuring that the measured system response reflects practical operational conditions. For example, it is useful to study the qubit response to a flux pulse which induces a frequency detuning of approximately 1 GHz. This detuning value is commonly employed in high-fidelity entangling gate implementations [55], [56], [57].

Reconstruction of the flux pulse is performed using the `savgol` filter from the SciPy library, with a polynomial order of 2 and a default `window_length` corresponding to 7 points, as discussed in Subsection (4.2.1) - Flux pulse reconstruction.

The Cryoscope protocol enables the extraction of the feedforward and feedback coefficients used to define these filters. In the `Qibo` setup, the resulting coefficients are stored in the platform configuration file (`parameters.json`), and, once the setup is updated via the `qq update` command, the filters are automatically applied to all flux pulses.

Regarding filter implementation, we have currently opted for a single exponential correction to compute the IIR taps. This decision is supported by the analysis presented in Subsection (4.2.1) - IIR corrections, and by the initial experiments in which cryoscope-derived filters were applied (see Subsection (4.2.1) - Results). The results suggest that a single exponential correction is sufficient for effective IIR filtering. For FIR filtering, the number of FIR filter taps can be selected by the user via the `fir` parameter, if not specified the default value is of 20 taps.

Listing 4.3: Runcard example to run a Cryoscope experiment.

```
platform: qw11q

targets: [B1]

actions:
  - id: cryoscope

    operation: cryoscope
    parameters:
      duration_max: 80
      duration_min: 1
      duration_step: 1
      flux_pulse_amplitude: 0.5
      relaxation_time: 50000
      fir: 30
```

[6]In Qibocal, all protocols are deployed through a YAML-based runcard, which allows users to specify experiment-specific parameters.

Chapter 5

Conclusions and outlook

In this thesis, after introducing the main theoretical concepts and the hardware and software tools employed, the focus shifted to the calibration procedures for single qubits and their optimization. The theoretical framework was focused on superconducting quantum devices, which constitute the experimental platform used throughout the work.

Calibration experiments As a preliminary step for this work, it was essential to gain practical experience with the calibration procedures required for operating a superconducting qubit. For this reason, the second Chapter of the thesis is devoted to describing in detail the full calibration workflow for a single-qubit device. In addition to outlining the experimental steps, I also aimed to provide basic explanations of the underlying physical principles, particularly those from quantum mechanics and circuit quantum electrodynamics (cQED), that enable control and readout of superconducting qubits.

Becoming familiar with the reasoning behind each calibration step was important for effectively working with both the hardware and the software tools used in the project. To support this understanding, I carried out a full calibration of a specific qubit line, namely the D-line of the `qw11q` chip. The results of these calibration procedures are summarized in Table (2.1), and served as the foundation for the subsequent experiments focused on gate fidelity optimization.

RB optimization In Chapter 3, I presented the results of an experimental study exploring the feasibility of automated optimization routines for improving the average fidelity of single-qubit Clifford gates, using Randomized Benchmarking (RB) as the primary figure of merit. The results show that, despite several technical challenges, such routines can yield measurable improvements in fidelity.

From a performance standpoint, the Nelder-Mead simplex algorithm demonstrated the most reliable convergence, although at the cost of high resource consumption due to the large number of RB evaluations required at each step. In contrast, `Optuna` allowed for efficient and broad exploration of the parameter space within a more limited runtime, especially when the number of trials was constrained. However, the optimization outcomes were often unstable: small changes in control parameters could lead to significant variations in the resulting avarage Clifford gate fidelity. This behavior highlights a limitation in the current approach, namely a possible lack of robustness in the system's response to parameter tuning due to parameters drift.

The main drawback of this optimization strategy lies in the time cost associated with evaluating the fidelity via RB. Each iteration of the optimization routine involves executing an RB sequence, making the overall process time-consuming. This is a substantial limitation, especially considering the broader applicability of RB as a fidelity estimator across different quantum platforms, including those with slower gate execution times than superconducting qubits.

To address this bottleneck, a natural direction for future work is to try optimize the RB routine itself. Reducing the depth of the Clifford sequences or the number of circuit samples used per evaluation could help shorten runtime without significantly sacrificing accuracy. This could be achieved by performing a systematic hyperparameter optimization of the RB settings.

Despite the current limitations, automating calibration remains an essential step toward scalable quantum control. Without such routines, calibration would rely on manual tuning, which involves iteratively guessing suitable parameter configurations; a slow and impractical approach, especially as systems grow in size and complexity. In summary, although the system is not yet ready for

deployment as a fully reliable calibration tool, this work provides an initial basis for exploring automated, hardware-level optimization procedures that could contribute to improving gate fidelity in a scalable and maintainable way.

Calibration routines In Chapter 4 of this work I presented the results of practical improvements to the calibration framework based on challenges and requirements that emerged from experimental operations. Specifically, we introduced a set of enhancements to the `Qibolab` and `Qibocal` libraries to support the $R_X(\pi/2)$ gate as a native operation. This involved both modifying the back-end to natively implement the gate and updating the associated calibration routines, which were originally designed only for the $R_X(\pi)$ gate. The inclusion of $R_X(\pi/2)$ as a native gate is important as such $\pi/2$ rotations are frequently used in quantum algorithms and benchmarking protocols. An inaccurate calibration would propagate systematic errors throughout the execution of quantum circuits, ultimately degrading their overall fidelity.

The chapter also introduces the implementation of the Cryoscope protocol to correct for dynamical distortions in the control lines used to apply flux pulses. In superconducting qubit platforms, accurate frequency control via flux modulation is fundamental for high-fidelity single- and two-qubit gates. However, as the flux signal propagates through the various components of the control line, it is subject to distortions resulting from finite bandwidth, reflections, and impedance mismatches. These effects introduce deviations between the intended and actual qubit trajectories, which in turn affect gate performance. The Cryoscope protocol enables a detailed, time-domain characterization of these distortions, providing the basis for computing pre-distorted waveforms that effectively cancel out the unwanted dynamics. The improvements obtained using these corrected pulses are particularly evident in the chevron calibration routine in `Qibocal` , which plays a central role in tuning flux-based two-qubit gates such as CNOT and iSWAP.

To further enhance the fidelity of flux pulses, additional corrective strategies can be employed. These include techniques designed to minimize leakage outside the computational subspace, compensate for pulse echoes resulting from impedance mismatches, and suppress residual ringing that follows sharp transitions in the control waveform. Other approaches focus on more precise correction of long-time distortions, such as the method described in the supplementary material of [58]. Collectively, these improvements contribute to higher gate fidelity, increased pulse robustness, and more accurate temporal control of qubit dynamics.

Appendix A

2D notch resonator resonance profile

A more accurate description of the resonance profile for a two-dimensional notch-type resonator is provided in the literature [59], where the transmitted signal is modeled as a complex function of frequency. The complete expression for the complex transmission coefficient is given by

$$S_{21}(f) = ae^{i\alpha}e^{-2\pi i f\tau}\left[1 - \frac{(Q_l/|Q_c|)e^{i\phi}}{1 + 2iQ_l(f/f_r - 1)}\right]. \tag{A.1}$$

This formulation provides a deeper insight into both the environmental and resonator-specific contributions to the observed response. The prefactor accounts for the attenuation and gain in the measurement chain through the amplitude a, introduces a global phase shift α, and models the effect of finite cable length via a frequency-dependent delay term characterized by the parameter τ. These components collectively represent systematic effects that shape the baseline response of the system.

The second part of the expression captures the physical behavior of the resonator itself. It includes the loaded quality factor Q_l, which reflects the total energy loss in the system, and the coupling quality factor Q_c, which quantifies the interaction between the resonator and the transmission line. The resonance frequency f_r marks the center of the frequency response, while the phase offset ϕ accounts for impedance mismatches and Fano-type interferences caused by reflections within the setup. This model exploits all the information encoded in both the magnitude and phase of the complex transmission signal, allowing for a more comprehensive characterization.

However, directly fitting the complete model involves a nonlinear optimization over seven parameters, which is inherently sensitive to initial conditions and often non-robust. To improve the stability and reliability of the fitting process, the problem is decomposed and approached in stages, each targeting a subset of parameters.

First, the cable delay is estimated by fitting a linear function to the phase response across frequency. The delay introduces a phase tilt proportional to the frequency, and its removal reveals the underlying circular shape of the resonance response in the complex plane. Once the delay is corrected, a geometric circle fit is performed on the data.

Then the phase angle θ is fit according to the following expression:

$$\theta(f) = \theta_0 - 2\pi\tau(f - f_r) + 2\arctan\left[2Q_l\left(1 - \frac{f}{f_r}\right)\right], \tag{A.2}$$

where an additional linear term accounts for any residual cable delay not captured in the initial correction. This protocol is implemented in `Qibocal` and available by selecting the `s21` fitting routine for the resonator spectroscopy, an example of the expected output obtained from running such experiment is shown in Figure (A.1).

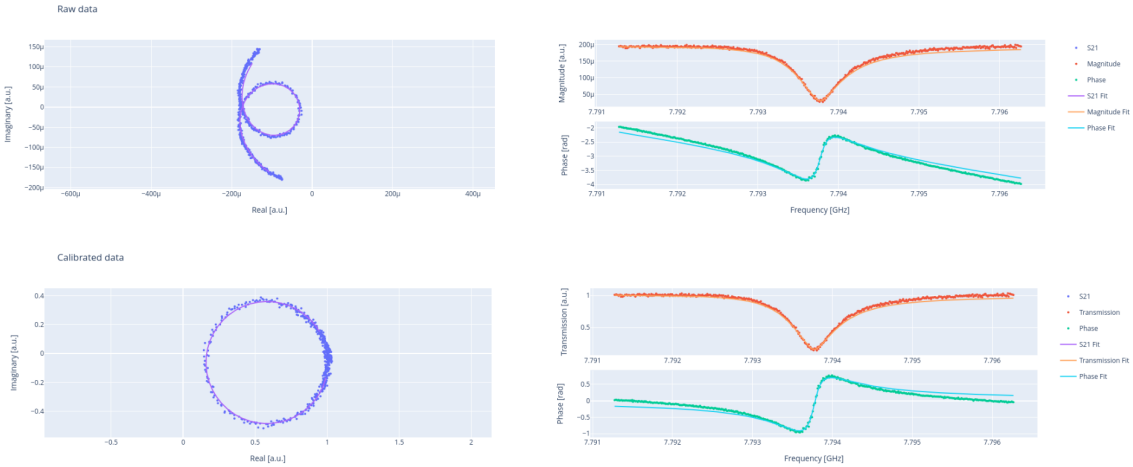


Figure A.1: Output of resonator spectroscopy performed with `s21` fit function.

Appendix B

Phase reconstruction

In general, it is possible to show that for different forms of the detuning flux

$$\Delta f(\Phi) = a\Phi^k \tag{B.1}$$

where $k \in \mathbb{Z}^+$, the relative phase φ_τ can be calculated as follows:

$$\varphi_\tau = 2\pi a \int_0^\infty \left[\int_0^\infty h(t-t')dt' - \int_0^\infty h(t-\tau-t')dt'\right]^k dt \tag{B.2}$$

$$= 2\pi a \int_0^\tau \left[\int_0^t h(t-t')dt'\right]^k dt + 2\pi a \int_\tau^\infty \left[\int_0^\tau h(t-t')dt'\right]^k dt, \tag{B.3}$$

From the calculations showed in (4.3) we know that the relative phase φ_τ for a general form of the detuning flux (B.1), with $T_{\text{sep}} = \infty$ is

$$\varphi_\tau = 2\pi \int_0^{+\infty} a\left[(s(t) - s(t-\tau))\right]^k dt = 2\pi a \int_0^{+\infty} \left[(s(t) - s(t-\tau))\right]^k dt \tag{B.4}$$

As hypothesis we know that voltage-to-flux step response of the control line is

$$s(t) = \left(1 - e^{-t/\tau}\right) \cdot u(t), \tag{B.5}$$

where $u(t)$ is the step function, and that the impulse response is

$$h(t) = \frac{\mathrm{d}s}{\mathrm{d}t} \tag{B.6}$$

If we substitute the expression of $s(t)$ given in (B.5) in equation (B.4) we obtain

$$\varphi_\tau = 2\pi a \int_0^{+\infty} \left[\int_0^{+\infty} h(t-t')dt' - \int_0^{+\infty} h(t-\tau-t')dt'\right]^k dt. \tag{B.7}$$

To do we have to show that

1.
$$s(t) = \int_0^{+\infty} h(t-t')dt' \tag{B.8}$$

2.
$$s(t-\tau) = \int_0^{+\infty} h(t-\tau-t')dt' \tag{B.9}$$

We start from the demonstration of equation (B.8). By definition (B.6) we can write

$$h(t) = \frac{d}{dt}\left[\left(1 - e^{-t/\tau}\right)u(t)\right] = \frac{e^{-t/\tau}}{\tau}u(t) + \left(1 - e^{-t/\tau}\right)\delta(t), \tag{B.10}$$

substituting Eq (B.10) in Eq (B.8) we obtain

$$\int_0^{+\infty} h(t-t')dt' = \int_0^{+\infty} \frac{e^{-\frac{(t-t')}{\tau}}}{\tau} u(t-t')dt' + \int_0^{+\infty} \left(1 - e^{-(t-t')/\tau}\right) \delta(t-t')dt', \tag{B.11}$$

by setting $t'' = t - t'$, $dt'' = -dt'$, we have $t'' \to -\infty$ for $t' \to +\infty$ and $t'' = t$ for $t' = 0$, the integral then becomes

$$\begin{aligned}
\int_0^{+\infty} h(t-t')dt' &= \int_t^{-\infty} -\frac{e^{-t''/\tau}}{\tau} u(t'')dt'' - \int_t^{-\infty} \left(1 - e^{-t''/\tau}\right) \delta(t'')dt'' && \text{(B.12)}\\
&= \int_{-\infty}^{t} \frac{e^{-t''/\tau}}{\tau} u(t'')dt'' + \int_{-\infty}^{t} \left(1 - e^{-t''/\tau}\right) \delta(t'')dt'' && \text{(B.13)}\\
&= \int_0^{t} \frac{e^{-t''/\tau}}{\tau} u(t'')dt'' + \left(1 - e^{-t''/\tau}\right)\Big|_{t''=0} && \text{(B.14)}\\
&= \left[-e^{-t''/\tau} u(t'')\right]_0^t + 0 && \text{(B.15)}\\
&= (1 - e^{-t/\tau})u(t) && \text{(B.16)}\\
& && \text{(B.17)}
\end{aligned}$$

that concludes the demonstration of Eq (B.8). To demonstrate equation (B.9) we starting again by using the definition of $s(t)$ to compute

$$\begin{aligned}
h(t-t'-\tau) &= \frac{d}{dt}\left[\left(1 - e^{-(t-t'-\tau)/\tau}\right) u(t-t'-\tau)\right] && \text{(B.18)}\\
&= \frac{e^{-(t-t'-\tau)/\tau}}{\tau} u(t-t'-\tau) + \left(1 - e^{-(t-t'-\tau)/\tau}\right) \delta(t-t'-\tau) && \text{(B.19)}\\
& && \text{(B.20)}
\end{aligned}$$

We can substitute (B.20) in equation (B.9) and obtain

$$\int_0^{+\infty} h(t-t'-\tau)\, dt' = \int_0^{+\infty} \frac{e^{-(t-t'-\tau)/\tau}}{\tau} u(t-t'-\tau)dt' + \int_0^{+\infty} \left(1 - e^{-(t-t'-\tau)/\tau}\right) \delta(t-t'-\tau)dt' \tag{B.21}$$

by setting $t'' = t - t' - \tau$, $dt'' = -dt'$, we have $t'' \to -\infty$ for $t' \to +\infty$ and $t'' = t - \tau$ for $t' = 0$, the integral then becomes

$$\begin{aligned}
\int_0^{+\infty} h(t-t'-\tau)dt' &= \int_{-\infty}^{t-\tau} \frac{-e^{-t''/\tau}}{\tau} u(t'')dt'' - \int_{t-\tau}^{\infty} (1 - e^{-t''/\tau})\delta(t'') && \text{(B.22)}\\
&= \int_{-\infty}^{t-\tau} \frac{e^{-t''/\tau}}{\tau} u(t'')dt'' + \int_{-\infty}^{t-\tau} (1 - e^{-t''/\tau})\delta(t'') && \text{(B.23)}\\
&= \int_0^{t-\tau} \frac{e^{-t''/\tau}}{\tau} u(t'')dt'' + \left(1 - e^{-t''/\tau}\right)\Big|_{t''=0} && \text{(B.24)}\\
&= \left[-e^{-t''/\tau} u(t'')\right]_0^{t-\tau} + 0 && \text{(B.25)}\\
&= (1 - e^{-t''/\tau})u(t'') && \text{(B.26)}\\
&= s(t'') = s(t-t'-\tau) && \text{(B.27)}
\end{aligned}$$

With this, we demonstrated that

$$\varphi_\tau = 2\pi a \int_0^{\infty} \left[\int_0^{\infty} h(t-t')\, dt' - \int_0^{\infty} h(t-\tau-t')dt'\right]^k dt, \tag{B.28}$$

We now need to show that

$$\begin{aligned}
&2\pi a \int_0^{\infty} \left[\int_0^{\infty} h(t-t')dt' - \int_0^{\infty} h(t-\tau-t')dt'\right]^k dt && \text{(B.29)}\\
&= 2\pi a \int_0^{\tau} \left[\int_0^{t} h(t-t')dt'\right]^k dt + 2\pi a \int_{\tau}^{\infty} \left[\int_0^{\tau} h(t-t')\, dt'\right]^k dt. && \text{(B.30)}
\end{aligned}$$

As first step we can try to rewrite the left-hand-side (LHS) in a different way:

$$2\pi a \int_0^{\infty} \left[\int_0^{\infty} h(t-t')dt' - \int_0^{\infty} h(t-\tau-t')dt' \right]^k dt \tag{B.31}$$

$$= 2\pi a \int_0^{\infty} \left[s(t) - s(t-\tau) \right]^k dt \tag{B.32}$$

$$= 2\pi\alpha \int_0^{+\infty} \left[\left(1 - e^{-t/\tau}\right) u(t) - \left(1 - e^{-(t-\tau)/\tau}\right) u(t-\tau) \right]^k dt \tag{B.33}$$

$$= 2\pi\alpha \int_0^{\tau} \left[\left(1 - e^{-t/\tau}\right) u(t) \right]^k dt + 2\pi\alpha \int_{\tau}^{+\infty} \left[\left(1 - e^{-t/\tau}\right) u(t) - \left(1 - e^{-(t-\tau)/\tau}\right) u(t-\tau) \right]^k dt \tag{B.34}$$

$$= 2\pi\alpha \int_0^{\tau} \left[\int_0^{\infty} h(t-t')dt' \right]^k dt + 2\pi\alpha \int_{\tau}^{+\infty} \left[s(t) - s(t-\tau) \right]^k dt \tag{B.35}$$

$$\tag{B.36}$$

In the first step, to get equation (B.33) I simply used the equations (B.8) and (B.9) that were demonstarted before, then by substituting the definition of $s(t)$ we obtain Eq (B.34). It is possible to separate the integral in equation (B.34) by using the definition of $u(t)$ which is null for $t\tau$ and of $u(t-\tau)$ which is null also for $0 < t < \tau$, doing this we obtain Eq, (B.35) By substituting back (B.8) in the first term of equation (B.35) we obtain the first term of equation (B.29) which we want to demonstrate, in the second term of the sum instead, we can substitute back the definition of $s(t)$.

At this point we only have to show that for $t > \tau$, which is the interval we are considering in the second term,

$$2\pi\alpha \int_{\tau}^{+\infty} \left[s(t) - s(t-\tau) \right]^k dt = 2\pi a \int_{\tau}^{\infty} \left[\int_0^{\tau} h(t-t')dt' \right]^k dt. \tag{B.37}$$

To do this we can first evaluate $s(t) - s(t-\tau)$ in the interval $t > \tau$ so that $u(t) = u(t-\tau) = 1$, we have

$$s(t) - s(t-\tau) = 1 - e^{-\frac{t}{\tau}} - 1 + e^{-\frac{(t-\tau)}{\tau}} = e^{-\frac{t}{\tau}} \left(e^{\frac{\tau}{\tau}} - 1 \right) = (e-1)e^{-\frac{t}{\tau}} \tag{B.38}$$

If we then compute $\int_0^{\tau} h(t-t')dt'$ we obtain

$$\int_0^{\tau} h(t-t')dt' = \int_0^{\tau} \left[\frac{e^{-\frac{(t-t')}{\tau}}}{\tau} u(t-t') - \left(1 - e^{-\frac{t-t'}{\tau}}\right) \delta(t-t') \right] dt' \tag{B.39}$$

$$= \int_0^{\tau} \frac{e^{-\frac{(t-t')}{\tau}}}{\tau} u(t-t')dt' \tag{B.40}$$

$$= \int_0^{\tau} \frac{e^{-\frac{(t-t')}{\tau}}}{\tau} dt' \tag{B.41}$$

$$= \int_0^{\tau} \frac{e^{-\frac{t}{\tau}} e^{\frac{t'}{\tau}}}{\tau} = e^{-\frac{t}{\tau}} e^{\frac{t'}{\tau}} \Big|_0^{\tau} = e^{\frac{-t}{\tau a u}} (e-1) \tag{B.42}$$

which is equal to (B.38) and then concludes the demonstration.

Appendix C

X and Y component measurements

In quantum computing, measurements are restricted to the computational basis, corresponding to projective measurements along the z-axis of the Bloch sphere. As a result, direct access is limited to determining the probability of a qubit being in the $|0\rangle$ or $|1\rangle$ state. To extract information about other components of the qubit state, such as its projections along the $x-$ or $y-$ axes, indirect measurement techniques are required. These involve applying single-qubit rotations that map the desired observable onto the $z-$axis before the readout. For example, in the cryoscope experiment we need to measure $\langle X\rangle$ and $\langle Y\rangle$ in order to then access the relative phase information.

$\langle X\rangle$ measurement To measure $\langle X\rangle$, we begin by applying a $\frac{\pi}{2}$ rotation around the y-axis, which transforms the ground state $|0\rangle$ into the superposition state $|0\rangle$ a $\frac{(|0\rangle+|1\rangle)}{\sqrt{2}}$. Next, a flux pulse of duration τ is applied, introducing a phase φ_τ between the computational basis states. After this evolution, the qubit state become $|\psi\rangle = \frac{|0\rangle+e^{i\varphi_\tau}|1\rangle}{\sqrt{2}}$.

The expectation value of the Pauli X operator in this state is:

$$\langle X\rangle = \left(\frac{\langle 0| + e^{-i\varphi_\tau}\langle 1|}{\sqrt{2}}\right)\hat{\sigma}_x\left(\frac{|0\rangle + e^{i\varphi_\tau}|1\rangle}{\sqrt{2}}\right) \tag{C.1}$$

$$= \left(\frac{\langle 0| + e^{-i\varphi_\tau}\langle 1|}{\sqrt{2}}\right)\left(\frac{|1\rangle + e^{i\varphi_\tau}|0\rangle}{\sqrt{2}}\right) \tag{C.2}$$

$$= \frac{e^{i\varphi_\tau}+e^{-i\varphi_\tau}}{2} = \cos(\varphi_\tau). \tag{C.3}$$

Since the measurement can only be performed along the z-axis, we apply a rotation $R_Y(\frac{\pi}{2})$ to map the x-axis component onto the z-axis. The operator for this rotation is

$$R_Y(\frac{\pi}{2}) = \exp\left\{\left(-i\frac{\pi}{4}\hat{\sigma}_y\right)\right\} \tag{C.4}$$

$$= \cos\left(\frac{\pi}{4}\right)\mathbb{I} - i\sin\left(\frac{\pi}{4}\right)\hat{\sigma}_y \tag{C.5}$$

$$= \frac{\sqrt{2}}{2}(\mathbb{I} - i\hat{\sigma}_y) = \begin{pmatrix}1 & -1\\ 1 & 1\end{pmatrix}. \tag{C.6}$$

Applying this rotation to the state $|\psi\rangle$ gives:

$$|\psi'\rangle = R_Y\left(\frac{\pi}{2}\right)|\psi\rangle = \frac{1}{\sqrt{2}}\begin{pmatrix}1 & -1\\ 1 & 1\end{pmatrix}\frac{1}{\sqrt{2}}\begin{pmatrix}1\\ e^{i\varphi_\tau}\end{pmatrix} = \frac{1}{2}\begin{pmatrix}1-e^{i\varphi_\tau}\\ 1+e^{i\varphi_\tau}\end{pmatrix} \tag{C.7}$$

We now measure the probability p_1 of finding the qubit in state $|1\rangle$:

$$p_1 = |\langle\psi'|1\rangle|^2 = \left|\frac{1}{2}\left((1-e^{i\varphi_\tau})\langle 0| + (1+e^{i\varphi_\tau})\langle 1|\right)|1\rangle\right|^2 = \left|\frac{1+e^{i\varphi_\tau}}{2}\right|^2 \tag{C.8}$$

$$= \frac{1}{4}(1+e^{-i\varphi_\tau})(1+e^{+i\varphi_\tau}) = \frac{1}{4}(1+e^{+i\varphi_\tau}+e^{-i\varphi_\tau}+1) \tag{C.9}$$

$$= \frac{(2+2\cos\varphi_\tau)}{4} = \frac{1+\cos\varphi_\tau}{2}. \tag{C.10}$$

Thus, the expectation value $\langle X\rangle$ is recovered via $\langle X\rangle = 2\cdot p_1 - 1$.

$\langle Y \rangle$ measurement To measure $\langle Y \rangle$, the same initial sequence is applied: a $\frac{\pi}{2}$ rotation around the y-axis followed by a flux pulse of duration τ, leading to the state: $|\psi\rangle = \frac{|0\rangle + e^{i\varphi_\tau}|1\rangle}{\sqrt{2}}$.

The expectation value of the Pauli Y operator is given by:

$$\langle Y \rangle = \left(\frac{\langle 0| + e^{-i\varphi_\tau} \langle 1|}{\sqrt{2}} \right) \hat{\sigma}_y \left(\frac{|0\rangle + e^{i\varphi_\tau}|1\rangle}{\sqrt{2}} \right) \tag{C.11}$$

$$= \left(\frac{\langle 0| + e^{-i\varphi_\tau} \langle 1|}{\sqrt{2}} \right) \left(\frac{|1\rangle + e^{i\varphi_\tau}|0\rangle}{\sqrt{2}} \right) \tag{C.12}$$

$$= \frac{e^{i\varphi_\tau} - e^{-i\varphi_\tau}}{2} = \sin(\varphi_\tau) \tag{C.13}$$

To access this observable through a z-axis measurement, we apply a rotation $R_X(\frac{\pi}{2})$:

$$R_X(\frac{\pi}{2}) = \exp\left\{ \left(-i\frac{\pi}{4}\hat{\sigma}_x \right) \right\} \tag{C.14}$$

$$= \cos\left(\frac{\pi}{4}\right)\mathbb{I} - i\sin\left(\frac{\pi}{4}\right)\hat{\sigma}_x \tag{C.15}$$

$$= \frac{\sqrt{2}}{2}(\mathbb{I} - i\hat{\sigma}_x) = \begin{pmatrix} 1 & -i \\ -i & 1 \end{pmatrix}. \tag{C.16}$$

The probability of measuring $|1\rangle$ is:

$$|\psi'\rangle = R_X\left(\frac{\pi}{2}\right)|\psi\rangle = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & -i \\ -i & 1 \end{pmatrix}\frac{1}{\sqrt{2}}\begin{pmatrix} 1 \\ e^{i\varphi_\tau} \end{pmatrix} = \frac{1}{2}\begin{pmatrix} 1 - ie^{i\varphi_\tau} \\ -i + e^{i\varphi_\tau} \end{pmatrix} \tag{C.17}$$

The probability of measuring $|1\rangle$ is:

$$p_1 = |\langle 1|\psi'\rangle|^2 = \left| \frac{1}{2}\langle 1| \left((1 - ie^{i\varphi_\tau})|0\rangle + (-i + e^{i\varphi_\tau})|1\rangle \right) \right|^2 = \left| \frac{-i + e^{i\varphi_\tau}}{2} \right|^2 \tag{C.18}$$

$$= \frac{1}{4}(i + e^{-i\varphi_\tau})(-i + e^{+i\varphi_\tau}) = \frac{1}{4}(1 + -ie^{-i\varphi_\tau} + e^{+i\varphi_\tau} + 1) \tag{C.19}$$

$$= \frac{(2 - 2\sin\varphi_\tau)}{4} = \frac{1 - \sin\varphi_\tau}{2}. \tag{C.20}$$

Therefore, the expectation value $\langle Y \rangle$ can be expressed as $\langle Y \rangle = -2 \cdot p_1 + 1$.

Appendix D

Flux amplitude frequency routine

Before running the cryoscope experiment, it is necessary to first execute the `flux_amplitude_frequency` routine. The purpose of this routine is to determine the coefficients of the quadratic relationship between the detuning experienced by the qubit and the amplitude of the applied magnetic flux pulse. According to equation, (D.1) This relationship is assumed to take the form:

$$f = c_1 A^2 + c_2 A + c_3, \tag{D.1}$$

where f is the detuning and A is the flux amplitude. Knowledge of this relation is essential for later reconstructing the time-dependent flux from the detuning profile extracted in the cryoscope measurement.

The pulse sequence used in this calibration is identical to that of the cryoscope experiment, and measurements are again performed in both the $\langle X(t)\rangle$ and $\langle Y(t)\rangle$ bases.

However, in this case, the duration of the flux pulse is fixed and specified by the user via the `duration` parameter, while the amplitude of the pulse is swept over a user-defined range. This range is set through the `amplitude_min`, `amplitude_max`, and `amplitude_step` parameters defined in the experiment's runcard configuration.

The postprocessing follows a procedure similar to that used for cryoscope data. First, the raw measurement probabilities are converted to expectation values by inverting the relations:

$$\langle X(t)\rangle = 2p_1^X(t) - 1, \tag{D.2}$$

$$\langle Y(t)\rangle = 1 - 2p_1^Y(t), \tag{D.3}$$

where $p_1^X(t)$ and $p_1^Y(t)$ are the measured probabilities of finding the qubit in the excited state $|1\rangle$ from the $\langle X\rangle$ and $\langle Y\rangle$ measurements, respectively.

From the reconstructed complex signal $s(t) = \langle X(t)\rangle + i\langle Y(t)\rangle$, the phase is extracted. Since this phase is wrapped within the interval $[-\pi, \pi]$, we apply the `numpy.unwrap` function to eliminate 2π discontinuities and recover a smooth, continuous phase trajectory. Knowing the idle qubit frequency, we then compute the detuning for each flux amplitude from the time derivative of the unwrapped phase.

Finally, the detuning values are fit as a function of the applied flux amplitude using a second-order polynomial. The resulting coefficients are stored and used in the cryoscope postprocessing to invert Eq. (D.1), enabling the reconstruction of the applied flux amplitude from the observed detuning.

Figure (D.1) shows an example of the output from this calibration routine.

It is important to note that the choice of `amplitude_step` must be sufficiently small to ensure accurate sampling of the $\langle X(t)\rangle$ and $\langle Y(t)\rangle$ traces. If the step size is too large, the oscillatory signals may be undersampled, leading to poor phase extraction and unreliable fit results, as illustrated in Figure (D.2).

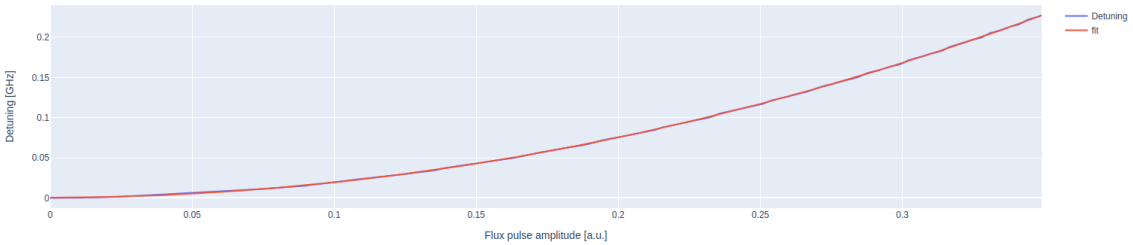


Figure D.1: Expected output of the `flux_amplitude_frequency` routine. Results obtained by executing the protocol on qubit `B2`

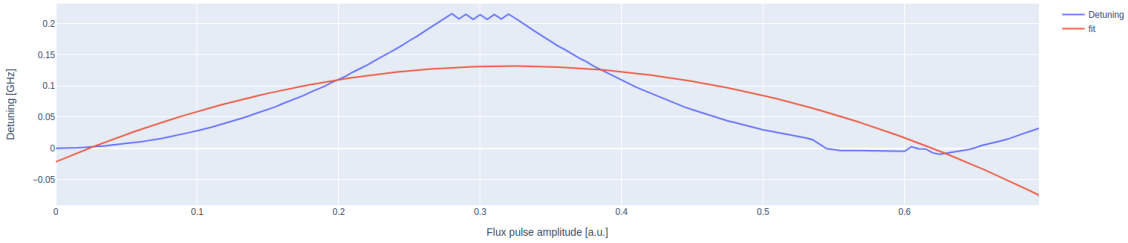


Figure D.2: Example of failed `flux_amplitude_frequency` routine due to large amplitude step size. Results obtained by executing the protocol on qubit `B4`

Bibliography

[1] M. A. Rol et al. "Time-domain characterization and correction of on-chip distortion of control pulses in a quantum processor". en. In: *Applied Physics Letters* 116.5 (Feb. 2020). arXiv:1907.04818 [quant-ph], p. 054001. ISSN: 0003-6951, 1077-3118. DOI: 10.1063/1.5133894. URL: http://arxiv.org/abs/1907.04818 (visited on 02/24/2025).

[2] Richard P. Feynman. "Simulating physics with computers". In: *International Journal of Theoretical Physics* 21.6 (June 1982), pp. 467–488. ISSN: 1572-9575. DOI: 10.1007/BF02650179. URL: https://doi.org/10.1007/BF02650179.

[3] Katherine L. Brown, William J. Munro, and Vivien M. Kendon. *Using Quantum Computers for Quantum Simulation*. 2010. DOI: 10.3390/e12112268. URL: https://doi.org/10.3390/e12112268.

[4] I. M. Georgescu, S. Ashhab, and Franco Nori. "Quantum simulation". In: *Reviews of Modern Physics* 86.1 (Mar. 2014), pp. 153–185. ISSN: 1539-0756. DOI: 10.1103/revmodphys.86.153. URL: http://dx.doi.org/10.1103/RevModPhys.86.153.

[5] Ashley Montanaro. "Quantum algorithms: an overview". In: *npj Quantum Information* 2.1 (Jan. 2016), p. 15023. ISSN: 2056-6387. DOI: 10.1038/npjqi.2015.23. URL: https://doi.org/10.1038/npjqi.2015.23.

[6] Z. Chen. "Metrology of Quantum Control and Measurement in Superconducting Qubits". ProQuest ID: Chen_ucsb_0035D_13771, Merritt ID: ark:/13030/m50p5xxf. PhD thesis. UC Santa Barbara, 2018. URL: https://escholarship.org/uc/item/0g29b4p0.

[7] David P. DiVincenzo. "The Physical Implementation of Quantum Computation". In: *Fortschritte der Physik* 48.9–11 (Sept. 2000), pp. 771–783. ISSN: 1521-3978. DOI: 10.1002/1521-3978(200009)48:9/11<771::aid-prop771>3.0.co;2-e. URL: http://dx.doi.org/10.1002/1521-3978(200009)48:9/11%3C771::AID-PROP771%3E3.0.CO;2-E.

[8] Riccardo Manenti and Mario Motta. *Quantum Information Science*. en. 1st ed. Oxford University PressOxford, Aug. 2023. ISBN: 978-0-19-878748-8 978-0-19-182959-8. DOI: 10.1093/oso/9780198787488.001.0001. URL: https://academic.oup.com/book/51893 (visited on 02/24/2025).

[9] S. Olivares. *Lecture Notes on Quantum Computing*. Nov. 2021.

[10] Jens Koch et al. "Charge insensitive qubit design derived from the Cooper pair box". en. In: *Physical Review A* 76.4 (Oct. 2007). arXiv:cond-mat/0703002, p. 042319. ISSN: 1050-2947, 1094-1622. DOI: 10.1103/PhysRevA.76.042319. URL: http://arxiv.org/abs/cond-mat/0703002 (visited on 02/24/2025).

[11] D. Vion et al. "Manipulating the quantum state of an electrical circuit". In: *Science* 296.5569 (2002), pp. 886–889. DOI: 10.1126/science.1069372.

[12] Thomas E. Roth, Ruichao Ma, and Weng C. Chew. "The Transmon Qubit for Electromagnetics Engineers: An introduction". In: *IEEE Antennas and Propagation Magazine* 65.2 (Apr. 2023), pp. 8–20. ISSN: 1558-4143. DOI: 10.1109/map.2022.3176593. URL: http://dx.doi.org/10.1109/MAP.2022.3176593.

[13] Cora N. Barrett et al. "Learning-Based Calibration of Flux Crosstalk in Transmon Qubit Arrays". In: *Phys. Rev. Appl.* 20 (2 Aug. 2023), p. 024070. DOI: 10.1103/PhysRevApplied.20.024070. URL: https://link.aps.org/doi/10.1103/PhysRevApplied.20.024070.

[14] E.T. Jaynes and F.W. Cummings. "Comparison of quantum and semiclassical radiation theories with application to the beam maser". In: *Proceedings of the IEEE* 51.1 (1963), pp. 89–109. DOI: 10.1109/PROC.1963.1664.

[15] Ben Chiaro and Yaxing Zhang. *Active Leakage Cancellation in Single Qubit Gates*. 2025. arXiv: 2503.14731 [quant-ph]. URL: https://arxiv.org/abs/2503.14731.

[16] Philip Krantz et al. "A Quantum Engineer's Guide to Superconducting Qubits". en. In: *Applied Physics Reviews* 6.2 (June 2019). arXiv:1904.06560 [quant-ph], p. 021318. ISSN: 1931-9401. DOI: 10.1063/1.5089550. URL: http://arxiv.org/abs/1904.06560 (visited on 02/24/2025).

[17] *QuantWare Contralto-D*. URL: https://www.quantware.com/product/contralto.

[18] *XLDsl dilution refrigerator, BLUEFORS*. URL: https://bluefors.com/products/4k-systems/xldsl-4k-measurement-system/.

[19] *opx1000 hybrid control platform, QUANTUM MACHINES*. URL: https://www.quantum-machines.co/products/opx1000/.

[20] Andrea Pasquale et al. *Qibocal: an open-source framework for calibration of self-hosted quantum devices*. en. arXiv:2410.00101 [quant-ph]. Oct. 2024. DOI: 10.48550/arXiv.2410.00101. URL: http://arxiv.org/abs/2410.00101 (visited on 02/24/2025).

[21] *Qibocal website*. URL: https://qibo.science/qibocal/stable/.

[22] *Qibocal source code, GitHub*. URL: https://github.com/qiboteam/qibocal.

[23] Stavros Efthymiou et al. "Qibolab: an open-source hybrid quantum operating system". en. In: *Quantum* 8 (Feb. 2024). arXiv:2308.06313 [quant-ph], p. 1247. ISSN: 2521-327X. DOI: 10.22331/q-2024-02-12-1247. URL: http://arxiv.org/abs/2308.06313 (visited on 02/24/2025).

[24] *Qibolab website*. URL: https://qibo.science/qioblab/stable/.

[25] *Qibolab source code, GitHub*. URL: https://github.com/qiboteam/qibolab.

[26] I. I. Rabi. "On the Process of Space Quantization". In: *Physical Review* 49.4 (1936), pp. 324–328. DOI: 10.1103/PhysRev.49.324.

[27] A. Wallraff et al. "Approaching Unit Visibility for Control of a Superconducting Qubit with Dispersive Readout". In: *Physical Review Letters* 95 (Aug. 2005), p. 060501. DOI: 10.1103/PhysRevLett.95.060501.

[28] F. Motzoi et al. "Simple Pulses for Elimination of Leakage in Weakly Nonlinear Qubits". In: *Physical Review Letters* 103.11 (Sept. 2009). ISSN: 1079-7114. DOI: 10.1103/physrevlett.103.110501. URL: http://dx.doi.org/10.1103/PhysRevLett.103.110501.

[29] R Barends et al. "Superconducting quantum circuits at the surface code threshold for fault tolerance". en. In: *Nature* 508.7497 (Apr. 2014), pp. 500–503.

[30] Yvonne Y. Gao et al. *A practical guide for building superconducting quantum devices*. en. arXiv:2106.06173 [quant-ph]. Sept. 2021. DOI: 10.48550/arXiv.2106.06173. URL: http://arxiv.org/abs/2106.06173 (visited on 02/24/2025).

[31] Matthias Baur. "Realizing quantum gates and algorithms with three superconducting qubits". In: 2012.

[32] J. M. Gambetta et al. "Analytic control methods for high-fidelity unitary operations in a weakly nonlinear oscillator". In: *Physical Review A* 83.1 (Jan. 2011). ISSN: 1094-1622. DOI: 10.1103/physreva.83.012308. URL: http://dx.doi.org/10.1103/PhysRevA.83.012308.

[33] F. Motzoi and F. K. Wilhelm. "Improving frequency selection of driven pulses using derivative-based transition suppression". In: *Physical Review A* 88.6 (2013), p. 062318. DOI: 10.1103/PhysRevA.88.062318.

[34] J. M. Chow et al. "Optimized driving of superconducting artificial atoms for improved single-qubit gates". In: *Phys. Rev. A* 82 (4 Oct. 2010), p. 040305. DOI: 10.1103/PhysRevA.82.040305. URL: https://link.aps.org/doi/10.1103/PhysRevA.82.040305.

[35] Matthew Reed. *Entanglement and Quantum Error Correction with Superconducting Qubits*. 2013. arXiv: 1311.6759 [quant-ph]. URL: https://arxiv.org/abs/1311.6759.

[36] Sarah Sheldon et al. "Characterizing errors on qubit operations via iterative randomized benchmarking". In: *Phys. Rev. A* 93 (1 Jan. 2016), p. 012301. DOI: 10.1103/PhysRevA.93.012301. URL: https://link.aps.org/doi/10.1103/PhysRevA.93.012301.

[37] E. Knill et al. "Randomized Benchmarking of Quantum Gates". en. In: *Physical Review A* 77.1 (Jan. 2008). arXiv:0707.0963 [quant-ph], p. 012307. ISSN: 1050-2947, 1094-1622. DOI: 10.1103/PhysRevA.77.012307. URL: http://arxiv.org/abs/0707.0963 (visited on 02/24/2025).

[38] J. Kelly et al. "Optimal Quantum Control Using Randomized Benchmarking". en. In: *Physical Review Letters* 112.24 (June 2014), p. 240504. ISSN: 0031-9007, 1079-7114. DOI: 10.1103/PhysRevLett.112.240504. URL: https://link.aps.org/doi/10.1103/PhysRevLett.112.240504 (visited on 02/24/2025).

[39] M. Mohseni, A. T. Rezakhani, and D. A. Lidar. "Quantum-process tomography: Resource analysis of different strategies". In: *Phys. Rev. A* 77 (3 Mar. 2008), p. 032322. DOI: 10.1103/PhysRevA.77.032322. URL: https://link.aps.org/doi/10.1103/PhysRevA.77.032322.

[40] Antonio Anna Mele. "Introduction to Haar Measure Tools in Quantum Information". In: *Quantum* 8 (May 2024), p. 1340. ISSN: 2521-327X. DOI: 10.22331/q-2024-05-08-1340. URL: http://dx.doi.org/10.22331/q-2024-05-08-1340.

[41] Joseph Emerson, Robert Alicki, and Karol Życzkowski. "Scalable noise estimation with random unitary operators". In: *Journal of Optics B: Quantum and Semiclassical Optics* 7.10 (Sept. 2005), S347. DOI: 10.1088/1464-4266/7/10/021. URL: https://dx.doi.org/10.1088/1464-4266/7/10/021.

[42] Pauli Virtanen et al. "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python". In: *Nature Methods* 17 (2020), pp. 261–272. DOI: 10.1038/s41592-019-0686-2.

[43] J. A. Nelder and R. Mead. "A Simplex Method for Function Minimization". In: *The Computer Journal* 7.4 (Jan. 1965), pp. 308–313. ISSN: 0010-4620. DOI: 10.1093/comjnl/7.4.308. eprint: https://academic.oup.com/comjnl/article-pdf/7/4/308/1013182/7-4-308.pdf. URL: https://doi.org/10.1093/comjnl/7.4.308.

[44] Dieter Kraft. *A Software Package for Sequential Quadratic Programming*. Forschungsbericht DFVLR-FB 88-28. Institut für Dynamik der Flugsysteme, Oberpfaffenhofen. Köln, Germany: Deutsche Forschungs- und Versuchsanstalt für Luft- und Raumfahrt (DFVLR), 1988.

[45] Masahiro Nomura and Masashi Shibata. *cmaes : A Simple yet Practical Python Library for CMA-ES*. 2024. arXiv: 2402.01373 [cs.NE]. URL: https://arxiv.org/abs/2402.01373.

[46] Andrew N. Sloss and Steven Gustafson. *2019 Evolutionary Algorithms Review*. 2019. arXiv: 1906.08870 [cs.NE]. URL: https://arxiv.org/abs/1906.08870.

[47] Nikolaus Hansen. *CMA-ES/pycma: Python implementation of CMA-ES (Commit f0e8160)*. https://github.com/CMA-ES/pycma/blob/f0e8160eaf0175db90822b7c055f37e289495786/cma/options_parameters.py. Version f0e8160. Accessed: 2025-05-14. 2024.

[48] Nikolaus Hansen and Stefan Kern. "Evaluating the CMA evolution strategy on multimodal test functions". In: *International conference on parallel problem solving from nature*. Springer. 2004, pp. 282–291.

[49] Takuya Akiba et al. "Optuna: A Next-generation Hyperparameter Optimization Framework". In: *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 2019.

[50] Frank Hutter, Holger H. Hoos, and Kevin Leyton-Brown. "Sequential Model-Based Optimization for General Algorithm Configuration". In: *Learning and Intelligent Optimization*. Ed. by Carlos A. Coello Coello. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 507–523. ISBN: 978-3-642-25566-3.

[51] Bobak Shahriari et al. "Taking the Human Out of the Loop: A Review of Bayesian Optimization". In: *Proceedings of the IEEE* 104.1 (2016), pp. 148–175. DOI: 10.1109/JPROC.2015.2494218.

[52] David C. McKay et al. "Efficient Z-gates for quantum computing". In: *Physical Review A* 96.2 (Aug. 2017). ISSN: 2469-9934. DOI: 10.1103/physreva.96.022330. URL: http://dx.doi.org/10.1103/PhysRevA.96.022330.

[53] *QuantWare Soprano-D*. URL: https://www.quantware.com/product/soprano.

[54] R. Sagastizabal et al. “Experimental error mitigation via symmetry verification in a variational quantum eigensolver”. In: *Phys. Rev. A* 100 (1 July 2019), p. 010302. DOI: 10.1103/PhysRevA.100.010302. URL: https://link.aps.org/doi/10.1103/PhysRevA.100.010302.

[55] N. K. Langford et al. “Experimentally simulating the dynamics of quantum light and matter at deep-strong coupling”. In: *Nature Communications* 8.1 (Nov. 2017), p. 1715. ISSN: 2041-1723. DOI: 10.1038/s41467-017-01061-x. URL: https://doi.org/10.1038/s41467-017-01061-x.

[56] C. C. Bultink et al. “Protecting quantum entanglement from leakage and qubit errors via repetitive parity measurements”. In: *Science Advances* 6.12 (Mar. 2020). ISSN: 2375-2548. DOI: 10.1126/sciadv.aay3050. URL: http://dx.doi.org/10.1126/sciadv.aay3050.

[57] A. Rol M. et al. “Fast, High-Fidelity Conditional-Phase Gate Exploiting Leakage Interference in Weakly Anharmonic Superconducting Qubits”. In: *Phys. Rev. Lett.* 123.12 (2019), p. 120502. DOI: 10.1103/PhysRevLett.123.120502.

[58] Ruichao Ma et al. “A dissipatively stabilized Mott insulator of photons”. In: *Nature* 566.7742 (Feb. 2019), pp. 51–57. ISSN: 1476-4687. DOI: 10.1038/s41586-019-0897-9. URL: http://dx.doi.org/10.1038/s41586-019-0897-9.

[59] Jiansong Gao. “The Physics of Superconducting Microwave Resonators”. PhD thesis. California Institute of Technology, 2008. DOI: 10.7907/RAT0-VM75. URL: https://doi.org/10.7907/RAT0-VM75.

Acknowledgement

Desidero innanzitutto ringraziare il professor Carrazza per avermi offerto l'opportunità di svolgere questa tesi all'interno del progetto Qibo e per il supporto fornito durante tutte le fasi del lavoro.

Un sentito ringraziamento va anche ai miei correlatori per la guida e il costante supporto. La loro disponibilità e il loro approccio aperto sono stati fondamentali per affrontare con maggiore consapevolezza il percorso di ricerca.
Ringrazio Alessandro per l'aiuto e i consigli ricevuti, in particolare durante le settimane trascorse al TII. La sua attenzione agli aspetti umani del lavoro di ricerca e la sua disponibilità hanno rappresentato un punto di riferimento importante.
Ringrazio Edoardo per la costante disponibilità, in particolare nelle fasi iniziali del progetto, durante le quali mi ha supportata nella comprensione della struttura e del funzionamento di `qibocal` e `qibolab`. Il suo aiuto è stato essenziale per orientarmi all'interno del codice.
Ringrazio inoltre Andrea per la disponibilità e il supporto puntuale, che si sono rivelati particolarmente utili ogni volta che ho avuto bisogno di chiarire aspetti complessi o approfondire passaggi tecnici.

Ringrazio i miei genitori che hanno reso possibile questo percorso, senza opporsi alle scelte che non condividevano o non capivano ma fidandosi della mia capacità di scegliere ciò che era meglio per me. Soprattutto ringrazio mia mamma Raffaella da cui ho imparato a impegnarmi nel mio lavoro e fare del mio meglio, e ringrazio mio papà che mi ha insegnato a non spaventarmi di fronte a qualcosa che non capisco ma a farmi domande e cercare di capire in profondità il funzionamento di quello che ho davanti. Ovviamente ringrazio mia sorella Anna, la mia cincips, che mi ha sempre sostenuto, spronato ad andare avanti e spinta a continuare anche quando non sono stata bene, anche quando le cose sembravano non andare e mi ha dato la forza di continuare passo dopo passo, è bello averti accanto. Ti ringrazio anche per tutte le foto di Wendy e Mila che mi ha mandato quando io non ero a casa ma tu sì e che mi hanno sempre strappato un sorriso o una lacrima per la tenerezza.

Ringrazio la nonna Palmira che mi ha insegnato a perseguire i miei obiettivi con ostinazione senza mai rinunciare ad aiutare chi incontravo nel mio percorso, e che mi ha sempre ripetuto che sarebbe andato tutto bene, questa tesi è dedicata anche a te. Ringrazio la nonna Rina che mi ha sempre ripetuto . Ringrazio anche i nonni, il nonno Giuseppe e il nonno Mario che porto entrambi dentro di me, questo traguardo l'ho raggiunto anche grazie a voi che avete sempre fatto il massimo per la famiglia.

Ringrazio tutti i miei compagni di corso con cui ho condiviso il mio percorso fino a questo punto, è stato bello incontrarsi e poter condividere tanti momenti che hanno reso piacevole il tempo trascorso in università. Ringrazio Tima per tutti gli abbracci che mi ha regalato, Paolo per i le risate e il caffè la mattina, perché cominciare la giornata insieme rende tutto un po' più facile, Ale per la carica di passione e voglia di fare che mi trasmette ogni volta che facciamo una chiacchierata. Ringrazio le mie compagne storiche del gruppo del laboratorio, Rosa e Daniela perché cominciare questo percorso con voi ha decisamente alleggerito le difficoltà, e fatto sembrare affrontabili anche i momenti meno semplici. Ringrazio soprattutto Alessia, in questi cinque, ormai sei, anni in cui ci conosciamo la nostra amicizia si è evoluta moltissimo, sei stata un'ispirazione, con tutta la tua energia, i tuoi progetti, la tua positività e la tua simpatia, sei anche stata una spalla a cui mi sono potuta appoggiare ogni volta che ne ho avuto bisogno. Sono molto contenta di poter condividere con te tutti i miei pensieri e le mie idee sapendo che troverò sempre una persona pronta ad ascoltarmi e dirmi con onestà che cosa pensa senza mai giudicarmi.

Un ringraziamento speciale è anche per tutte le persone che ho incontrato durante quella *vacanza* meravigliosa che è stato il tirocinio al Fermilab, ognuno in modo diverso mi avete aiutato tutti a crescere. Sicuramente ringrazio Tom per tutti i consigli che mi ha sempre dato, Marco che non ha mai paura di condividere le sue idee, di cui è sempre stato aperto a discutere con me e che durante i due mesi trascorsi insieme ci ha coinvolti e trascinati tutti in gite bellissime. Ringrazio il mio compagno d'ufficio, Ste, in quei due mesi abbiamo condiviso tanto, dai gossippini alle paure e difficoltà, che fossero legate all'università o alla vita. Ringrazio moltissimo Sara e Silvia, grazie a voi ho sinceramente scoperto il valore di avere accanto due donne incredibili che ti sostengono e ti spronano a fare il tuo meglio, la nostra amicizia mi ha fatto scoprire quanto incredibile possa essere l'alleanza tra donne. Siete state le mie prime confidenti per qualsiasi problema io abbia avuto durante i due mesi in cui abbiamo vissuto insieme, siete state una sorpresa fantastica di cui non sapevo nemmeno di avere bisogno. Come ci siamo già dette molte volte, vivere insieme ad altre due sconosciute avrebbe potuto essere difficile e invece l'avete resa la cosa più bella e naturale che si possa immaginare, non vedo l'ora di poter vedere tutta la strada che sono sicura percorrerete, ognuna nel suo ambito.

Ringrazio tutte le persone che non ho nominato ma che mi sono state accanto, il supporto di tutte è stato fondamentale!