Numerical optimization of quantum gates on IBMQ systems

Master thesis

Santana Apolonio Lujan
Matriculation Number 0059144

Conducted at the
German Space Operations Center (GSOC) of the German Aerospace Center (DLR)
Oberpfaffenhofen
and the
University of Applied Sciences Munich (HM)
Department of Computer Science and Mathematics

Supervisors:
Prof. Dr. S. Tornow (HM)
Dr. A. Spörl (DLR)
Dr. N. Pomplun (DLR)

Auditors:
Prof. Dr. S. Tornow
Prof. Dr. A. Zugenmaier

Hochschule München
University of Applied Sciences

Fakultät für Informatik
und Mathematik

Die folgende Erklärung ist in jedes Exemplar der Masterarbeit einzubinden und jeweils persönlich zu unterschreiben.

Lujan, Santana Apolonio	München, 30.01.2023
(Familienname, Vorname)	(Ort, Datum)
14.11.1991	IG / WS 2022
(Geburtsdatum)	(Studiengruppe / WS/SS)

Erklärung
Gemäß § 40 Abs. 1 i. V. m. § 31 Abs. 7 RaPO

Hiermit erkläre ich, dass ich die Masterarbeit selbständig verfasst, noch nicht anderweitig für Prüfungszwecke vorgelegt, keine anderen als die angegebenen Quellen oder Hilfsmittel benützt sowie wörtliche und sinngemäße Zitate als solche gekennzeichnet habe.

(Unterschrift)

Abstract

As the number of qubits in modern quantum computers grows, the demand shifts to high-fidelity quantum gates which complement the large number of qubits to enable deep quantum circuits with low error rates. The prime goal of this thesis is to exploit the tools provided by Optimal Control Theory (OCT) to numerically optimize single qubit quantum gates for exemplary quantum backends based on superconducting qubits provided by IBM Quantum. This was realized by designing and implementing a modular software framework able to generate pulse level controls for a desired unitary evolution independent of the target hardware or the optimization algorithm. A reference implementation of the Gradient Ascent Pulse Engineering (GRAPE) algorithm in conjunction with a self-implemented closed system simulator optimize piecewise constant amplitudes for given gate durations. We interface the hardware with the implemented IBMQ-Pulse-Adapter, which translate the optimization results to executable experiments. The framework includes automated qubit spectroscopy experiments to characterize the intrinsic parameters of the target qubits required for optimizing tailored pulse sequences. Experiments with optimized pulse sequences of multiple durations are concluded for the X-Gate and the H-Gate on 7–qubit IBM Quantum backends based on Transmon technology. The experiment results are analyzed and compared with the default gates provided by IBM Quantum, yielding comparable performances in terms of fidelity at the cost of longer gate times.

Acknowledgements

I want to thank Prof. Dr. S. Tornow for supervising this thesis and for providing the scientific problem statement, as well as the access to the IBM Quantum systems.

Great appreciation also goes to Dr. N. Pomplun and Dr. A. Spörl for supporting me in both technical and general aspects concerning this thesis, as well as for introducing me to the wonderful field of control theory.

Most importantly, I must thank my love Julia for supporting me on every single step of this journey.

The project/research is part of the Munich Quantum Valley (K8), which is supported by the Bavarian state government with funds from the Hightech Agenda Bayern Plus.

Contents

Acronyms 1

List of Figures 2

1. Motivation and overview 5

2. Fundamentals 7
2.1. Formalism . . . 7
2.2. Linear algebra . . . 9
2.2.1. Hilbert space . . . 9
2.2.2. Operators . . . 9
2.2.3. Superoperators . . . 11
2.2.4. Lie groups and algebra . . . 11
2.3. Quantum mechanics . . . 12
2.3.1. Mathematical tools for describing quantum systems . . . 12
2.3.2. Quantum channels . . . 15
2.4. Superconducting quantum computers . . . 16
2.4.1. Transmon qubits . . . 17
2.4.2. Single Transmon Hamiltonian . . . 18
2.4.3. Multi Transmon Hamiltonian . . . 19

3. Quantum optimal control 21
3.1. Controllability . . . 21
3.2. Advantages . . . 21
3.3. GRAPE algorithm . . . 22
3.3.1. Choosing ϵ . . . 24
3.4. Evaluating the quality on real hardware . . . 26
3.4.1. Quantum State Tomography . . . 27
3.4.2. Quantum Process Tomography . . . 28

4. Software 30
4.1. Requirements . . . 30
4.2. Architecture . . . 31
4.3. GRAPE algorithm module . . . 31
4.4. Closed system simulator backend . . . 32
4.5. IBMQ Pulse Adapter . . . 32
4.5.1. OpenPulse . . . 33
4.5.2. Qiskit Pulse . . . 33

4.5.3. Translation of control amplitudes 36

5. Experiments **39**
5.1. Qubit characterization 39
5.1.1. Determining the qubit frequency 39
5.1.2. Calibrating a π-Pulse 41
5.1.3. Correcting the qubit frequency value 42
5.1.4. Determining the Rabi frequency Ω_R 43
5.2. Single Qubit Gates 48

6. Conclusion **57**
6.1. Summary 57
6.2. Discussion and outlook 57

7. References **59**

Appendix **65**
A. UML Diagrams 66
B. Qubit drive amplitude duration relation diagrams 69

Acronyms

API Application Programming Interface. 33

AWG Arbitrary Waveform Generator. 33

cQED Circuit Quantum Electrodynamics. 18

DRAG Derivative Removal by Adiabatic Gate. 35, 36

GRAPE Gradient Ascent Pulse Engineering. 3, 5, 21–24, 30–32, 57

MLE Maximum Likelihood Estimation. 28

NISQ Noisy Intermediate Scale Quantum. 5, 16

NMR Nuclear Magnetic Resonance. 5, 18, 22, 31

OCT Optimal Control Theory. 3, 5, 57

QASM Quantum Assembly Language. 33

QPT Quantum Process Tomography. 27, 28, 48, 57, 58

QST Quantum State Tomography. 2, 27, 28, 57, 58

List of Figures

2.1. Overview on the main mathematical representations of completely positive maps and how one can switch between them. From [17] 15
2.2. **(a)** LC-circuit for quantum harmonic oscillator (QHO) **(b)** Corresponding energy diagram, energy spacings between the stationary states are all $\hbar\omega_r$. **(c)** Circuit for quantum anharmonic oscillator with coil replaced by a Josephson junction **(d)** The non-linear inductance reshapes the energy diagram to have non-equidistant spacing between the states, enabling isolation of the computational subspace (qubit). Image taken from [21] . 17
2.3. Schematic of a capacitive coupling between two Transmon qubits (left and right) with a coupler in form of a linear resonator (center) [21]. 20

3.1. Schematic representation of a single set of control amplitudes. Each amplitude $u_k(j)$ is applied for a fixed time $\Delta t = \frac{T}{N}$ and each timestep indexed by j. Arrows pointing in vertical directions visualize the gradient $\frac{\delta\Phi}{\delta u_k(j)}$, the direction indicating how each amplitude should be modified in the next iteration to increase the quality. Taken from [35] 23
3.2. Development of ϵ (red trace) and quality Φ (blue trace) during optimization of a pulse sequence implementing a σ_X-Gate 26
3.3. Block diagram of a QST. Preparation of ρ_{out} 27

4.1. Schematic representation of the signal modulation. The envelope (left) is multiplied with a carrier signal (center) resulting in modulated output signal (right). Taken from [49] 34
4.2. Summary of the measurement output levels [46] 35
4.3. Overview on the parametrized pulses provided by Qiskit Pulse. All pulses shown have a duration of 300 dt and amplitude of 0.15. 35
4.4. Final control amplitudes for optimization of a Hadamard unitary with a duration of $T \approx 100$ ns. Control 0 corresponds to the u_1-amplitudes driving a σ_X-rotation, while Control 1 represents the u_2-amplitudes for a σ_Y-rotation. 37

5.1. Result of a Frequency Sweep experiment executed on the backend *ibm_perth* for qubit 0 for different pulse shapes. 40
5.2. Rabi experiment 41

5.3. Ramsey experiment sequence: **(a)** Initial state $|0\rangle$ **(b)** State after application of the $\frac{\pi}{2}$–pulse **(c)** Free evolution with frequency $\Delta = \omega_q - \omega_d$ for some time t_{delay} **(d)** Final state $|0\rangle$ after second application of $\frac{\pi}{2}$–pulse for the exemplary case a phase of $\varphi = \frac{\pi}{2}$ has been accumulated. **(e)** Pulse schedule for a single Ramsey circuit 42
5.4. Results of the Ramsey experiment for different detuning frequencies δ with the precession time t_{delay} ranging from 0 ns to 1000 ns. The experiment was concluded on qubit 0 of *ibm_perth*. 43
5.5. Multiple Rabi experiments with varying pulse durations on qubit 0 of *ibmq_guadalupe*. Each experiment was conducted with a different amplitude ranging from $u_{min} = 0.02$ to $u_{max} = 1.0$. 45
5.6. Results of the Rabi experiment with varying duration for amplitudes $0 \leq u \leq 3.1$ executed on qubit 0 on *ibmq_guadalupe*. Image **(a)** shows an inverse linear relation between amplitude and pulse duration for a 2π–pulse. The corresponding frequency in **(b)** consequentially shows an approximately linear relation. Figure **(c)** shows the calculated Rabi frequency Ω_R via (5.1). We utilize the median (dashed line) to eliminate outliers. 46
5.7. Comparison of the performance of π–pulse sequences executed on qubit 0 on backend *ibmq_guadalupe* and in the simulated unitary evolution. The Gaussian experiments show two additional data points, as the extracted pulse durations for these amplitudes are shorter than allowed. 47
5.8. Qubit arrangement and coupling on *ibm_perth* and *ibm_lagos*. 48
5.9. **(a), (b)** Optimized pulses sequences with different initial shapes for implementing a X-Gate. We observe minimal deviations between the optimized and initial shapes. **(c)** Provided X-Gate for qubit 3 on *ibm_perth*. 51
5.10. Simulated evolution of the expectation value $\langle \sigma_Z \rangle$ for pulses of different durations with initial constant shape implementing a X-Gate. 52
5.11. Simulated evolution of the expectation value $\langle \sigma_Z \rangle$ for pulses of different durations with initial Gaussian shape implementing a X-Gate. 53
5.12. Comparison of the error $(1 - \mathcal{F}_{pro})$ of the provided X-Gate (blue) with the optimized pulse sequences of initial constant (orange) and Gaussian shaped (green) samples. Simulation of the optimized pulses yielded error values $\leq 10^{-5}$ and are therefore not shown. The duration corresponding to the pulse with the lowest error is shown on top of each bar. The experiments were executed on *ibm_perth*. 54
5.13. Simulated evolution of the expectation values σ_Z for H-Gate implemented by the optimized pulse sequence of Figure 5.15a for initial states $|0\rangle$ and $|1\rangle$. The amplitudes u_1 and u_2 drive rotations about the x- and the y-axis respectively. 54

5.14. Comparison of the error $(1 - \mathcal{F}_{pro})$ of the provided H-Gate (blue) with the optimized pulse sequences with initial samples of constant amplitude (orange). The duration corresponding to the pulse with the lowest error is shown on top of each bar. Simulation of the optimized pulses yielded error values $\leq 10^{-5}$. The experiments were executed on *ibm_lagos*. 55
5.15. Optimized pulse sequence and provided H-Gate for qubit 2 on *ibm_lagos*. 56

1. GRAPE flowchart 66
2. Software Architecture 67
3. Grape Algorithm Class Diagram 68
4. Periods of sinus fits extracted from single Rabi experiments with varying duration and constant amplitude per experiment. Each data point represents a single duration for which a drive of the given amplitude results in a 2π rotation about the x-axis of the Bloch sphere. 69
5. Frequencies $f_{2\pi} = \frac{1}{t_{2\pi}}$ calculated from data in Figure 4 70
6. Rabi frequency calculated from data in Figure 4 via (5.1) 71

1. Motivation and overview

The invention of binary computing machines during the 20th century [1] revolutionized the way recurring arithmetic problems were solved. Empowered by a bi-yearly linear growth of computational power predicted by Gordon Moore in 1964 [2], the most powerful supercomputer in the present time is able to calculate over 1 million floating point operations per second [3]. To achieve such a performance, these systems are composed of multiple interconnected systems, each limited in terms of their computational power by the size of their electrical components on the integrated circuits which eventually cannot be further reduced. Even the strongest computers to date are not able to solve problems involving larger number of particles in quantum systems due to the exponential increase of the degrees of freedom for each additional particle. For this reason, Richard Feynman pioneered the idea to utilize quantum systems to simulate quantum mechanical behavior [4] in 1982. About 30 years later, Noisy Intermediate Scale Quantum (NISQ)-computers are not only available to researchers in a laboratory, but also to the public [5] over the cloud. The road to quantum supremacy is still far, as the currently available quantum computers are prone to noise and errors from a variety of sources.

One source of errors are quantum operations, also known as quantum gates, which every quantum algorithm is composed of. Depending on the algorithm, a quantum circuit can be composed of hundreds of gates, each single one possibly causing errors which propagate and multiply throughout the program flow. The individual error rate strongly depends on the physical implementation of the qubits and operations as well as the number of qubits it operates on. Common interaction techniques use pulse sequences, e.g. radiofrequency pulses on spin-based qubits used in NMR [6], laser pulses on neutral atoms in optical lattices [7] or microwave pulses on superconducting qubits [8]. One approach to reduce the errors induced by these operations is the idea to utilize proven concepts from the field of OCT and adapt the notions to the quantum realm. In the course of this thesis we will investigate and utilize a number of these concepts with the goal to generate and apply optimized piecewise constant pulse sequences to superconducting qubits on systems provided by IBM Quantum.

The thesis introduces the fundamental concepts of linear algebra, quantum mechanics, and superconducting qubits required for the subsequent chapters in Chapter 2. In Chapter 3 we proceed to elaborate on the general ideas and requirements for utilizing OCT on quantum systems, specifically the theoretical foundation of the implemented GRAPE algorithm as well as techniques used to extract comparable results. Chapter 4 presents an overview on the implemented software framework and the Qiskit Pulse library used to translate the pulse sequences into executable jobs. Next, we explore the IBM Quantum systems in Chapter 5 demonstrating the automated qubit spectroscopy experiments followed by experimentally comparing the performance of the optimized

pulse sequences for the X- and the H-Gate with the provided default gates on all qubits of a 7–qubit backend. The thesis is concluded with a discussion on possible improvements and ideas for further experiments and follow-up research topics in Chapter 6.

2. Fundamentals

2.1. Formalism

This section aims to give an overview on the formalism used in this thesis. As most of the notations and mathematical formalism should be familiar to the experienced reader, this chapter will only present a brief recap without going into much detail.

Dirac Notation

Dirac- or "bra-ket"-notation, is a notation which is often utilized for doing calculations with quantum states. In its essence, it is just a simple way of representing column- and row-vectors providing an elegant way of depicting common actions quantum mechanics.

Let v be a vector of the complex m-dimensional vector space, so $v \in \mathbb{C}^m$. A column vector is denoted by the *ket* symbol:

$$|v\rangle = \begin{bmatrix} v_0 \\ v_1 \\ \vdots \\ v_m \end{bmatrix}, \tag{2.1}$$

Row vectors are denoted by a *bra*:

$$\langle v| = \begin{bmatrix} v_0^* & v_1^* & \cdots & v_m^* \end{bmatrix} \tag{2.2}$$

Transforming a *ket* into a *bra* and vice versa is done by complex conjugating the entries and transposing the vector (conjugate transpose):

$$|v\rangle^\dagger = \begin{bmatrix} v_0^* \\ v_1^* \\ \vdots \\ v_m^* \end{bmatrix}^T = \begin{bmatrix} v_0^* & v_1^* & \cdots & v_m^* \end{bmatrix} = \langle v| \tag{2.3}$$

These are just the basics rules, further progression into advanced topics of this thesis will solidify the upsides of the notation.

Pauli operators and basis

A commonly used set of operators are the Pauli operators, which in this work will be identified by a σ_i symbol with $i \in \{X, Y, Z\}$ or sometimes also just the letter. They correspond to the following matrices:

$$\sigma_X = X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \quad \sigma_Y = Y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix} \quad \sigma_Z = Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \tag{2.4}$$

Together with the identity matrix

$$\sigma_I = \sigma_0 = I = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{2.5}$$

they form a complete operator basis $P = \{\sigma_I, \sigma_X, \sigma_Y, \sigma_Z\}$. Allowing real coefficients, the basis P spans the set of all $\mathbb{C}^{2\times 2}$ Hermitian matrices. This basis set can be easily expanded to the n-qubit case:

$$P_n = \{\sigma_I, \sigma_X, \sigma_Y, \sigma_Z\}^{\otimes n}, \quad span(P_n) = \mathbb{C}^{2^n \times 2^n} \tag{2.6}$$

The explicit formulation of the Pauli basis describing all operations on a two qubit system therefore is:

$$P_2 = \{\sigma_I\sigma_I, \sigma_I\sigma_X, \sigma_I\sigma_Y, \sigma_I\sigma_Z, \sigma_X\sigma_I, \sigma_X\sigma_X, \ldots, \sigma_Z\sigma_Y, \sigma_Z\sigma_Z\} \tag{2.7}$$

Ladder operators

For discussing quantum theories involving multiple particles or more than two stationary states, a pair of operators have been introduced. The *creation* or *raising* operator

$$a^\dagger = \sigma_+ = \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix} = \frac{1}{2}\left(\sigma_X - i\sigma_Y\right) \tag{2.8}$$

and the *annihilation* or *lowering* operator

$$a = \sigma_- = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} = \frac{1}{2}\left(\sigma_X + i\sigma_Y\right), \tag{2.9}$$

collectively known as the *ladder operators*, are used as an elegant mathematical description of adding or reducing energy in a quantum system.

Product operators

When referring about operators acting on composite systems we make use of a shorthand notation. Suppose we have a 3–qubit system, product operators are abbreviated similar to the following example:

$$\sigma_Z^{(0)} \otimes \sigma_X^{(1)} \otimes \sigma_Y^{(2)} = \sigma_Z\sigma_X\sigma_Y = ZXY \tag{2.10}$$

2.2. Linear algebra

2.2.1. Hilbert space

A vector space V over the complex numbers $\mathbb{C}$ is called a *Hilbert Space*, if it defines a *scalar product* or *inner product* on two elements $(\cdot,\cdot) : V \times V \to \mathbb{C}$ which is complete with respect to the norm $||\cdot|| := \sqrt{(\cdot,\cdot)}$ [9]. We will only be dealing with finite dimensional Hilbert Spaces, that is Hilbert Spaces of some fixed dimension n, therefore $V \in \mathbb{C}^n$.

Given two vectors $|v\rangle, |w\rangle \in V$ the *inner product* $\langle v|w\rangle$ quantifies a measure the overlap of the two vectors. For a pair of orthogonal vectors the inner product consequentially equals 0:

$$\langle 0|1\rangle = \begin{bmatrix}1 & 0\end{bmatrix} \begin{bmatrix}0\\1\end{bmatrix} = 0 \cdot 1 + 1 \cdot 0 = 0 \tag{2.11}$$

The *outer product* of $|v\rangle$ and $|w\rangle$ is commonly called a projector and results in a matrix with dimension $m \times m$. An example of a projector is the σ_+ operator:

$$|1\rangle\langle 0| = \begin{bmatrix}0\\1\end{bmatrix} \begin{bmatrix}1 & 0\end{bmatrix} = \sigma_+ \tag{2.12}$$

And finally the *tensor product* of v and w, also known as the *Kronecker* or *outer product* :

$$|v\rangle \otimes |w\rangle = |vw\rangle = \begin{bmatrix} v_0 \begin{bmatrix} w_0 \\ \vdots \\ w_{n-1} \end{bmatrix} \\ v_1 \begin{bmatrix} w_0 \\ \vdots \\ w_{n-1} \end{bmatrix} \\ \vdots \\ v_{n-1} \begin{bmatrix} w_0 \\ \vdots \\ w_{n-1} \end{bmatrix} \end{bmatrix} = \begin{bmatrix} v_0 w_0 \\ v_0 w_1 \\ v_1 w_0 \\ v_1 w_1 \\ \cdots \\ v_{n-1} w_{n-1} \end{bmatrix} \tag{2.13}$$

The outer product describes a composition of Hilbert spaces, combining the spaces of the two factors, resulting in a Hilbert space of dimension $dim(|v\rangle) \times dim(|w\rangle)$.

2.2.2. Operators

Operators are linear transformations, mapping elements from one vector space V to another vector space W. They are normally denoted with a hat on top $\hat{\cdot}$, although the hat is oftentimes omitted for simplicity. Even tough we will omit a vast majority, few special types of operators are extensively used in the field of quantum mechanics and quantum computing, justifying a review of their attributes.

Every operator can also be represented by a matrix M. Common matrix operations such as the *complex conjugation* $M \mapsto M^*$, the *transposition* $M \in \mathbb{C}^{k\times l} \mapsto M^T \in \mathbb{C}^{l\times k}$ and the *adjoint* $M \mapsto M^\dagger = (M^*)^T$ can therefore be applied.

An operator $\hat{U}$ is called *unitary* if its adjoint is also its inverse:

$$\hat{U}^{-1} = \hat{U}^\dagger \quad \hat{U}\hat{U}^\dagger = \mathbb{1} \tag{2.14}$$

The eigenvalues of unitary operators are of the form $e^{i\theta}$, thus preserving the norm of the vector it acts on. This property induces a geometrical interpretation of unitaries in finite-dimensional Hilbert spaces, as they can be understood as rotations, leaving the length of the vectors acted on unchanged. As we will see later, this property has widespread implications for the application in quantum mechanics.

Hermitian operators are another type of essential operators in quantum mechanics. An operator $\hat{A}$ is labeled Hermitian if it is self-adjoint, that is:

$$\hat{A} = \hat{A}^\dagger \tag{2.15}$$

After analyzing the eigenvalues of these operators, one finds them all to be real valued, a fact that entails great physical relevance. Operators representing physically measurable entities, such as momentum, energy or position must have real expectation values and must therefore be Hermitian. Calculating the expectation value of some measurable entity $\hat{A}$ for some state $|\psi\rangle$ is done by:

$$\left\langle \hat{A} \right\rangle = \frac{\langle\psi|\, \hat{A}\, |\psi\rangle}{\langle\psi|\psi\rangle} \tag{2.16}$$

Operators are said to be *skew-Hermitian* or *anti-Hermitian* if:

$$\hat{A}^\dagger = -\hat{A} \tag{2.17}$$

A frequently taken measure of a square operator $\hat{A} \in \mathbb{C}^{n\times n}$ is the *trace*, a sum over the diagonal entries:

$$tr(\hat{A}) = \sum_{i=1}^{n} a_{ii} \tag{2.18}$$

Using the trace, the *Frobenius inner product* can be calculated. This operation takes two operators and returns a scalar:

$$\left\langle \hat{A}|\hat{B} \right\rangle = tr\left(\hat{A}^\dagger B\right) \tag{2.19}$$

The commutator of two operators $\hat{A}, \hat{B}$ plays an important role in quantum mechanics and is defined by:

$$\left[\hat{A}, \hat{B}\right] = \hat{A}\hat{B} - \hat{B}\hat{A} \tag{2.20}$$

In case $\left[\hat{A}, \hat{B}\right] = 0 \Rightarrow \hat{A}\hat{B} = \hat{B}\hat{A}$ we say that $\hat{A}$ and $\hat{B}$ *commute*, implying that the order in which the operators are applied can be exchanged.

2.2.3. Superoperators

Superoperators are linear operators acting on a vector space of linear operators [10]. To represent superoperators, we first have to introduce the notion of *vectorization* of a matrix. Vectorizing a matrix can be done by either row-stacking or column-stacking the matrix columns or rows on top of each other. We will focus on the column-stacking method which transforms a matrix of dimension $m \times n$ to a $1 \times mn$-vector [11], which we will denote as the *vec*-operation. Given some operator $\hat{M}$ with a matrix representation $M \in \mathbb{C}^{m \times n}$ with rows $\overrightarrow{M_m}$ and columns $\overrightarrow{M_n}$, the vectorization operation yields:

$$vec(M) = \begin{bmatrix} \overrightarrow{M_0} \\ \overrightarrow{M_1} \\ \vdots \\ \overrightarrow{M_{n-1}} \end{bmatrix} \tag{2.21}$$

Instead of $vec(M)$, the notations $|M\rangle$ and $\hat{\hat{M}}$[12] as well as $|M\rangle\rangle$[13] are commonly used for specifying superoperators.

2.2.4. Lie groups and algebra

Group Theory is an important apparatus used all over mathematics studying the algebraic structures known as groups. A group consists of a set of elements $\mathcal{G} = \{G_1, G_2, \cdots G_n\}$ and an operation $G_j \circ G_k$ combining any two elements of the group to produce another element of the group. In every group, there exists a single identity element I, leaving the element G_n of the set unchanged $G_n \circ I = I \circ G_n = G_n$ when the operation is applied. Groups require that every element G_n has an element G_n^{-1} such that $G_n \circ G_n^{-1} = G_n^{-1} \circ G_n = I$ called the inverse element. Lastly, a group requires associativity, that is for all elements $G_n, G_j, G_k \in \mathcal{G}$ the following has to hold $(G_N \circ G_J) \circ G_K = G_N \circ (G_J \circ G_K)$.

In the extent of this thesis we are only interested in a single type of groups, the *Matrix groups* with regular matrix multiplication as the operation of choice. The proof that group conditions hold for matrix groups will not be given here, but can be found in any decent algebra textbook, such as [14].

A descriptive mathematical tool linking operators used in quantum mechanics to geometrical interpretations are the *Lie groups*, a family of continuous matrix groups. The most general group of linear $n \times n$ invertible matrices is denoted by $GL(n, \mathbb{C})$ with group elements G. Any continuous closed subgroup of the general linear group is a Lie group [15], for example:

- the **unitary** subgroup: $U(n, \mathbb{C})$ with $G \in GL(n)$ and $G^\dagger = G^{-1}$
- the **special unitary** subgroup: $SU(n, \mathbb{C})$ with $G \in U(n)$, $G^\dagger = G^{-1}$ and $det(G) = 1$

The characteristic property of the elements in the unitary groups is the length preservation when applied to a vector $v \in \mathbb{C}^n$.

Every element U of a Lie group can be represented by a matrix exponential

$$U = e^{\beta x} \tag{2.22}$$

where β corresponds to some rotational angle and x is the *generator* of the group element. The entirety of generators $\{x_k\}$ is called the *Lie algebra* and is used to determine the operators needed to construct a desired unitary target operator. Lie algebras are named according to their Lie group but with lowercase letters. All elements of the unitary algebra $x \in u(N)$ are skew-Hermitian $x = -x^\dagger$, elements of the special unitary $x \in su(N)$ algebra are additionally traceless $tr(x) = 0$.

2.3. Quantum mechanics

The behavior of systems at a sub-molecular level vastly differs from the classical intuition we as humans gain by experiencing and exploring classical macroscopic phenomenons in our day-to-day life. By continuously looking closer, increasing our insight on properties of the microscopic world which defy our natural understanding, a new type of intuition and appreciation for the small-scale events surrounding can be attained. Duality of particles and waves can explain interference effects, which otherwise would be impossible to understand. As a result of treating particles as waves, inherent properties of wave-like systems such as the uncertainty relation between canonically conjugate variables, e.g. position x and momentum p, have to be taken into consideration.

Open questions in the early decades of the last century, such as the reason to why electrons do not fall into the nucleus, can be explained as a consequence of these particles behaving like waves, leading to a quantization of properties ought to be continuous.

Quantum mechanics provides a set of postulates, a framework, allowing the formulation of specific quantum theories solving specific problems in their fields. This section provides a summary on the relevant postulates required for this thesis.

2.3.1. Mathematical tools for describing quantum systems

Descriptions of quantum mechanical systems can be divided into two types. *Closed quantum systems* describe completely isolated systems with no environmental influence, conserving their energy and therefore not losing any information as time goes on. They represent noiseless, perfectly self-contained settings, thus removing the fragility of real world quantum states and are a perfect starting point for grasping the basic rules of quantum mechanics. *Open quantum systems* on the other hand describe real world systems more closely as it is never possible to perfectly isolate a quantum system from the outside. They characterize and specify effects such as dissipation, which lead to information loss of the system to the surrounding environment.

A state, by definition, encodes all the information we have about a system under observation, independently of the physical rules it obeys. These rules could be of classical nature, such as Newtonian or Hamiltonian mechanics, but could also be quantum mechanical. Contrary to classical theories, in which the outcome of each single measurement

can be predicted with certainty, quantum theories and quantum states are of probabilistic nature, consequently only predicting the *probabilities* of certain measurement outcomes.

Referring to a *pure state* implies precise knowledge, without any degree of uncertainty, about the state itself. The most general form describing a pure state for a single two-level system

$$|\psi\rangle = a\,|0\rangle + b\,|1\rangle \tag{2.23}$$

is a linear combination of two basis states, in this case

$$|0\rangle = \begin{bmatrix}1\\0\end{bmatrix} \quad |1\rangle = \begin{bmatrix}0\\1\end{bmatrix} \tag{2.24}$$

with complex coefficients $a, b \in \mathbb{C}$. Provided that the elements are orthonormal, we can choose any set of vectors as a basis. A set of basis vectors that equivalently describes the state in (2.23) could therefore be $\{\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle), \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)\}$. These coefficients can be interpreted as probability amplitudes, resulting in the corresponding probability densities after taking the modulus squared. The probability for the state in (2.23) to be in either $|0\rangle$ or $|1\rangle$ is then:

$$P(|0\rangle) = |\langle 0|\psi\rangle|^2 = |a|^2 \quad P(|1\rangle) = |\langle 1|\psi\rangle|^2 = |b|^2 \tag{2.25}$$

Pure quantum states must comply with the *normalization condition*, meaning that all probability densities must add up to 1:

$$|a|^2 + |b|^2 = 1 \quad \langle\psi|\psi\rangle = 1 \tag{2.26}$$

An example of a commonly used, non-trivial quantum state is the equal superposition of the basis states $|\psi\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$, where to probability of finding it in either of the basis states is 0.5.

In real systems for example, we cannot be sure if the desired initial state can consistently be prepared with complete confidence. This kind of state, an ensemble or statistical mixture of multiple pure states, is called a *mixed state* and is described by a density operator

$$\hat{\rho} = \sum_i p_i\,|\psi_i\rangle\,\langle\psi_i| \tag{2.27}$$

$$\sum_i p_i = tr(\hat{\rho}) = 1 \tag{2.28}$$

It is important to keep in mind, the lack of knowledge leading to the formulation of states as density operators is not due to any quantum mechanic traits, but is of plain classical nature.

In general, *measurement* in quantum mechanics refers to the process of determining a single numerical real value from a quantum state. We are mostly dealing with *projective measurements*, which are a special case of quantum measurements with some practical properties. More background on general measurements and other specialized measurement techniques such as Positive Operator-Valued Measures (POVM) can be found in [16]. A projective measurement is done by projecting a state $|\psi\rangle$ on one of the eigenstates with

eigenvalue m of an observable $\hat{M}$, where $\hat{M}$ is a Hermitian operator with orthogonal projectors $\hat{P}_m$ [16]:

$$\hat{M} = \sum_m e\hat{P}_m \tag{2.29}$$

Recall that the Hermitian condition implies real eigenvalues of the operator, the measurement result m is therefore a real scalar number. The probability of measuring a certain value m can then be calculated by:

$$p(m) = \langle\psi| \hat{P}_m |\psi\rangle \tag{2.30}$$

Expectation values of Hermitian operators are calculated as shown in (2.16). The wording "measuring in a certain basis" consequently refers to multiple projective measurements using a specific set of observables.

The *time evolution* of a closed quantum system is governed by the Schrödinger equation [16]:

$$i\hbar\frac{d}{dt}|\psi\rangle = \hat{H}|\psi\rangle \tag{2.31}$$

where $\hbar$ is Planck's constant and the Hermitian operator $\hat{H}$ represents the *Hamiltonian* of the system. The Hamiltonian is a special operator which describes the dynamics of a closed system completely. A description of the operator propagating a quantum state for a time t is then given as a solution to the Schrödinger equation with a time-independent Hamiltonian $\hat{H}$:

$$\hat{U}(t) = e^{-i\hat{H}t} \tag{2.32}$$

From this equation it is fairly obvious, that the dynamic of a quantum system is entirely dependent on the Hamiltonian operator. This naturally results in a large interest for finding Hamiltonians that properly describe the characteristics of systems of interest. Assuming the Hamiltonian is a Hermitian operator, the time evolution propagator $\hat{U}$ is always unitary guaranteeing trace preservation and therefore reversibility. Given a unitary operator $\hat{U}$, a pure state $|\psi\rangle$ then evolves according to:

$$|\psi\rangle = \hat{U}|\psi_0\rangle \tag{2.33}$$

The time evolution of a mixed state is described by [16]:

$$\rho = \sum_i p_i |\psi_i\rangle\langle\psi_i| \xrightarrow{\hat{U}} \sum_i p_i \hat{U}|\psi_i\rangle\langle\psi_i|\hat{U}^\dagger = \hat{U}\rho\hat{U}^\dagger \tag{2.34}$$

As previously mentioned, propagation of a state by a unitary can always be reversed by utilizing the fact that the inverse of an unitary operator is its adjoint $U^{-1} = U^\dagger$:

$$|\psi_0\rangle = \hat{U}^\dagger|\psi\rangle \quad \rho_0 = \hat{U}^\dagger\rho\hat{U} \tag{2.35}$$

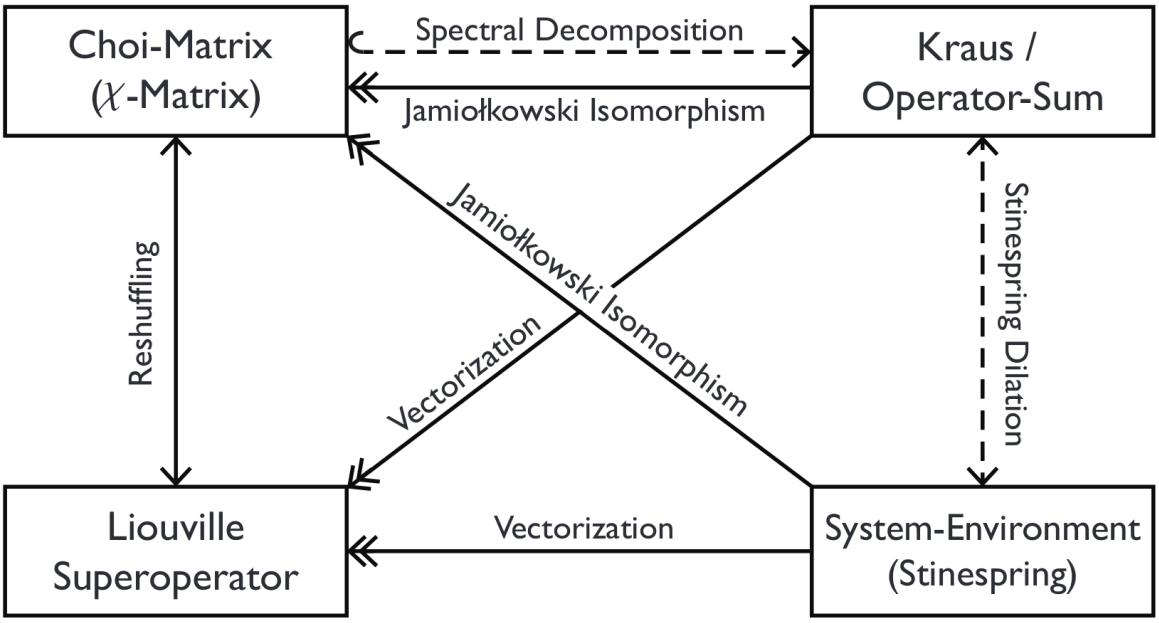


Figure 2.1.: Overview on the main mathematical representations of completely positive maps and how one can switch between them. From [17]

2.3.2. Quantum channels

For describing the complete evolution of a real quantum system, we require a more general approach than the one described by unitary propagation. This motivates a more abstract view, which includes the previous descriptions of state-to-state mappings but also describes the actions of mapping operators to other operators.

Such a mapping is called a quantum channel $\mathcal{E}$ and is described by a completely positive (CP) linear map, mapping one complex vector space to another. In this thesis we are mainly concerned with quantum channels which map input density operators to output density operators as seen in (2.34) [16]:

$$\rho = \mathcal{E}(\rho_0) \tag{2.36}$$

The quantum channel is the most general operation which maps a density operator to another density operator [17].

Completely positive maps and therefore quantum channels can be described by a number of interchangeable mathematical representations, each highlighting different aspects of the channel. The choice of the used formalism depends on the properties under investigation, an overview of the existing representations can be found in Figure 2.1 One can elegantly move from one representation to another using tensor networks in combination with a type of graphical calculus as shown in [13].

During the course of this thesis, our main motivation for using quantum channels is the comparison of ideal channels with imperfect channels implemented by pulse sequences on real quantum hardware. We make use of two of the representation types, the superoperator

and the Choi– or χ–Matrix representations. The former allows us to calculate a fidelity measure between the ideal and the implemented channel, while the latter can be used to visualize the effects of a quantum channel in terms of well-understood operators. We continue with a derivation of the χ–matrix as well as a description on how to convert it to the superoperator formalism which is used in Section 3.4.2. Although these calculations are not carried out by hand, but with the help of the `qiskit.quantum_info` library, understanding the underlying concepts support the interpretation of the final results.

A quantum channel can be interpreted as a sum of operations E_i applied on a given density operator ρ via (2.34):

$$\mathcal{E}(\rho) = \sum_i E_i \rho E_i^\dagger \tag{2.37}$$

The applied operators E_i can be expressed in any arbitrary basis set $\tilde{E}_i$ and can therefore be written as a linear combination of the basis operators [16]:

$$E_i = \sum_m e_{i,m} \tilde{E}_m \tag{2.38}$$

with complex coefficients $e_{i,m}$. After inserting this back into (2.37) we receive [16]:

$$\mathcal{E}(\rho) = \sum_{m,n} \chi_{mn} \tilde{E}_m \rho \tilde{E}_n^\dagger \tag{2.39}$$

$$\chi_{mn} \equiv \sum_i e_{im} e_{in}^* \tag{2.40}$$

The is often referred to as the *process matrix* or just χ–matrix and can be interpreted as complex weights, controlling the individual contributions of the basis operators. For intuitively interpretable results, it has prevailed to use the Pauli basis, consisting of the three Pauli matrices in addition to the identity matrix. A quantum channel under question can consequently be completely determined by the χ–matrix. Converting to the superoperator formalism is done by applying the bipartite reshuffling operation [13], which refers to the column stacking *vec*-operation from (2.21).

The interested reader can refer to [13] for a broader overview on quantum channels and their representations.

2.4. Superconducting quantum computers

Quantum computers built from superconducting circuit elements experienced a tremendous rise in popularity ever since the first introduction of qubit-like behavior of a Cooper-pair box coupled with a Josephson junction in 1999 [18]. More than two decades later, quantum processors based on superconducting technology have become popular platforms for NISQ algorithms and experiments, while also sustaining development to reach higher fault tolerance as well as a larger number of qubits. IBM has announced a 433-qubit device based on superconducting technology [19] and so called *Transmon* qubits, a technology which is also employed by Google in their Sycamore quantum processor [20]

This section aims to give an overview on the main idea behind *Transmon* qubits, from now on just referred to as *Transmons*, and how we can interact with them.

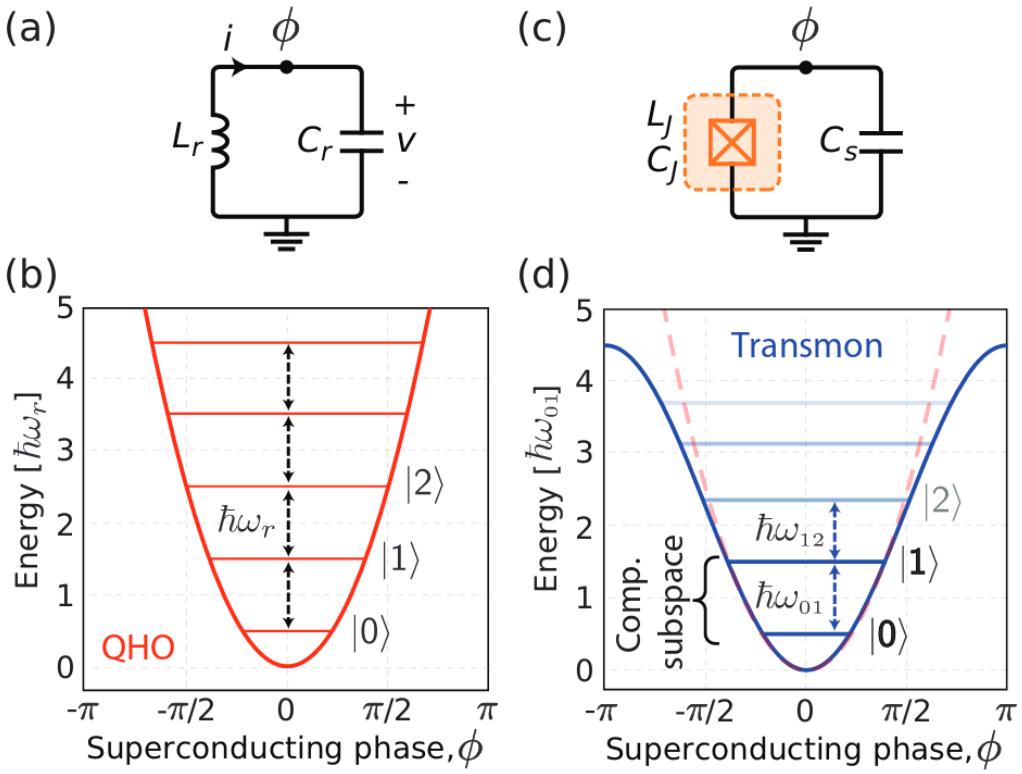


Figure 2.2.: **(a)** LC-circuit for quantum harmonic oscillator (QHO) **(b)** Corresponding energy diagram, energy spacings between the stationary states are all $\hbar\omega_r$. **(c)** Circuit for quantum anharmonic oscillator with coil replaced by a Josephson junction **(d)** The non-linear inductance reshapes the energy diagram to have non-equidistant spacing between the states, enabling isolation of the computational subspace (qubit). Image taken from [21]

2.4.1. Transmon qubits

At heart, the Transmon can be understood as an anharmonic oscillator with energy levels split by discrete, non-equidistant spacings. These energy levels E_n are the eigenvalues of the Hamiltonian operator $\hat{H}$ of the system with the corresponding eigenstates $|\psi_n\rangle$ which are referred to as *stationary states*.

$$\hat{H}\,|\psi_n\rangle = E_n\,|\psi_n\rangle \tag{2.41}$$

The term *anharmonic oscillator* itself can be split into two defining traits, where oscillator pertains to the fact that the total energy in the system oscillates between kinetic and potential energy. An analogue classical counterpart to this is a LC-Circuit, fluctuating kinetic (electrical) energy stored in the capacitor to potential (magnetic) energy stored in the inductor. Just like with classical oscillators, a characteristic frequency ω that is dependent on the dimensions of the circuit elements is defined.

In fact, the Transmon is a quantized version of the LC-circuit replacing the coil with a non-linear inductance component. This component, commonly known as a *Josephson*

junction, plays the key role to uniquely identify two states of the system enabling the usage as a qubit. It introduces an *anharmonicity* in the spacings between the energy levels of the stationary states, effectively reducing the chance of unwanted transitions to non-qubit states. This effect is also known as *state-leakage* and reducing its occurrence is one concern when trying to design high fidelity controls for quantum computing.

The circuit shown in Figure 2.2 depicts only a single realization type of possible Transmon designs, namely the *Fixed-frequency*-Transmon. Other designs, for example *Flux-tunable qubits*, empower the user to tune the transition frequencies ω during runtime at the cost of more noise sensitivity.

In the course of this thesis we will only be concerned with Fixed-frequency-Transmons, as these are the specific type deployed by IBM in their quantum computing hardware. These type of qubits have a fixed transition frequency, determined at manufacturing time by the choice of the circuit element dimensions. As we will see in Section 5.1, it is important to design the single qubit circuits in a way that avoids qubits in proximity to have similar transition frequencies. If one wants to learn more about this topic, references [22] and [23] are a good starting point.

2.4.2. Single Transmon Hamiltonian

The climb of superconducting qubits was accompanied by the need for a theory predicting the dynamics of such systems in response to excitation and perturbation. This theory, known as *Circuit Quantum Electrodynamics* (cQED), provides effective Hamiltonian models for light-matter interaction, enabling the capability to facilitate microwaves accomplishing coherent quantum operations. From this point forward, hats on the operators are omitted for simplicity. Additionally, we will set $\hbar = 1$ and absorb it into the Hamiltonian by dividing by $\hbar$. The static Hamiltonian of a single Transmon in absence of a drive is given by [21]

$$H_0 = \omega_q a^\dagger a + \frac{\alpha}{2} a^\dagger a^\dagger a a \tag{2.42}$$

with α as the anharmonicity and $a^\dagger, a$ as creation and annihilation operators. The frequency $w_q = w_{0\to1}$ multiplied with $\hbar$ specifies the energy difference between the ground and excited states, taking a similar role as the Larmor frequency in NMR. It is therefore sometimes also called the *qubit frequency*.

If one takes only the two computational states into account, the Hamiltonian further simplifies:

$$H_0 = \frac{\omega_q}{2} \sigma_Z \tag{2.43}$$

Obviously, this simplification constitutes an approximation neglecting the effects of possible state leakage. This fact should be kept in mind, especially when planning to apply control techniques to real systems.

Driving the Transmon

We can manipulate the state of a Transmon by inducing controlled amounts of energy to the system. Inducing energy can be done by applying radiation, where the extent

of energy is given by the frequency of the wave. The amount of energy needed for a transition is given by the energy spacing (see Figure 2.2). Typical transition frequencies for Transmons are around 5 GHz and are therefore considered as microwaves. Waves are characterized by their frequency ω_d, a phase offset ϕ_d and a time dependent amplitude $u(t)$ and can be described by the following drive Hamiltonian [23]:

$$H_d = u(t)\left(a^\dagger e^{-i(\omega_d t-\phi_d)} + a e^{i(\omega_d t+\phi_d)}\right) \tag{2.44}$$

Taking the static Hamiltonian H_0 into account and applying the qubit approximation we receive:

$$H = H_0 + H_d = \frac{\omega_q}{2}\sigma_Z + u(t)\Omega_R\left[\cos(\omega_d t+\phi_d)\sigma_X + \sin(\omega_d t+\phi_d)\sigma_Y\right] \tag{2.45}$$

Finally, entering a frame rotating at drive frequency ω_d the Hamiltonian can be written down in an incredibly descriptive form [23]:

$$H = \frac{\omega_q-\omega_d}{2}\sigma_Z + u(t)\frac{\Omega_R}{2}\left[\cos(\phi_d)\sigma_X + \sin(\phi_d)\sigma_Y\right] \tag{2.46}$$

It is easy to see from the first term, that for resonant drives at $\omega_d = \omega_q$ the rotation axis on the Bloch sphere is either X or Y depending only on the phase ϕ_d. A resonant drive with phase $\phi_d = 0$ therefore corresponds to a rotation around the X-axis, whereas a phase $\phi_d = \frac{\pi}{2}$ generates a rotation around the Y-axis. The rate of the generated rotation is described by the Rabi frequency Ω_R.

To summarize, the available knobs for controlling a Transmon qubit are the drive amplitude $u(t)$, the frequency ω_d and the phase ϕ_d. Transition frequency ω_q and Rabi frequency Ω_R are system defining parameters and have to be determined experimentally. A description how this is done can be found in Section 5.1.

2.4.3. Multi Transmon Hamiltonian

Coupling multiple Transmons can be achieved in various ways. Again, we will be focusing on the coupling used in IBM systems, which is of the type called *capacity coupling with a coupler* [21]. Physically, this coupling is realized via resonator busses with microwave cavities taking the role of resonators (Figure 2.3). A single Transmon coupled to a resonator can be analytically described by the well-known Jaynes-Cummings Hamiltonian [24]

$$H = \omega_r a^\dagger a + \omega_q\frac{\sigma_Z}{2} + g(a\sigma_+ + a^\dagger\sigma_-) \tag{2.47}$$

with ω_r as the resonator frequency, ω_q as the qubit frequency and g as the coupling strength. Without going further into detail, a properly detuned two qubit system using the resonator as mediator of a J coupling can be described by [23]:

$$H = \omega_{q0}\frac{\sigma_Z I}{2} + \omega_{q1}\frac{I\sigma_Z}{2} + J\left(\sigma_+\sigma_- + \sigma_-\sigma_+\right)^1 \tag{2.48}$$

$$= \omega_{q0}\frac{\sigma_Z I}{2} + \omega_{q1}\frac{I\sigma_Z}{2} + \frac{J}{2}\left(\sigma_X\sigma_X + \sigma_Y\sigma_Y\right) \tag{2.49}$$

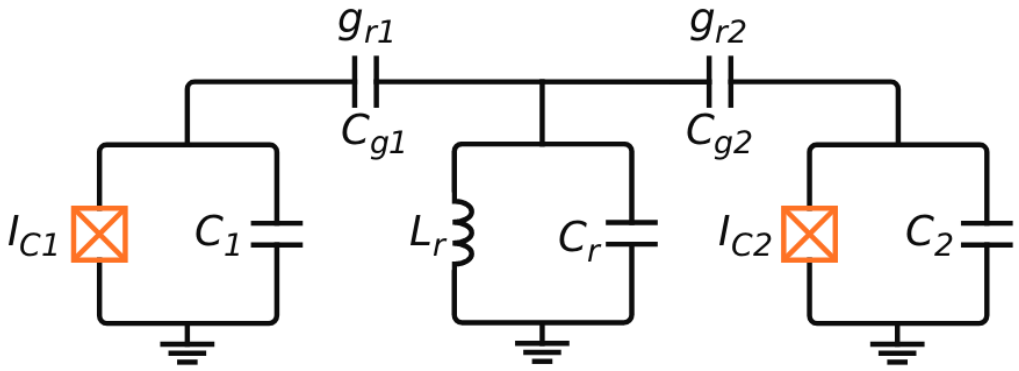


Figure 2.3.: Schematic of a capacitive coupling between two Transmon qubits (left and right) with a coupler in form of a linear resonator (center) [21].

Creating entanglement can be done with a technique known as *cross-resonance* [25]. The main idea of the cross-resonance interaction is to drive the control qubit at the frequency of the target qubit. This introduces a $\sigma_Z\sigma_X$ interaction which is one component necessary to create a CNOT-Gate. When defining qubit 0 as the control and qubit 1 as the target, the effective cross-resonance Hamiltonian for the drive takes on the form [25]

$$H_d = u(t)\cos(\tilde{\omega}_1 t)\left(\sigma_X I - \frac{J}{\Delta_{01}}\sigma_Z\sigma_X + m_{12} I\sigma_X\right) \tag{2.50}$$

with $u(t)$ as the drive amplitude, $\tilde{\omega}_1 = \omega_1 - \frac{J}{\Delta_{01}}$ as the dressed target qubit frequency, Δ_{01} as the qubit-qubit detuning and m_{01} representing a spurious crosstalk due to stray electromagnetic coupling.

[1] Inserting (2.8) and (2.9) with $^{(0)}$,$^{(1)}$ referring to the qubit indices:

$$\begin{aligned}
&\sigma_+^{(0)}\sigma_-^{(1)} + \sigma_-^{(0)}\sigma_+^{(1)} = \frac{1}{4}\left(\sigma_X^{(0)} - i\sigma_Y^{(0)}\right)\left(\sigma_X^{(1)} + i\sigma_Y^{(1)}\right) + \frac{1}{4}\left(\sigma_X^{(0)} + i\sigma_Y^{(0)}\right)\left(\sigma_X^{(1)} - i\sigma_Y^{(1)}\right) \\
&= \frac{1}{4}\left(\sigma_X^{(0)}\sigma_X^{(1)} + i\sigma_X^{(0)}\sigma_Y^{(1)} - i\sigma_Y^{(0)}\sigma_X^{(1)} + \sigma_Y^{0}\sigma_Y^{1} + \sigma_X^{(0)}\sigma_X^{(1)} - i\sigma_X^{(0)}\sigma_Y^{(1)} + i\sigma_Y^{(0)}\sigma_X^{(1)} + \sigma_Y^{(0)}\sigma_Y^{(1)}\right) \\
&= \frac{1}{4}\left(2\sigma_X^{(0)}\sigma_X^{(1)} + 2\sigma_Y^{(0)\sigma_Y^{(1)}}\right) = \frac{1}{2}\left(\sigma_X^{(0)}\sigma_X^{(1)} + \sigma_Y^{(0)}\sigma_Y^{(1)}\right)
\end{aligned}$$

3. Quantum optimal control

Initial applications of quantum theories gave insight on phenomenons on a microscopic scale, leading to groundbreaking inventions such as the lasers and transistors, the latter lighting the spark for designing digital calculating machines (now known as computers). Many years after the initial introduction of quantum mechanics, innovative fabrication techniques and other technological advances enable us to effectively isolate single quantum systems. Controlling these systems, utilizing their quantum abilities for novel algorithms and calculations is sometimes referred to as the *Second Quantum Revolution*[26].

This chapter briefly discusses quantum optimal control, its prerequisites and advantages. Afterwards we describe a specific application of the theory, the GRAPE algorithm, which is able to generate numerically optimized control sequences for, but not only, quantum systems.

3.1. Controllability

Loosely speaking, full control of a quantum system refers to the ability to generate all unitary propagators required to drive an initial unitary operator $U = \mathbb{1}$ to any unitary target operator U_{target} desired. This property is labeled fully *operator controllable* [27] and can be efficiently formulated and validated with the help of Lie algebras.

A N-dimensional quantum system is *operator controllable*, if the drift and control terms in its Hamiltonian span the complete Lie algebra $su(N)$ after repeated commutator constructions [12]. This criterion can be efficiently mathematically validated for small systems, however with growing dimensionality other methods utilizing graph theory should be considered [28].

A valid Lie algebra $su(2)$, offering full operator controllability in the case of one qubit, therefore generating the entirety of the $SU(N)$ with $N := 2^n$ and $n = 1$, is the set of Pauli matrices $\{\sigma_X, \sigma_Y, \sigma_Z\}$.

3.2. Advantages

By utilizing optimal control, several system specific characteristics and constraints can be modelled into the control protocol, effectively fitting the generated controls to the target system. Additional advantages, leveraging the capabilities of optimal control are listed below. It should be noted that this is by no means a complete list, since this is a rapidly evolving area research ([29], [30], [31], [32]), many ideas and protocols are currently under investigation.

- Time-optimal: Find the shortest pulse sequences implementing the target unitary
- Relaxation-optimal: Modelling relaxation effects into the equation of motion to take dissipation into account
- Adding robustness against inhomogenities, for example discrepancies in the applied frequencies of the control fields due to non-perfect hardware
- Taking into account restraints, such as power or amplitude limitations
- Counteract free evolution of the drift Hamiltonian or always-on coupling

3.3. GRAPE algorithm

Originally emerging from laser spectroscopy ([33], [34]), the Gradient Ascent Pulse Engineering (GRAPE) algorithm has proven versatile at dealing with a wide variety of problems. It has been successfully applied to generate NMR pulse sequences for optimal control of coupled spin systems [35], but is not restricted to the quantum domain [36]. In the extent of this thesis, we will only be discussing the application of the algorithm on quantum systems in the absence of relaxation using the Schrödinger equation as the rule for time evolution. Nevertheless, one can consider adding dissipative effects by exchanging the underlying equation of motion with the Lindbladian master equation [37].

The main objective of the GRAPE algorithm is to optimize fixed-step piecewise constant control amplitudes $u(t)$ from some initial values $u_{init}(t)$ to an optimal solution $u_{optimal}(t)$, which will steer a system from the initial state at time $t = 0$ to the desired state at $t = T$. In our case, time evolution of a quantum system is entirely described by a unitary operator U_{target} (see (2.33), (2.34)). The ultimate goal of the algorithm can therefore be quantified as generating a unitary $U(T)$ that matches the target as closely as possible. One measure of closeness is the normalized overlap between the target and the result further referred to as *quality* Φ

$$\Phi = \frac{1}{d} \operatorname{Re} \langle U_{target} | U(T) \rangle = \frac{1}{d} \operatorname{Re} \left\{ \operatorname{tr} \left(U_{target}^{\dagger} U(T) \right) \right\} \tag{3.1}$$

with the dimension d of the unitaries as a normalization factor. This measure of closeness is sensitive to differences in the global phases of target and therefore differentiate between some unitary $U = e^{i\varphi} U_{target}$ and $U = U_{target}$ for $\varphi > 0$. A phase insensitive variant of the quality calculation can be found in [35]. Optimizing this scalar up to a certain precision threshold, in the ideal case $\Phi = 1$, is the main objective of the GRAPE algorithm.

The overall unitary operator or propagator is generated from a sequence of N equidistantly spaced timesteps j [12]:

$$U(T) = U_N \cdots U_j \cdots U_1 U_0 = e^{-iH_N \Delta t} \cdots e^{-iH_j \Delta t} \cdots e^{-iH_1 \Delta t} \tag{3.2}$$

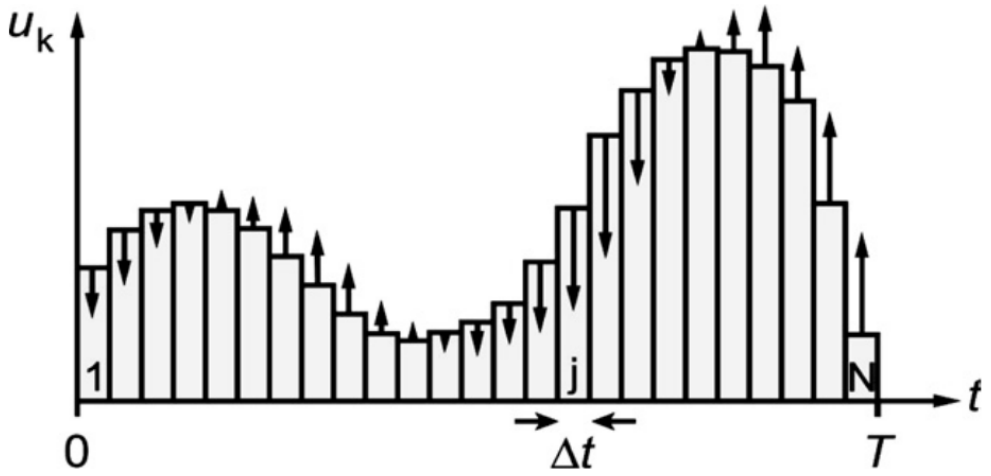


Figure 3.1.: Schematic representation of a single set of control amplitudes. Each amplitude $u_k(j)$ is applied for a fixed time $\Delta t = \frac{T}{N}$ and each timestep indexed by j. Arrows pointing in vertical directions visualize the gradient $\frac{\delta\Phi}{\delta u_k(j)}$, the direction indicating how each amplitude should be modified in the next iteration to increase the quality. Taken from [35]

Each individual unitary acts for a discrete duration $\Delta t = \frac{T}{N} = t_j - t_{j-1}$ and is generated by inserting the piecewise constant Hamiltonian H_j at each step

$$H_j = \underbrace{H_0}_{\text{drift}} + \underbrace{\sum_k u_k(j) H_k}_{\text{controls}} \tag{3.3}$$

into (2.32). The drift or static term H_0 is system specific, neither control- nor adaptable and therefore has to be treated as a given. In contrast, the control terms H_k define our influence on the system. Provided that the Hamiltonian enables full operator controllability and given a minimal implementation time, the timestep dependent control amplitudes $u_k(j)$ can be adjusted such that the system can be steered into the appropriate direction. We therefore optimize over a total number of $k \times j$ parameters. An infinite number of possible pulse sequences exist for implementing a certain target unitary, hence the final solution found by the GRAPE algorithm is just one of potentially many solutions. The specific solution that is found depends strongly on the chosen initial amplitudes [35].

Every iteration, each of the control amplitudes is corrected based on calculating first-order gradients with respect to the quality(Figure 3.1), in turn pointing to a local maximum of the optimization landscape.

For determining the impact of small perturbations to the control amplitudes, in effect gaining knowledge of the "correct" direction (the direction in which the quality increases), we first propagate the initial unitary $U_0 = \mathbb{1}$ for every timestep $j \leq N$ and store the results:

$$U(j) = U_j \cdots U_1 U_0 \tag{3.4}$$

Next we propagate our target U_{target} with the adjoint unitaries generated by the time reserved control amplitudes, again storing the results which will now be denoted P_j:

$$P_j = U_{j+1}^{\dagger} \cdots U_N^{\dagger} U_{target} \tag{3.5}$$

The first-order gradient for every control amplitude is then calculated:

$$\frac{\delta \Phi}{\delta u_k(j)} = -\, Re \left\{ tr \left[P_j^{\dagger} \left(i \Delta t H_k U(j) \right) \right] \right\} \tag{3.6}$$

To conclude the iteration, the control amplitudes $u_k(j)$ can now be updated in dependence of some step size ϵ:

$$u_k(j) \rightarrow u_k(j) + \epsilon \frac{\delta \Phi}{\delta u_k(j)} \tag{3.7}$$

The basic procedure of the GRAPE algorithm is visualized in Figure 1 in form of a flowchart.

After the desired quality $\Phi \approx 1$ has been attained, our final result is a set of k vectors, one for every control term in the Hamiltonian, with j control amplitudes. In Section 5.2 we use this algorithm to optimize pulse sequences of differing durations for specific unitary gates, which are then applied to real hardware.

3.3.1. Choosing ϵ

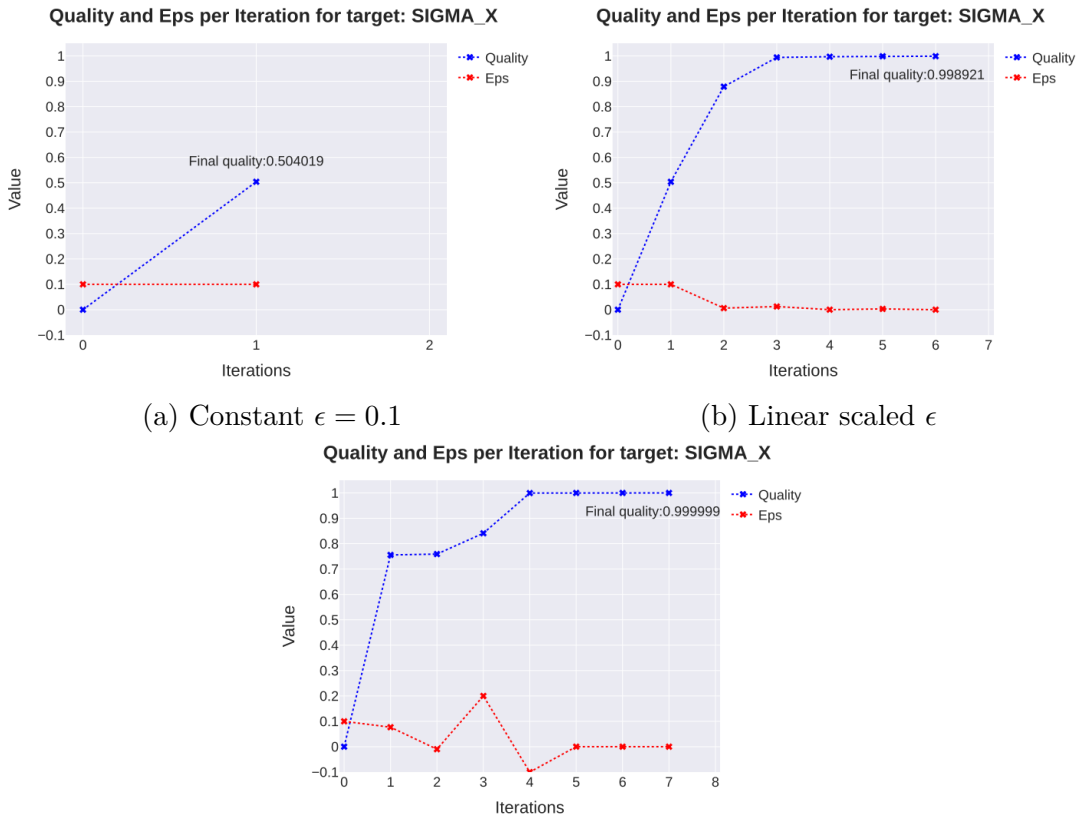
Choosing an elaborate ϵ increases the chances and speed of convergence to high quality values. Three approaches have been implemented and asserted:

Constant Always use a fixed step size that is defined at configuration time. If the updated control amplitudes do not lead to a quality increase, the algorithm is finished and the previous highest quality is returned. The application of this method leads to unsatisfying results for most of the non-trivial optimization tasks, either caused by choosing a step size which is too large, in term causing the optimization to terminate early as overshooting the target often leads to a decreased quality. On the contrary, a very small step size reduces the chance of overshooting but can be very computationally expensive as it increases the number of iterations required to converge. The main drawback of this approach is not taking the optimization landscape into account at all.

Linear Scaling The control amplitudes are updated with an initial step size defined in the configuration. Afterwards the new propagator is calculated followed by a quality measure. In case of a quality increase, the next iteration is started. Otherwise, the step size is multiplied by a scaling factor in the range $0 < f_{scale} < 1$ with typical values around $f_{scale} = 0.5$. As we converge closer to the maximum, the distance decreases and the step size therefore has to shrink as well. The main advantage of this method is the robustness against early terminations, coming at the cost of a higher number of quality calculations each iteration. Application of this method leads to significantly better results, approaching the target quality more closely and reducing the number of early terminations.

Parabola fit This method models the quality as a function of ϵ, so $f(\epsilon) = \Phi$. This approach based on the assumption, that the quality function takes on a parabolic shape near the optimum. Three points are needed to fit a parabola, we therefore use the current quality value $f(\epsilon_0 = 0)$ and two supplementary data points, gained by evaluating $f(\epsilon_1)$ and $f(\epsilon_2)$ with $\epsilon_0 < \epsilon_1 < \epsilon_2$. The peak point with coordinates $(x = \epsilon_{peak}, y = f(\epsilon_{peak}))$ of the fitted parabola function can be calculated, providing another conformed assumption for a good ϵ value. Depending on the deviation of the fitted peak quality from the actual calculated quality $f(\epsilon_{peak})$, this point is either included in the comparison between the resulting quality values or discarded. Again, if none of the evaluated step sizes result in a quality gain the initial ϵ is scaled done by multiplying a configurable factor.

Figure 3.2 clearly shows, the desired quality values near one can only be achieved with an adaptive choice of ϵ. Using the parabola fit method returns the best results at the price of calculating three more propagations of the system which can become costly, depending on its dimension. It should be noted that it is possible to parallelize each of these propagations, negating this downside if enough computational resources are available.



(a) Constant $\epsilon = 0.1$

(b) Linear scaled ϵ

(c) ϵ from parabola fit

Figure 3.2.: Development of ϵ (red trace) and quality Φ (blue trace) during optimization of a pulse sequence implementing a σ_X-Gate

3.4. Evaluating the quality on real hardware

The previous section discussed how we can optimize and evaluate the quality of piecewise constant pulse sequences in a mathematical simulation of a quantum system. In Section 5.2 these sequences will be executed on real systems, requiring methods to calculate a quality factor from measurement results. This section discloses a family of techniques to construct comparable quality measures from the repeated application of the optimized sequences.

The application of a pulse sequence can be described as the actions of a quantum channel $\mathcal{E}$ (Section 2.3.2) on some input state ρ_{in}, yielding an output state ρ_{out}. Our optimization target defines the actions of the ideal desired quantum channel and therefore acts as the baseline comparison for a quality measure. This comparison can only be achieved if the quantum channel resulting from the applied pulse sequences is well identified. Due to the projective nature of measurement, a complete characterization of a quantum channel or state is not possible from a single measurement. A family of tomography experiments can be used to fully reconstruct a quantum state or channel. As the name suggests, the gist of these experiments is taking multiple images (measurements)

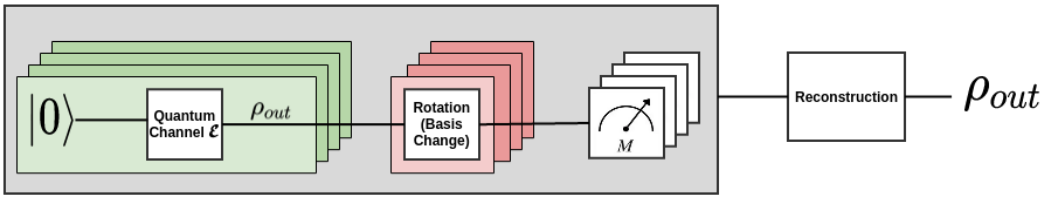


Figure 3.3.: Block diagram of a QST. Preparation of ρ_{out} (green) is identical each time, whereas the post-rotation (red) changes with the choice of measurement basis. Naturally the number of shots per circuit must be large enough gather meaningful probability distributions, which are then used to reconstruct the state. Adapted from [38]

from different angles (basis) and reconstructing a complete description of the channel afterwards. The reconstructed quantum channel can then be compared against the ideal target, resulting in a single scalar value which measures how well the applied pulse sequences perform. As the tomography experiments require to measure the resulting state from multiple angles, the ability to reliably apply the quantum channel under question to the system is a hard requirement.

If one wishes to only reconstruct a resulting quantum state from a single input state after the channel has been applied, the *Quantum State Tomography* (QST) experiment is sufficient. In contrast, reconstructing the full quantum channel has to be done with the *Quantum Process Tomography* (QPT). As the QPT is an extension of the Quantum State Tomography to multiple input states, we will shortly introduce key aspects of the QST below. Afterwards we will proceed to the QPT, which is the experiment that was ultimately applied to gain the result data.

3.4.1. Quantum State Tomography

The concern of a QST is the reconstruction of an unknown density operator ρ_{out} from repeated measurements and the resulting counts. As previously mentioned, a reliable preparation of the unknown state required to execute the experiment. Generally, we assume that the state needs to be prepared an infinite amount of times, tough in practice it only has to be prepared n times, where n is the number of observables needed to fully reconstruct the state.

The chosen set of observables P is arbitrary, barring the requirement of being tomographically complete [17]. This means that P needs to span an operator basis on the Hilbert space the state lives in. In other words, the operators have to be able to fully describe our unknown quantum state ρ_{out}. Hence, our set will contain at least d^2 elements where d is the dimension of the Hilbert space it has to span. The most frequently used basis set, due to its well-known nature, is the Pauli basis (see Section 2.1). Because measurements on IBM systems are always in the σ_Z basis, we have to apply post-rotations depending on the observable that is to be measured. After preparing the unknown state and applying the required post-rotation we acquire a single measurement

outcome. To extract a meaningful probability distribution, this process is run for 2048 shots. We repeat this for every observable, gathering one probability distributions for each observable which are used to reconstruct the final state.

There are many techniques for performing the reconstruction, *Maximum Likelihood Estimation* (MLE) being one of the most popular approach used in practice [17], [39]. For our use case it is sufficient to understand the main procedure of QST as well as the inputs and outputs (Figure 3.3). If one urges a deeper understanding of the topic, the references mentioned in this chapter are a good starting point.

We now determine the ideal target state ψ_{ideal} via time evolution of the initial, assumedly pure state, by the target unitary U_{target}:

$$|\psi_{ideal}\rangle = U_{target}\,|0\rangle \tag{3.8}$$

A measurement of closeness between the ideal and the output state, commonly known as *state fidelity*, is calculated via [40]:

$$\mathcal{F}_{state}\left(\rho_{out}, |\psi_{ideal}\rangle\langle\psi_{ideal}|\right) = \langle\psi_{ideal}|\,\rho_{out}\,|\psi_{ideal}\rangle \tag{3.9}$$

With this result, we can assess the quality of the applied quantum channel to a single input state at the cost of $|P|$ additional circuits. Striving for a more universal quality metric, one has to consider preparing multiple different input states, leading us to the QPT.

3.4.2. Quantum Process Tomography

The main goal of the QPT is to reconstruct a complete description of an arbitrary quantum channel. Full characterization of some quantum channel $\mathcal{E}$ requires a tomographically complete set of measurement operators P in addition to a tomographically complete set of input states Q[17]. After preparation of an input state $\rho_j \in Q$ through pre-rotations, it is sent through the unknown quantum channel $\mathcal{E}$. Each resulting state $\rho_{out,j} = \mathcal{E}(\rho_j)$ is then analyzed in a QST experiment (Section 3.4.1) using P as the set of measurement operators. After the acquisition of $|Q|$ output states, the full quantum channel is reconstructed. Details of the reconstruction, as well as alternative methods can be found in [39], [41], [13]. The result of the reconstruction is a χ–matrix as described in (2.39). From this result, we can calculate the *process fidelity* between a quantum channel $\mathcal{E}$ and a unitary U [42], which is done with the help of the `qiskit.quantum_info.process_fidelity` function:

$$\mathcal{F}_{pro}\left(\mathcal{E}, U\right) = \frac{Tr\left(\hat{\hat{U}}^{\dagger}\hat{\hat{\mathcal{E}}}\right)}{d^2} \tag{3.10}$$

$\hat{\hat{U}}^{\dagger}\hat{\hat{\mathcal{E}}}$ are the superoperator version of the ideal unitary and the reconstructed quantum channel with d as the dimension of the channel.

For performing a QPT, the total number of circuits that need to be prepared and executed is given by the product of the number of input states $|Q|$ and the number of elements in the measurement basis $|P|$. The application of the QPT is not feasible for

large systems, as the required amount of input states and measurement operators to satisfy the tomographical completeness each scale exponentially with the dimension of the quantum system[13]. During the course of this thesis, we are only dealing with single qubit systems and can therefore apply the tomography techniques without running into resource limitations. For future applications or moving onto larger systems, alternative approaches such as randomized benchmarking should be taking into consideration [43].

4. Software

One of the goals of this thesis is the design and implementation of a software architecture, which allows the usage of different quantum optimal control algorithms on a multitude of backends. This section first introduces the main ideas and design decisions in the implemented software, proceeding to a description of the reference implementation for the GRAPE algorithm and the closed system simulator backend. Lastly, the adapter bridging the gap between optimization output and IBM hardware is illustrated.

4.1. Requirements

Software projects are always accompanied by a set of requirements, specifying the boundaries and obligations. Main use case of the developed software is scientific research, the formulation of the requirements is therefore not as stringent as one would expect from commercial software projects which are shipped to customers. Nevertheless, key aspects such as reusability and maintainability of the design and the resulting code are always key properties a software developer should strive for.

Following we will use the term *backend* for describing a system on which an optimal control algorithm can run on. This can either be a simulation, for example simulated quantum system, or a real physical system. The framework must meet the following requirements:

- Modular design, enabling easy extension by
  - new optimal control algorithms
  - new backends
- Integration of the existing software
- Reference implementation of a quantum control algorithm (GRAPE)
- Quantum closed system simulator backend which can be used for GRAPE
- Experimental setup enabling experiments on real IBM Quantum systems

The programming language chosen for implementing the proposed design is the *Python* language. This choice is largely motivated by the availability of most popular quantum computing libraries and interfaces, as well as the ease of usage and lack of compilation.

4.2. Architecture

A schematic overview on the implemented design can be found in Figure 2. The diagram depicts a high-level view, emphasizing the key usage and aggregation relationships of the modules, in turn leaving concrete implementation details open. Following is an overview of the main design pieces and their responsibilities.

Algorithms Every algorithm that shall be integrated must provide at least a *Backend-Interface*, specifying concrete methods to be implemented by the backends. Each backend supporting this algorithm inherits from the interface and is therefore forced to implement the required functionality. The backends can then be type checked against the corresponding interface class to figure out if a certain algorithm is supported.

Backends Specific implementations for the required functionality of the algorithms. Can support a multitude of algorithms by inheriting its interface. Depending on the type of backend, either simulation or hardware, additional functionality needs to be supplied by the hardware adapters.

Hardware Adapter Provides the actual interface to the hardware. Depending on the hardware this could mean for example interfacing a proprietary driver (NMR) or establishing and maintaining connection with cloud bases services (IBM).

Experiments Entry points for multiple different experiments. Specific experiment configuration parameters are parsed from the command line or configuration files. These classes essentially build experimental pipelines, using the previously mentioned modules as building blocks, connecting the output of one module to another modules input.

Common utilities Single thematically grouped functions useful in many circumstances, not limited to any of the previously mentioned modules. Includes functionality for fitting, plotting, math and more.

4.3. GRAPE algorithm module

The detailed sequence of the GRAPE algorithm is described in Section 3.3, leaving us with the task of distilling the required functions into an abstract interface. Following is a brief summary on the functions and parameters of the *GrapeBackendInterface*, as well as a mention of the supplementary classes. A full overview of the class members and interactions in the GRAPE module can be found in Figure 3.

The first main functionality any backend has to provide when implementing the GRAPE algorithm is the calculation of the quality. The quality is the result of the application of the current piecewise constant control amplitudes for a sampling time dt and following comparison of the outcome with some target unitary matrix. Simulator backends additionally require the definition of a Hamiltonian. The second required

function is the calculation of the gradients, which expects the same parameters as the quality calculation.

The algorithm specific configuration values, such as the termination criteria or amplitude clipping values are grouped in a separate class *GrapeConfig*. A container for storing current and previous parameters such as the control amplitudes or the gradients is provided by the *GrapeData* class. The logical algorithm flow is then executed by the *GrapeRunner*, independently of the underlying backend.

4.4. Closed system simulator backend

The closed system simulator backend acts as a reference implementation for the interface defined in Section 4.3. Following from this, the simulation implements all the mathematical equations described in the Section 3.3, in particular:

- Calculation of a single unitary propagator $U(t)$ given a Hamiltonian and a duration t (2.32)
- Forward propagation of the initial unitary (3.4)
- Backward propagation of the target unitary (3.5)
- Calculation of the quality Φ (3.1)
- Calculation of the gradients $\frac{\delta\Phi}{\delta u_k(j)}$ (3.6)

The popular `numpy` [44] library is used for matrix representation and arithmetic, while the matrix exponential is provided by the `scipy` library [45].

4.5. IBMQ Pulse Adapter

The IBMQ Pulse Adapter takes on the responsibility for infrastructural tasks such as creating, maintaining and closing the connection to the IBM cloud services as well as submitting the jobs and gathering the results.

In addition to the networking and administrative duties, it links the output of the optimal control algorithms to executable jobs for the hardware. Quantum optimal control algorithms, such as GRAPE, produce piecewise constant amplitudes weighting control terms in the Hamiltonian on simulated quantum systems. The simulated systems do not oblige any physical constraints, nor do they provide an interpretation of how these abstract control terms can be mapped to real equipment which is able to exert the desired actions.

With the help of the *Qiskit Pulse* library, which employs the *OpenPulse* specification, the IBMQ Pulse Adapter translates the output control sequences of the algorithms into the language of *pulse schedules*, which can be executed by the target hardware.

4.5.1. OpenPulse

In addition to the widespread and commonly used circuit-based paradigm to program on quantum computers, an alternative approach was published with the *OpenPulse* specification [46]. This specification proposes a common API for pulse-level programming of quantum computers and is at least partly supported by a number of popular quantum computing libraries, such as Qiskit (IBM) or Braket (Amazon). The pulse-level programming model caters to experimentalists, motivating the exploration of the systems via microwave pulses. Additional advanced readout capabilities, for example the acquisition of raw unkerneled data points from measurements, further underline the exploration aspect.

A counterpart to this is the QASM [47] programming model, which defines quantum programs as circuits composed of unitary gates, therefore operating on a much higher abstraction layer. The usage of the circuit model requires less knowledge about underlying physics and focuses on the development of the algorithms themselves. OpenPulse on the other hand grants additional degrees of freedom, allowing to move the focus on to the smaller building blocks of the algorithms, enabling the application of sophisticated and system-tailored optimization techniques. Both approaches aim at different target audiences, each focusing the on different aspects and goals that they want to achieve.

Altogether, we can understand the pulse level programming model and its uniform specification as an additional tool, empowering us to efficiently characterize, calibrate and understand the dynamics of a system in a way that was not feasible before.

4.5.2. Qiskit Pulse

Qiskit Pulse is embedded in the Qiskit SDK [48] and implements the OpenPulse specification. From here on out, the term backend will refer to a single quantum system that is provided by IBM and can run the pulse programs. The library provides a number of convenience functions and utilities to elegantly define microwave pulse schedules, which can initially be done in a backend independent fashion.

Schedules are built of single pulses of arbitrary shape which are applied to *channels*. A pulse is constructed of n complex-valued amplitudes d_i describing the envelope of the drive signal. Each of these amplitudes or samples absolute value is limited between $[-1, 1]$. The actual signal D_i played from the AWGs to the quantum hardware is a modulation of this envelope with a configurable carrier frequency $\omega_{d,i}$ and phase offset $\phi_{d,i}$ generated by a local oscillator (Figure 4.1):

$$D_i = Re\left\{d_i e^{i\omega_{d,i}dt+\phi_{d,i}}\right\} \tag{4.1}$$

Qiskit Pulse allows to set the modulation frequency of any given channel with the `SetFrequency` function. The phase ϕ_d of the carrier can be either set to an absolute value with the `SetPhase` or shifted by a value relative to the current phase with the `ShiftPhase` function. As each sample is specified by a complex amplitude $d_i \in \mathbb{C}$, the phase can also be encoded into the amplitude as the imaginary component of the complex

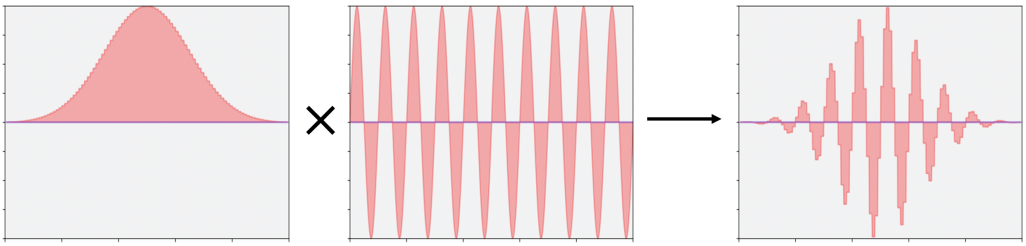


Figure 4.1.: Schematic representation of the signal modulation. The envelope (left) is multiplied with a carrier signal (center) resulting in modulated output signal (right). Taken from [49]

value. This fact is used in Section 4.5.3 and Section 5.2 to simultaneously drive σ_X and σ_Y rotations.

Each sample is played for a certain duration, the *sampling time dt*, which is provided by the backend configuration. The typical sampling time for all examined backends is 0.22 ns. Compliance to a number of timing constraints is necessary before executing a schedule on a real backend:

- Minimum pulse duration of 64 dt $\approx$ 14.2 ns
- Granularity, total pulse duration must be a multiple of 16 dt $\approx$ 3.52 ns

This list is not complete, as it only mentions the constraints relevant to our experiments.

A process similar to the transpilation of a quantum circuit objects is provided, determining if these constraints are fulfilled. Once validated, the pulse schedule can then be executed by the backend as a job similar to regular circuit-type jobs. For targeting a qubit, the library provides different types of *channels*:

- Drive channels D_i are typically used for driving qubit i at its qubit frequency $\omega_{q,i}$
- Control channels U_i are used for signals associated with arbitrary control terms in the Hamiltonian
- Measure channels M_i for applying the measurement stimulus
- Acquire channels A_i connected to the readout components, capable of acquiring data

In practice we are only interested in the first two types of channels, as they allow us to apply our control techniques. We conclude the pulse schedules with measurements of the driven qubits, which is done via the two latter channels. The data resulting from a measurement can be extracted at different abstraction levels, each level representing an additional processing step being applied to the raw data (Level 0). At the time of writing, the systems provided by IBM do not allow access to the raw data, the lowest supported measurement level is Level 1. An overview on the measurement levels is shown

Level	User Inputs	Limitations	Output
0	N/A	High bandwidth, should only be used in averaging mode, no feedback	Time sequences of complex amplitudes
1	Kernel (from list)	No feedback	IQ values
2	Kernel and Discriminator (from list)	Single shot only	Qubit state

Figure 4.2.: Summary of the measurement output levels [46]

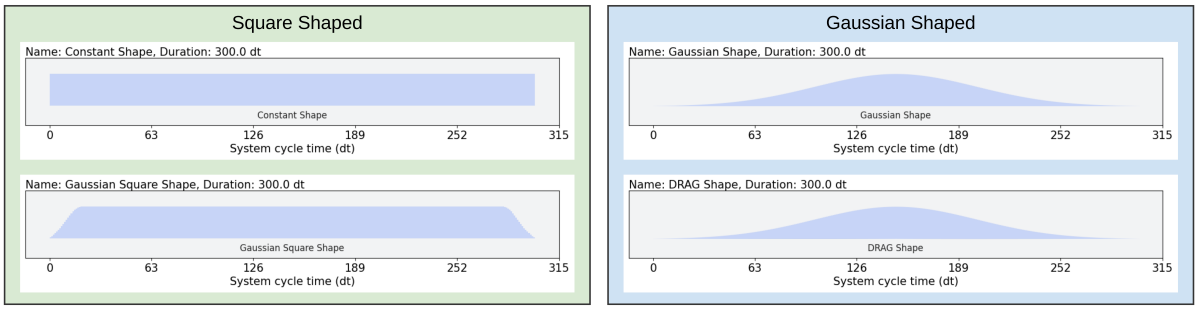


Figure 4.3.: Overview on the parametrized pulses provided by Qiskit Pulse. All pulses shown have a duration of 300 dt and amplitude of 0.15.

in Figure 4.2. Characterization experiments (Section 5.1) make use of measurement level 1, while for the execution of the optimized level 2 is used.

Qiskit Pulse also supplies a set of parametrized common pulse shapes (Figure 4.3), removing the necessity of specifying every sample separately. The choice of the pulse shape has an effect on the impulse response of the qubit it is applied to and will be further discussed in Section 5.1. They can be grouped into two families:

Square shaped Pulses consisting of a horizontal plateau parametrized by an amplitude and a duration in units of 1 dt. For the simplest one, the *constant* shape no additional parameters are required. It consists only of an ordinary square pulse with ideal vertical rise and fall.

The *Gaussian Square* pulse substitutes the constant edges with a Gaussian shaped rise and fall. It is additionally parametrized by the width of the plateau and a σ-factor characterizing the width of the rise and fall.

Gaussian shaped Pulses of this family can be described by a peak amplitude, a duration and a σ-factor specifying the standard deviation. The regular *Gaussian* pulse corresponds to the well-known shape of a normal distribution with a mean centered at half of the duration and truncation at the sides.

A very similar looking pulse shape, known as the *DRAG (Derivative Removal by Adiabatic Gate)* pulse [50] [51],extends the Gaussian pulse by an additional term

weighted by a parameter β. The DRAG pulse shape, originally introduced in 2009, is designed to prevent leakage out of the computational subspace of anharmonic oscillators and is now the underlying pulse shape for the default standard gate set on IBM systems.

Parametrized pulses are extensively used during qubit characterization experiments in Section 5.1, while for the experiments involving optimized pulse sequences in Section 5.2 we specify the samples individually.

4.5.3. Translation of control amplitudes

Based of the Transmon Hamiltonian (2.46) we can differentiate two control Hamiltonians H_1, H_2

$$H_1 = u_1(t)\frac{\Omega_R}{2}\cos(\phi_1)\sigma_X \tag{4.2}$$

$$H_2 = u_2(t)\frac{\Omega_R}{2}\sin(\phi_2)\sigma_Y \tag{4.3}$$

with real amplitudes $u_1(t), u_2(t) \in \mathbb{R}$, Rabi frequency Ω_R and phase $\phi_1, \phi_2 \in [0, 2\pi]$. We know from Section 4.5.2, that the phase is freely configurable and can therefore assume the ability to set $\phi_1 = 0$ and $\phi_2 = \frac{\pi}{2}$, leaving us with

$$H_1 = u_1(t)\frac{\Omega_R}{2}\sigma_X \tag{4.4}$$

$$H_2 = u_2(t)\frac{\Omega_R}{2}\sigma_Y \tag{4.5}$$

as the control Hamiltonians used in the numerical optimization. Inserting the combined Hamiltonian into (2.32), assuming $\omega_q = \omega_d$ and dt as the duration for a single propagation step j, the generated unitary propagators take the form:

$$U(j) = e^{-i\frac{\Omega_R}{2}(u_1(j)\sigma_X + u_2(j)\sigma_Y)dt} \tag{4.6}$$

The result of the optimization as shown in Figure 4.4 are two real amplitude vectors u_1, u_2 with length j corresponding to σ_X and σ_Y rotations. We have two possibilities for translating these control amplitude vectors into Qiskit Pulse schedules. A straightforward approach is based on the idea of creating a pulse schedule by concatenating the real amplitudes $u_1(j)$ and $u_2(j)$, separated by a phase shift of $\frac{\pi}{2}$ (Figure 4.5a). The second idea implicitly encodes the phase shift by calculating a complex amplitude $c(j) = u_1(j) + iu_2(j)$ which consequentially halves the duration of the required pulse schedules (Section 4.5.3). As we are interested in time-optimal implementations of the target unitaries, the latter approach is used to conclude the experiments in Section 5.2.

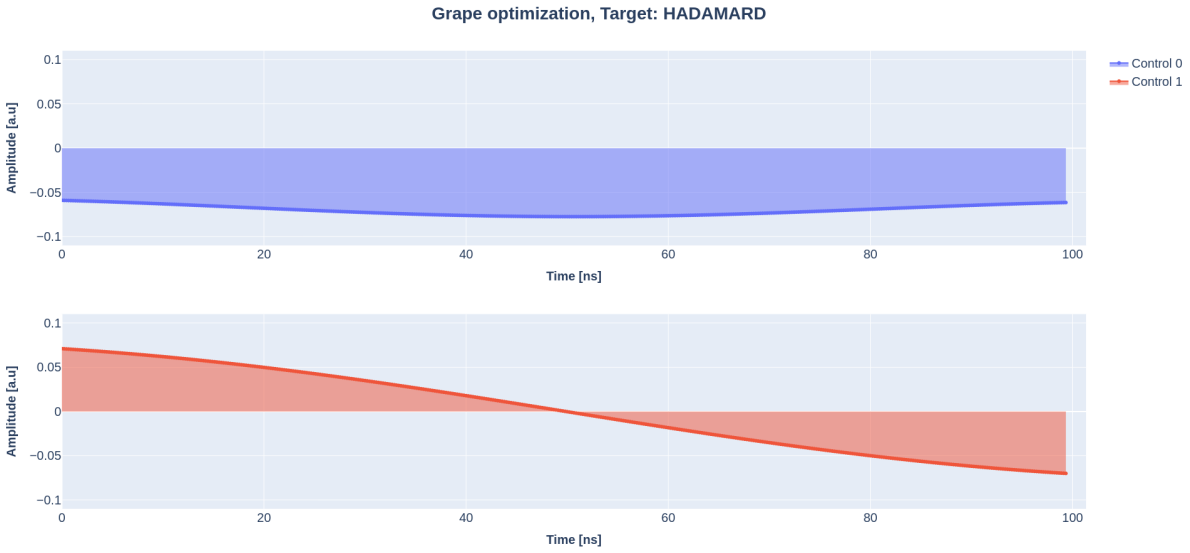
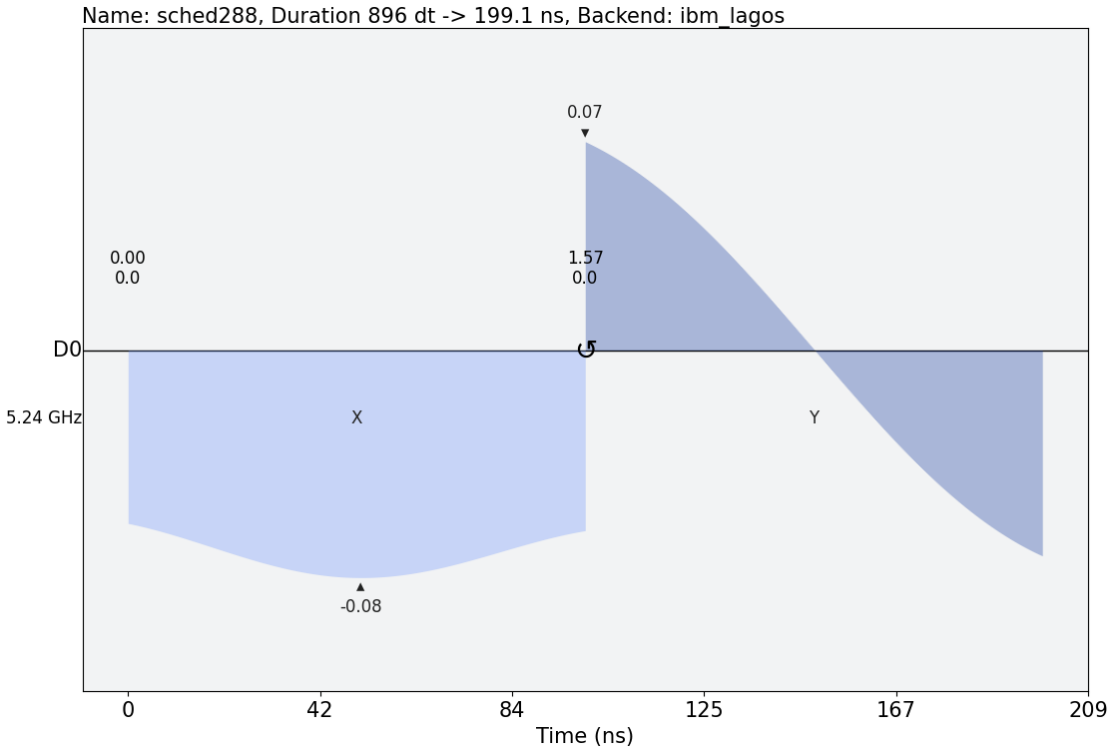
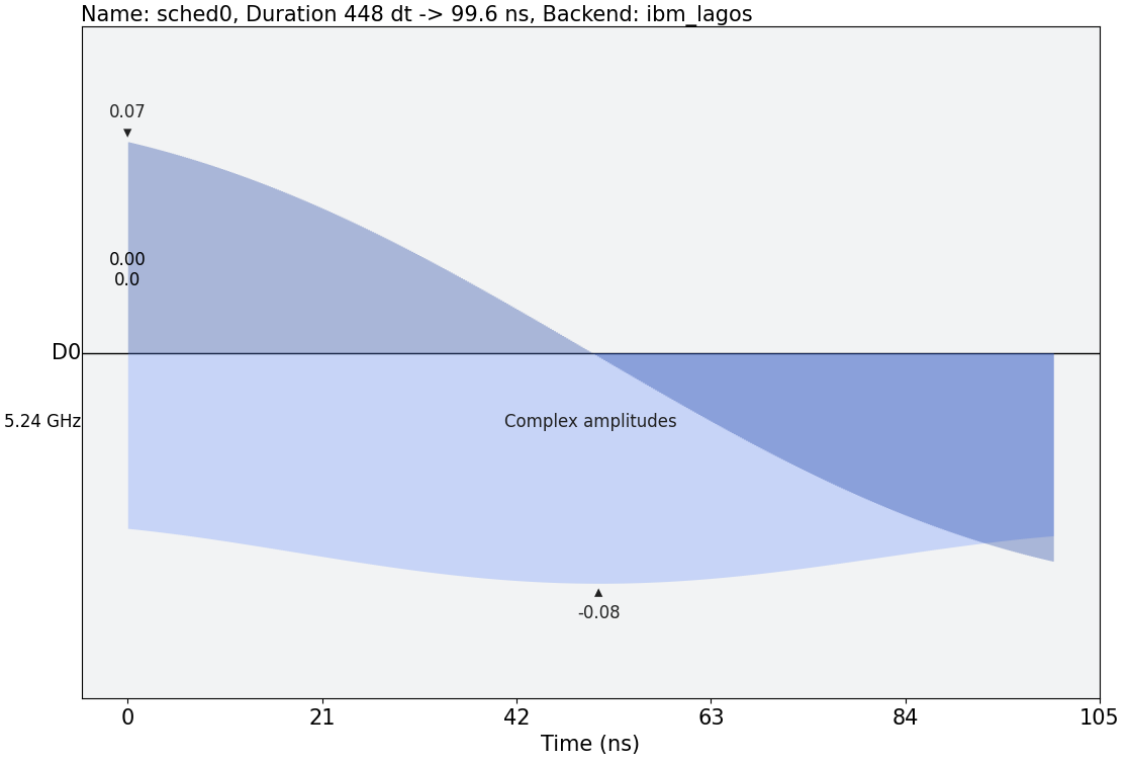


Figure 4.4.: Final control amplitudes for optimization of a Hadamard unitary with a duration of $T \approx 100\,\text{ns}$. Control 0 corresponds to the u_1-amplitudes driving a σ_X-rotation, while Control 1 represents the u_2-amplitudes for a σ_Y-rotation.



(a) Sequential scheduling of $u_1(j)$ and $u_2(j)$ from Figure 4.4



(b) Complex amplitude scheduling as $c(j) = u_1(j) + iu_2(j)$ with amplitudes from Figure 4.4

5. Experiments

The previous chapters laid the theoretical foundation for carrying out experiments on real systems. Building on this, the following section will introduce common experimental setups and techniques for determining the required model parameters to numerically optimize quantum gates for a particular system. These experiments have been fully automated and can be executed on any IBMQ backend on any desired qubit.

Further we will use the acquired parameters in the optimization, resulting in pulse sequences which implement single qubit gates for a specific qubit. The performance in terms of the achieved process fidelity of the numerically optimized gates is then compared with the process fidelity of the gates provided by IBM.

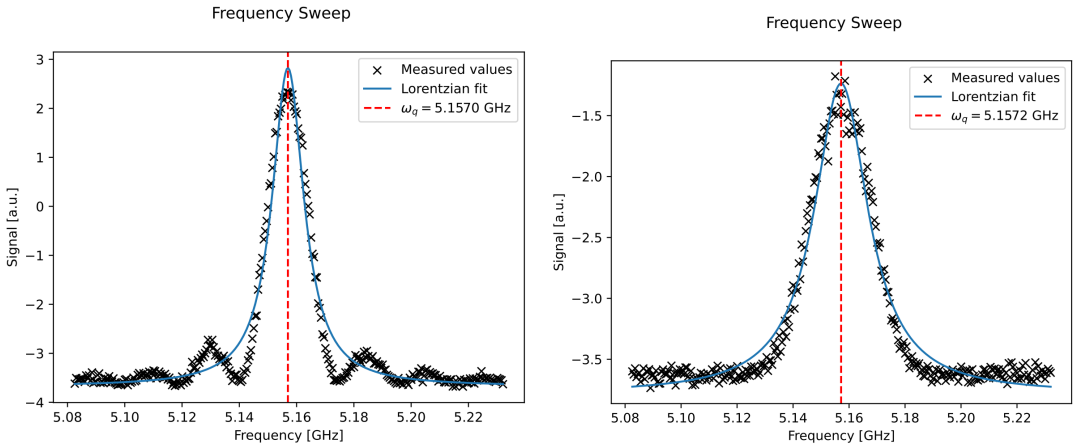
5.1. Qubit characterization

The application of quantum optimal control requires an adequate model of the system it is applied to. We defined the structure of this model when discussing Transmon qubits in Section 2.4.2, which resulted in the definition of the single qubit Hamiltonian (2.46). This leaves us with the task of experimentally determining the characteristic qubit parameters, namely its qubit frequency w_q and the Rabi frequency Ω_R. These sequence of these experiments are sometimes also known as *qubit spectroscopy*.

5.1.1. Determining the qubit frequency

First objective and prerequisite for all following experiments is a rough determination of the qubits transition frequency ω_q. Long pulse durations in this experiment cause errors, such as inhomogenities in control fields, to accumulate. These errors lead to an amount of uncertainty in the frequency discovered from this experiment, yet it is precise enough to enable follow-up experiments which will later give us the chance to correct the rough value.

Recalling from Section 2.4.2, state transitions are driven by the application of an electromagnetic field modulated at the resonance frequency ω_q for a particular duration. As Qiskit Pulse allows us to set the modulation frequency ω_d of the driving signal, we can set up an experiment known as a *Frequency Sweep*. In this experiment we create a series of pulse schedules, each consisting of the same pulse modulated at slightly different frequencies ω_d. The exact amplitude and duration of the applied pulse are of secondary importance, but should be sufficiently large to trigger a transition from the ground to the first excited state. From previous experiments it can be concluded that an amplitude of 0.05 and a duration of 50 ns is sufficient.



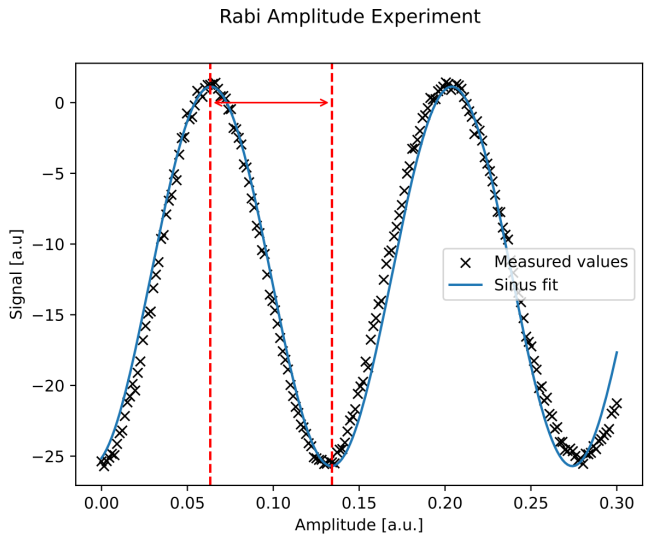
(a) Constant pulse (b) Gaussian pulse

Figure 5.1.: Result of a Frequency Sweep experiment executed on the backend *ibm_perth* for qubit 0 for different pulse shapes.

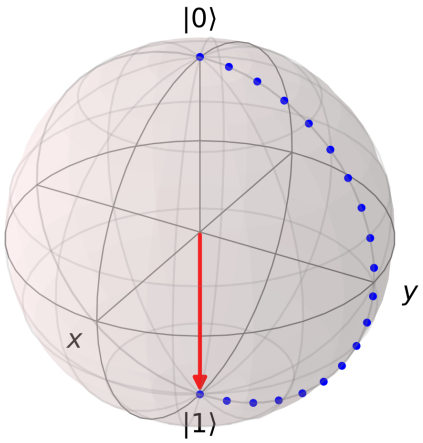
The data is acquired as a single complex value (Measurement level 1) and averaged for every shot, yielding one data point for every frequency value in the sweep. It should be emphasized at this point, that in this and the following experiments, we are not interested in the actual scalar value of the measurements and will therefore treat the outputs as arbitrary units. Most of the time, the received values have been scaled linearly to remove powers of ten which do not have any influence on the spectated effects. Our interest lies in the relation of the measured values as a function of some other quantity, in this experiment the frequency. After plotting the real part of the measured signal, we search for signs of absorption in the spectrum by fitting a Lorentzian shaped curve.

Carrying out this experiment with different pulse shapes reveals differences in the responses depending on the type of pulse used (Figure 5.1). The fitted center frequency $f_q = \frac{\omega_q}{2\pi}$ for the pulse shapes differs by approximately 0.2 MHz, which is negligible assuming we follow up with the Ramsey experiment described in Section 5.1.3. A much more meaningful difference can be found in the general appearance of the responses. The response of the constant pulse shows secondary peaks to both sides of the main peak, reminiscent of the Fourier transform of a square pulse, whereas the response for the Gaussian pulse remains flat before and after the center peak. This has implications on the design of the quantum hardware, as the transition frequencies of neighboring qubits have to be sufficiently detuned to avoid accidentally driving qubits on the same transition line [23]. Experiments stimulating qubits via the sidebands, by high amplitude square pulses on a neighboring qubit have been conducted but have not resulted in any considerable correlation.

Another characteristic feature is the width of the aperture, appearing broader in the case of stimulation with a Gaussian pulse. Again, this underlines the need for qubits on the same transition lines to differ in their transition frequencies for at least the width of this curve.



(a) Result of a Rabi experiment executed on backend *ibmq_guadalupe* for qubit 0. The interval between the dashed vertical lines shows a π–amplitude of 0.0702.



(b) Trajectory of a π–pulse on the Bloch sphere.

Figure 5.2.: Rabi experiment

5.1.2. Calibrating a π-Pulse

After finding the rough value for ω_q, the next step in characterizing a qubit is to determine a pair of amplitude and duration that will drive a qubit from the ground to the excited state. Such a pulse is called a π–Pulse, corresponding to a rotation of 180° around the x-axis on the Bloch sphere (Figure 5.2b).

A typical experiment for this task is the *Rabi experiment*. The setup is similar to the previous one used in the Frequency Sweep, except in this occurrence the applied drives are modulated at a constant frequency $\omega_d = \omega_q$ and the amplitudes are swept. For consistency with the previous experiment we chose the same pulse duration of 50 ns. Again we configure the experiment to yield a single complex value for each amplitude. Figure 5.2a shows the results of this experiment for amplitudes between $u_{min} = 0.0$ and $u_{max} = 0.3$ with a spacing of $\Delta u = 0.0015$ which translates to $n = 200$ data points. We observe the function of the received signal against the amplitude to be of sinusoidal shape, oscillating between ground and excited state. The inverse of the frequency resembles the amplitude of a full oscillation, we can therefore extract the π–amplitude by applying a factor of 0.5. For the specific example of qubit 0 on backend *ibmq_guadalupe* the analysis results in an amplitude of 0.0702. Experiments on a number of different backends and qubits resulted in values between 0.05 and 0.25.

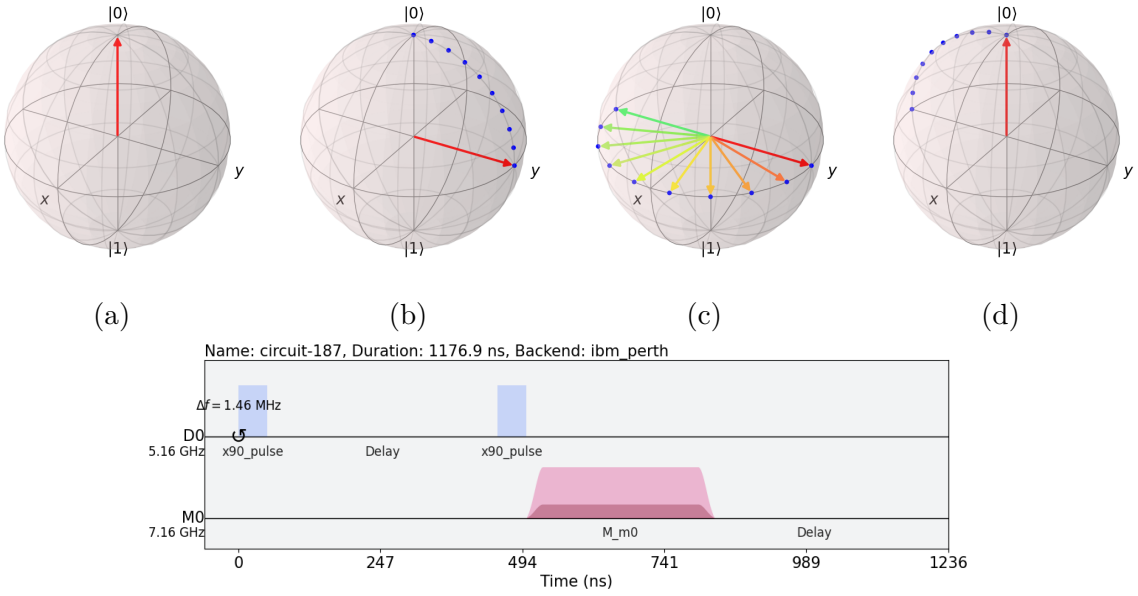
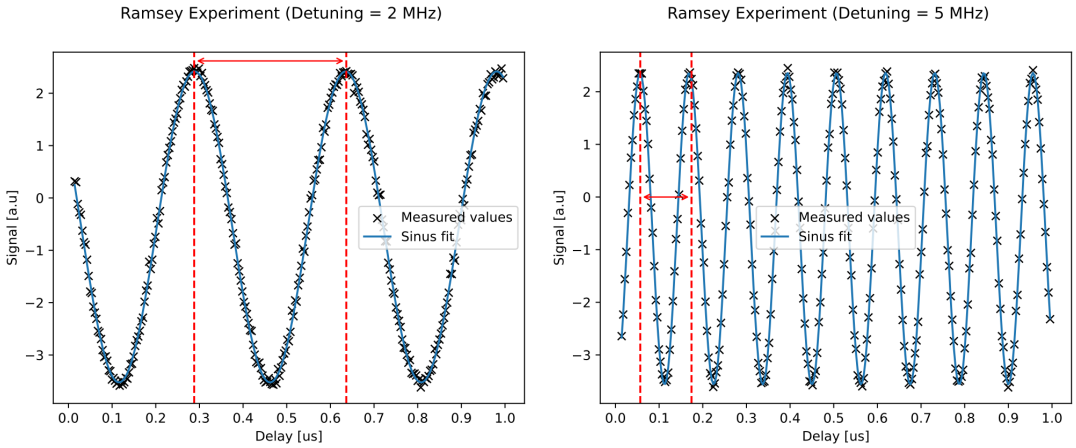


Figure 5.3.: Ramsey experiment sequence: **(a)** Initial state $|0\rangle$ **(b)** State after application of the $\frac{\pi}{2}$–pulse **(c)** Free evolution with frequency $\Delta = \omega_q - \omega_d$ for some time t_{delay} **(d)** Final state $|0\rangle$ after second application of $\frac{\pi}{2}$–pulse for the exemplary case a phase of $\varphi = \frac{\pi}{2}$ has been accumulated. **(e)** Pulse schedule for a single Ramsey circuit

5.1.3. Correcting the qubit frequency value

After finding the amplitude and duration needed for a π–pulse we can now carry out an experiment to determine the qubit frequency acquired in Section 5.1.1 to a greater precision. Following we will refer to the exact qubit frequency as ω_q and to our earlier estimation as $\omega_{q\_estimation}$. In contrast to the Frequency Sweep experiment, the main interaction parts are notably shorter, thus leading to fewer errors and a more accurate value of ω_q. This experiment was originally introduced as the *separated oscillatory field method* [52], but is nowadays known as the *Ramsey interferometry*. The experiment is based on the idea of measuring the accumulated net phase caused by slightly off-resonant drives.

To gain an intuition on the procedure of this experiment, it is helpful to visualize the Bloch sphere with basis states $|0\rangle$ or $|1\rangle$ at the north and the south poles. Precession is then given by a rotation around the z-axis, which does not affect the basis states as they reside on this axis. The first step is therefore a rotation of $\frac{\pi}{2}$ about the x-axis to the qubit, taking the qubit to a state that lies on the xy-plane. To accomplish this, we use the π–pulse from Section 5.1.2 and scale the amplitude by a factor of 0.5. In the next step, we let the system evolve freely for a configurable time t_{delay} in which it picks up some phase value φ. After the delay period we follow up with another $\frac{\pi}{2}$–pulse, projecting the state back onto the z-axis. The final state is now dependent on the phase



(a) $\delta = 2\,\text{MHz}$ (b) $\delta_{detune} = 5\,\text{MHz}$

Figure 5.4.: Results of the Ramsey experiment for different detuning frequencies δ with the precession time t_{delay} ranging from 0 ns to 1000 ns. The experiment was concluded on qubit 0 of *ibm_perth*.

φ that has been picked up during the free evolution time, which itself is a function of the delay time t_{delay} and the speed of precession. Each experiment for a single delay duration t_{delay} has to be repeated multiple times to extract a meaningful output value. Figure 5.3 visualizes a single exemplary Ramsey sequence on the Bloch sphere as well as a pulse schedule implementing the setup.

As previously mentioned, the rate at which the phase φ is acquired depends on the net speed of the rotation about the z-axis. By moving into a frame rotating at the frequency of the applied drive ω_d (2.46), this precession frequency Δ can be interpreted as difference between the exact qubit frequency ω_q and the drive frequency ω_d. As a consequence, we can deliberately induce a small additional detuning by offsetting the drive modulation by a known amount δ, resulting in the updated drive frequency $\omega_d' = \omega_d + \delta$.

Figure 5.4 shows the output signal values as a function of the delay time t_{delay} captured from the experiment. After fitting a sinusoidal and extracting its frequency ω_{ramsey}, we can subtract the ancillary detuning, leaving us with only the deviation to our estimation of $\omega_{error} = \omega_{ramsey} - \delta$. The estimation of the qubit transition frequency can then be corrected with $\omega_q = \omega_{q\_estimate} + \omega_{error}$.

5.1.4. Determining the Rabi frequency Ω_R

In this section we investigate the relation between the duration $t_{2\pi}$ and the amplitude u required to implement a 2π–pulse for different pulse shapes. After gathering the durations, one per amplitude, we calculate the Rabi frequency Ω_R as the inverse of the duration scaled by the corresponding amplitude:

$$\Omega_R = \frac{1}{t_{2\pi} u} \tag{5.1}$$

This relation is based on the assumption that the Rabi frequency scales linearly with the inverse of the amplitude, which we will show to be true up to some amplitude limit. Based on the discovered range at which we can assume a linear relationship, we conduct additional experiments within these limits, allowing us to calculate a mean Ω_R to be used in the Hamiltonian (2.46) for our optimal control algorithms.

The full setup comprises a number of single Rabi experiments similar to the ones described in Section 5.1.2. Contrary to the previous Rabi procedure, we vary the duration of the pulse and keep the amplitude constant. Consequentially, the extracted period $t_{2\pi}$ describes the duration for which a pulse of the corresponding amplitude has to be applied for to drive a full 2π–rotation around the x-axis of the Bloch sphere. The duration interval ranges from the shortest allowed pulse duration $t_{min} = 64\,\mathrm{dt} \approx 14.22\,\mathrm{ns}$ up to $t_{max} = 1344\,\mathrm{dt} \approx 295.1\,\mathrm{ns}$. The step size of $t_{step} = 16\,\mathrm{dt} \approx 3.6\,\mathrm{ns}$ is limited by the granularity constraint from Section 4.5.2, yielding a total number of 80 data points per Rabi experiment. After each single experiment, the amplitude is increased and another experiment is conducted.

Figure 5.5 compiles the results for a number of single runs with amplitudes between $u_{min} = 0.02$ and $u_{max} = 1.0$ and a constant pulse shape. For amplitudes larger than $u = 0.3$, fitting a proper sinusoidal shape becomes increasingly difficult as there are not enough samples in between the oscillations. The granularity constraint defines the lower bound on the increments of the duration steps, prohibiting the acquisition of more samples during a given interval. Figure 4 depicts an overview of the extracted 2π–periods for all tested pulse shapes encouraging the assumption of an inverse linear relationship between amplitude and duration. Figure 5 reinforces this, as the corresponding frequencies $f_{2\pi} = \frac{1}{t_{2\pi}}$ appear in approximately a straight line. With rising amplitudes, the linear relationship lessens and the curve flattens towards the end. The effect of this non-linearity is even more obvious after calculating Ω_R via (5.1), as can be seen in Figure 6. Provided that the previous assumption holds true we expect a slope close to zero, which is clearly not the case for larger amplitudes. As a result of this, we will only consider amplitudes in the interval between $0.02 \leq u \leq 0.30$ for determining the Rabi frequency Ω_R.

By analyzing the diagrams shown in Appendix B with regard to the pulse shapes, we find that pulse shapes of the same family (Section 4.5.2) yield largely similar results. As our access to the computational resources is limited, we will therefore conduct further experiments only with constant and Gaussian shaped pulses. Gaussian shaped pulses require longer durations to achieve the same rotation angle than Square shaped pulses, which can be attributed to the numerically lower area under pulse envelope. The extent of this deviation depends on the σ parameter of the Gaussian pulse, as this parameter significantly determines the width and thus the enclosed area below the curve. In the next step, we execute the Rabi experiments in the amplitude range $0 \leq u \leq 3.1$ with the goal of finding a single Ω_R to use in the optimization algorithm. Figure 5.6 illustrates the results, again showing a difference by a factor of approximately 0.5 between the constant and the Gaussian shaped pulses. To extract a single value per pulse shape we apply the median and are left with two candidates for the Rabi frequency. We evaluate the performance of both by constructing pulse sequences driving a π–rotation from the gathered period amplitude pairs (dividing the period by two) and simulate them for each

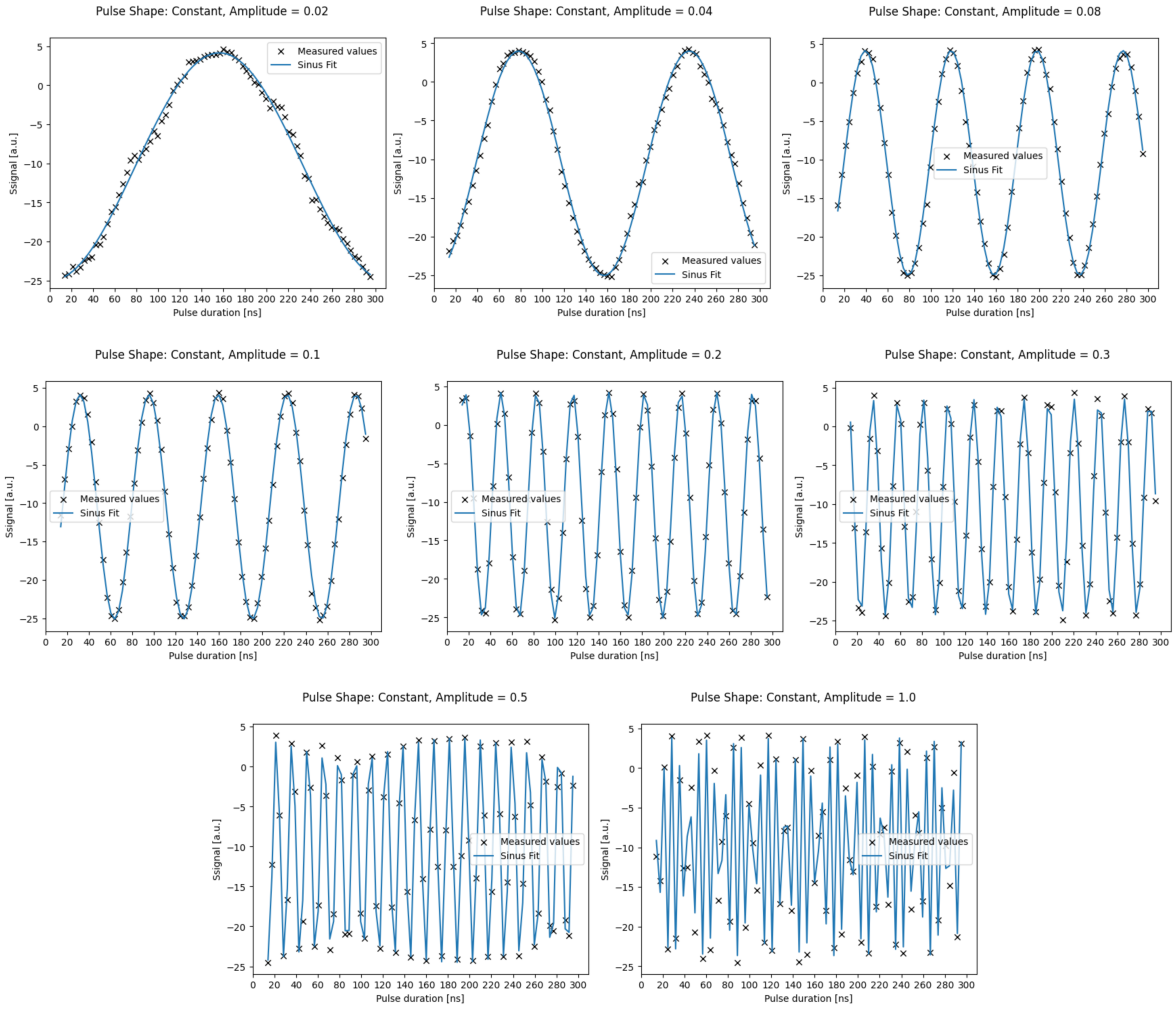
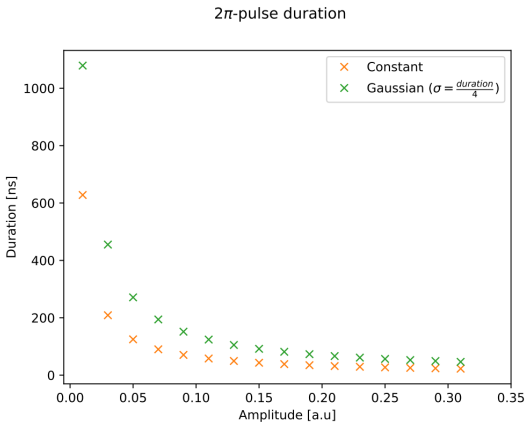
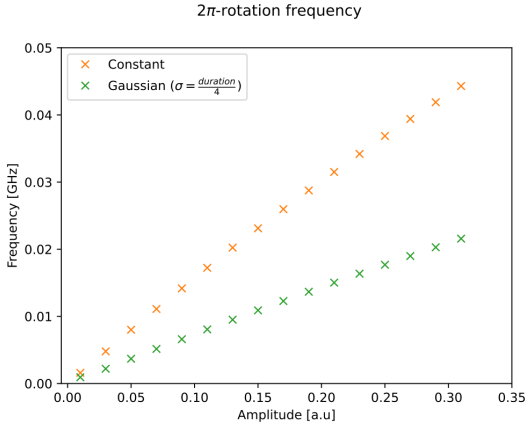


Figure 5.5.: Multiple Rabi experiments with varying pulse durations on qubit 0 of *ibmq_guadalupe*. Each experiment was conducted with a different amplitude ranging from $u_{min} = 0.02$ to $u_{max} = 1.0$.

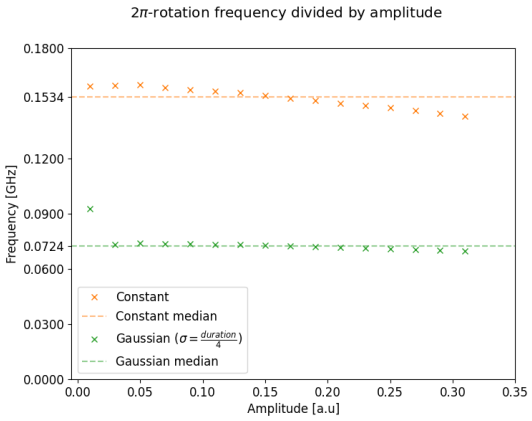
candidate by unitary time evolution with a propagator (3.2) and Hamiltonian (2.46). The propagator is then applied to the initial state $|0\rangle$ via (2.33), resulting in a new state from which we calculate the probability of being measured in $|1\rangle$. We cross-validate the pulse sequences by also executing them on the physical backend. Figure 5.7 shows that the probability for the final state to end up in $|1\rangle$ is mostly above 95% for both pulse shapes in case of the execution on the real backend. The simulation based on the Rabi frequency extracted from the Gaussian pulse shapes performs poorly, never reaching more than 75%, whereas the probability using constant pulse shapes match the expectation of $\approx 100\%$. We will therefore use the constant pulse shape in our experiments determining the Rabi frequency.



(a) Duration for a 2π–pulse



(b) Inverse of the duration (frequency) for a 2π-pulse



(c) Scaled frequency for a 2π–pulse

Figure 5.6.: Results of the Rabi experiment with varying duration for amplitudes $0 \leq u \leq 3.1$ executed on qubit 0 on *ibmq_guadalupe*. Image **(a)** shows an inverse linear relation between amplitude and pulse duration for a 2π–pulse. The corresponding frequency in **(b)** consequentially shows an approximately linear relation. Figure **(c)** shows the calculated Rabi frequency Ω_R via (5.1). We utilize the median (dashed line) to eliminate outliers.

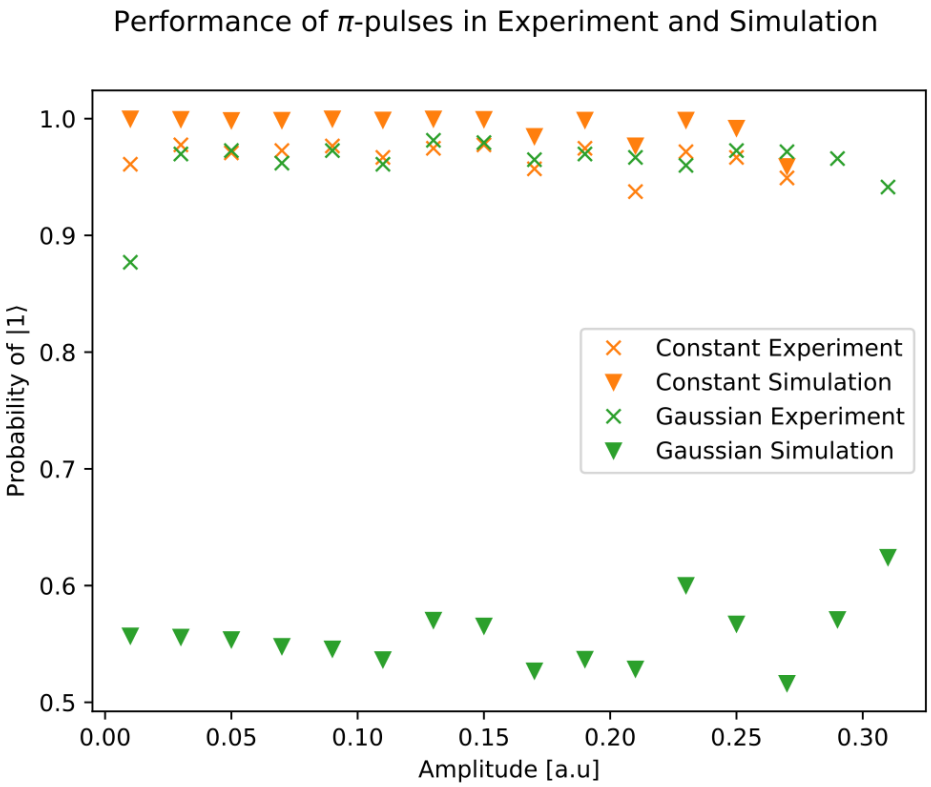


Figure 5.7.: Comparison of the performance of π–pulse sequences executed on qubit 0 on backend *ibmq_guadalupe* and in the simulated unitary evolution. The Gaussian experiments show two additional data points, as the extracted pulse durations for these amplitudes are shorter than allowed.

5.2. Single Qubit Gates

We will now assemble the pieces from the previous chapters into a complete experimental setup, aiming to analyze the performance of the optimized pulse sequences on real backends. A single run of the experiment consists of:

1. Determination of the characteristic qubit parameters ω_q and Ω_R as shown in Section 5.1
2. Optimization of control amplitudes $u(t)$ that implement a desired unitary Section 3.3
3. Translation of the control amplitudes into pulse schedules Section 4.5.3
4. Creation of the quantum circuits for the QPT as described in Section 3.4.2
5. Execution of the jobs on the IBMQ quantum computers
6. Calculation of the process fidelity with the `qiskit.quantum_info` library (Section 3.4.2)

In addition to running the optimized pulse sequences, we also compose the analogous circuit from the gate provided by the backends themselves and measure the process fidelity for comparison. The first set of experiments where conducted on the 7-qubit

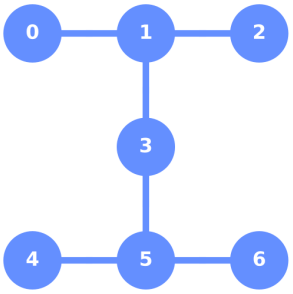


Figure 5.8.: Qubit arrangement and coupling on *ibm_perth* and *ibm_lagos*.

backend *ibm_perth* (Figure 5.8), optimizing pulse sequences of rising durations for the implementation of a X-Gate. We use σ_X with a global phase of $-i$ to stay in the $SU(2)$ as our target unitary:

$$U_{target} = -i \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \tag{5.2}$$

Since this gate only requires rotations about the x-axis, it is sufficient to use only the H_1 control term as defined in (4.4). We assume $\omega_q = \omega_d$, canceling the impact of the σ_Z-term in the drift Hamiltonian, leaving us with only the control term in the Hamiltonian $H = H_1$ used in the optimization. The sampling time extracted from the backend configuration is $dt \approx 0.22\,\text{ns}$. Pulse sequences are optimized for durations ranging from $T_{min} = 96\,\text{dt} \approx 21.3\,\text{ns}$ up to $T_{max} = 896\,\text{dt} \approx 199.1\,\text{ns}$ with initial amplitudes of both constant shape $u_0 = 0.0005$ and Gaussian shape $u_0 = 0.2, \sigma = 0.15T$.

Figure 5.9 shows an example of the optimized pulse schedules for qubit 3 on *ibm_perth* with a duration of $T \approx 40\,\text{ns}$ and the corresponding schedule provided by IBM for the same qubit. We observe the final shapes of the optimized pulse sequences to be largely similar than the shape of the initial pulse sequences.

A descriptive explanation for this behavior can be constructed from the choice of the particular target unitary, as it only requires monotone rotation about a single axis, thus leading to similar gradients for all control amplitudes. As mentioned in Section 4.5.2, the provided default gate set always utilizes the DRAG shape for the underlying pulse sequences to reduce the effects of state leakage. The second main difference, which is also an effect of the differing pulse shapes, is the discrepancy in the amplitudes. This effect is similar to the one observed in Section 5.1.4, caused by the proportionality between the required area under the pulse and the angle of the desired rotation.

In Figure 5.10 we can observe the exchange in population during the simulated application of two optimized pulses of constant initial shape with durations $T \approx 130\,\text{ns}$ and $T \approx 40\,\text{ns}$ on the initial states $|0\rangle$ and $|1\rangle$. Applying a constant pulse can be viewed as linearly increasing the rotational angle every timestep, reaching the equal superposition at $\frac{T}{2}$ and the full population exchange after the full time T in both cases. The shorter pulse (Figure 5.10b) requires a higher amplitude to drive the population exchange in a shorter time, which seems to have a negative effect on the process fidelity achieved in the experiment. This effect might be a consequence of the state leakage occurring for higher amplitudes, a claim that should be studied more thoroughly in future work.

Figure 5.11 depicts the evolution of $\langle \sigma_Z \rangle$ for optimized pulse sequences with Gaussian shaped initial amplitudes. For the longer pulse with a duration of $T \approx 130\,\text{ns}$ (Figure 5.11a) we can see that the population exchanges multiple times, which may act as a source of error due to the unnecessary long interaction. A shorter duration with the same amplitude (Figure 5.11b) is therefore a better choice, combining the advantages of a short duration and comparable $\mathcal{F}_{pro}$, deviating only at the fourth decimal. Comparing the performance of the short Gaussian pulse with the long constant shaped pulse reveals a difference in $\mathcal{F}_{pro}$ by about $5 \cdot 10^{-3}$ for the Gaussian shaped pulse. As the Gaussian shaped pulse is more than three times shorter than the constant pulse, one might classify the Gaussian pulse higher depending on the prioritization between duration and fidelity of the specific use-case.

Figure 5.12 compares the best performing pulse sequences of either shapes for the respective qubits against the provided gate. The x-axis shows the minimal error $1 - \mathcal{F}_{pro}$ achieved for all durations between T_{min} and T_{max} of pulse sequences. It can be seen that our optimized pulse sequences outperform the default gates for all qubits in terms of the minimal error. This comes at the cost of longer gate times, which is especially apparent in the case of the constant amplitudes. Inspecting the arrangement of the qubits in Figure 5.8 and the results in Figure 5.12, a correlation between the number of adjacent qubits and the error values cannot be spotted. Qubit 1 and 5 both neighbor three other qubits but do not show higher error values than for qubit 0 and 4 with only a single neighbor. This result is rather unexpected, as one would anticipate the influence of local couplings between adjacent qubits to negatively affect the achieved fidelities.

The second set of experiments was conducted on the 7-qubit backend *ibm_lagos*,

optimizing pulses for the target

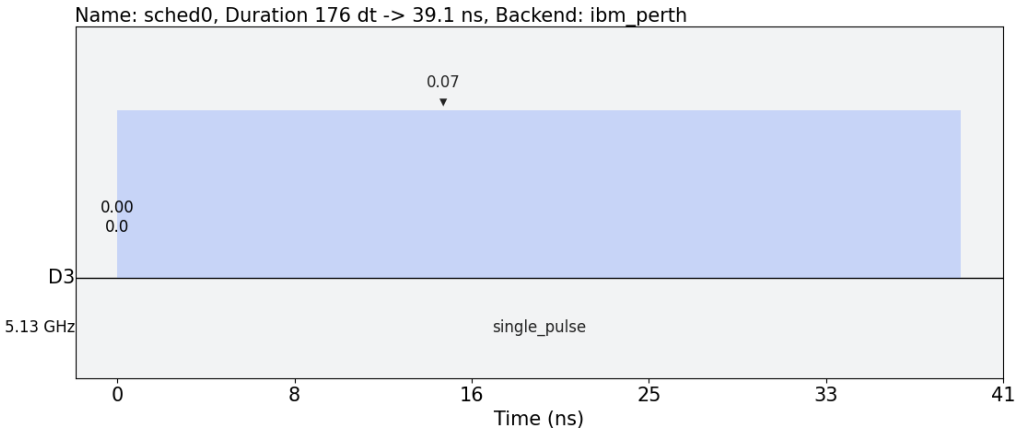
$$U_{target} = \frac{i}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \tag{5.3}$$

which represents the unitary implementing the Hadamard-Gate with a global phase of i.

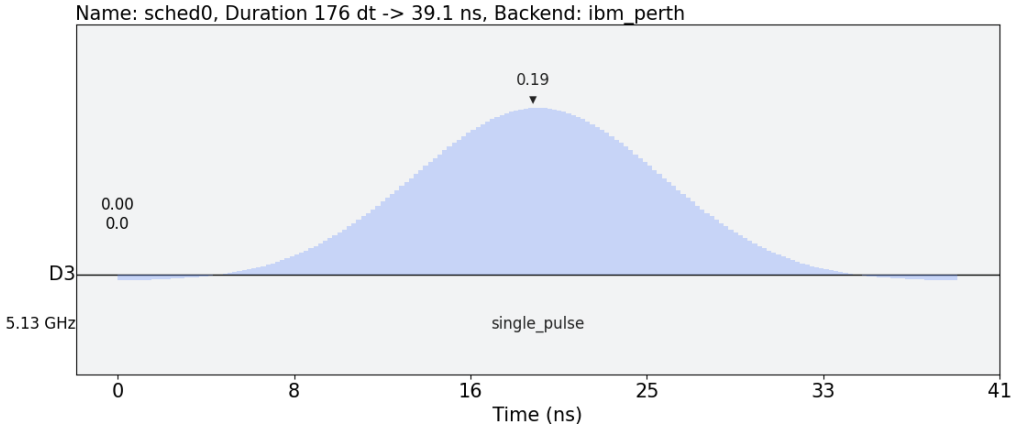
The control operations for the Hamiltonian employed in the optimization algorithm consist of a σ_X and σ_Y term as defined in (4.4) and (4.5). Again, cancellation of the σ_Z term due to $\omega_q = \omega_d$ is assumed. The sampling time $dt \approx 0.22\,\text{ns}$ as well as the qubit arrangement (Figure 5.8) are equal to the backend used in the previous experiment. Pulse sequences are optimized for durations between $T_{min} = 96\,\text{dt} \approx 21.3\,\text{ns}$ and $T_{max} = 896\,\text{dt} \approx 199.1\,\text{ns}$ with constant initial amplitudes $u_0 = u_1 = 0.0005$ for both the σ_X- and σ_Y-controls.

Figure 5.15b exposes the fact that the provided H-Gate does not actually implement the assumed target unitary. The pulse schedule starts off with a relative phase shift of $\phi = \phi - \frac{\pi}{2}$, followed by a drive pulse of rotational angle $\frac{\pi}{2}$ concluding with another phase shift of $\phi = \phi - \frac{\pi}{2}$. In consequence, the pulse drives a rotation of $-\frac{\pi}{2}$ about the y-axis leading to the same result as a H-Gate. The optimized pulse sequence (Figure 5.15a) in contrast constructs the H-Gate by a combination of simultaneous σ_X- and σ_Y-rotations, which could potentially lead to a larger error due to the additional applied control fields.

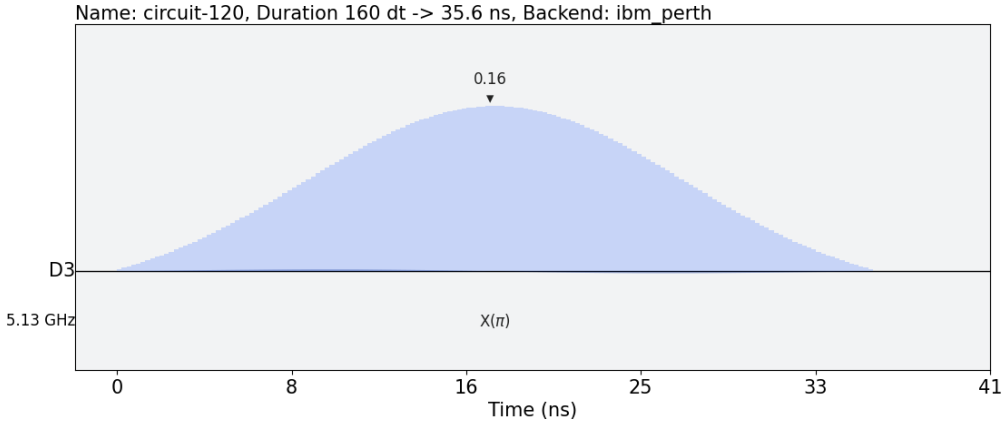
This assumption does not always hold, as Figure 5.14 shows that the optimized pulses outperform the provided gate in 4 out of 7 cases. Figure 5.13 depicts the simulated evolution of $\langle\sigma_Z\rangle$ for the basis states $|0\rangle$ and $|1\rangle$, resulting in the desired equal superposition of the basis states both with $\langle\sigma_Z\rangle = 0$ at $T \approx 28\,\text{ns}$. We compare the errors $1 - \mathcal{F}_{pro}$ between the provided H-Gate and the best performing pulse sequences per qubit in Figure 5.14. The high error of the optimized pulses for qubit 1 immediately jumps the eye, which cannot be explained after inspection of the generated control amplitudes, as the shape and amplitudes look similar to the ones executed on the other qubits. We therefore conclude this as an outlier which could have been caused due to unknown environmental issues or issues with the control electronics. Again we observe that the optimized pulse sequences outperform the provided gate on 4 out of 7 qubits.



(a) Constant initial amplitudes

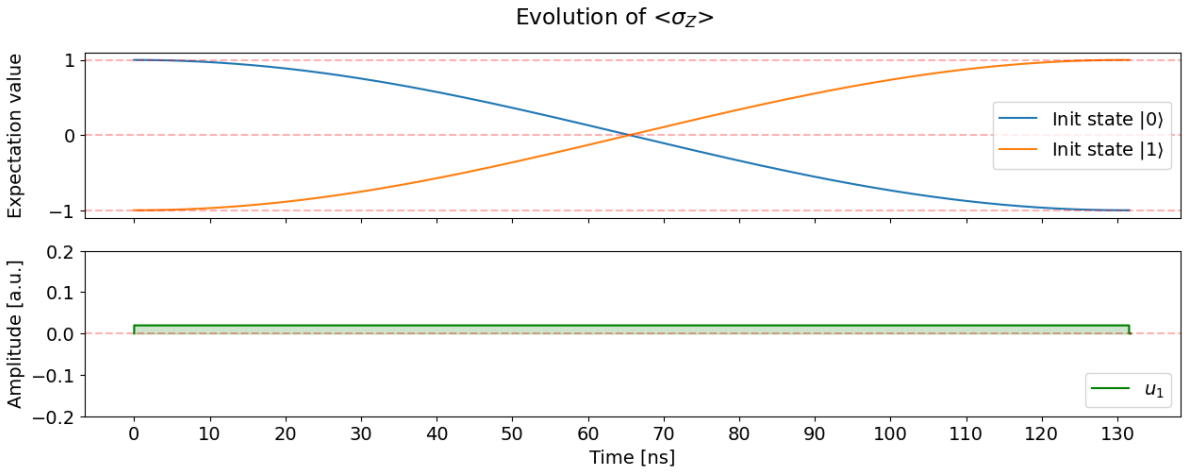


(b) Gaussian shaped initial amplitudes

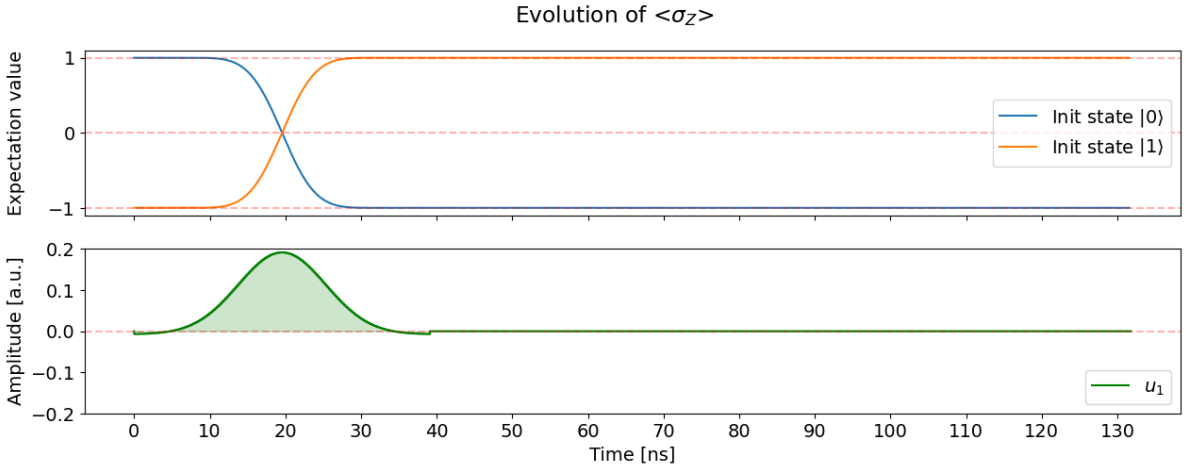


(c) Pulse schedule for default X-Gate

Figure 5.9.: **(a), (b)** Optimized pulses sequences with different initial shapes for implementing a X-Gate. We observe minimal deviations between the optimized and initial shapes. **(c)** Provided X-Gate for qubit 3 on *ibm_perth*.

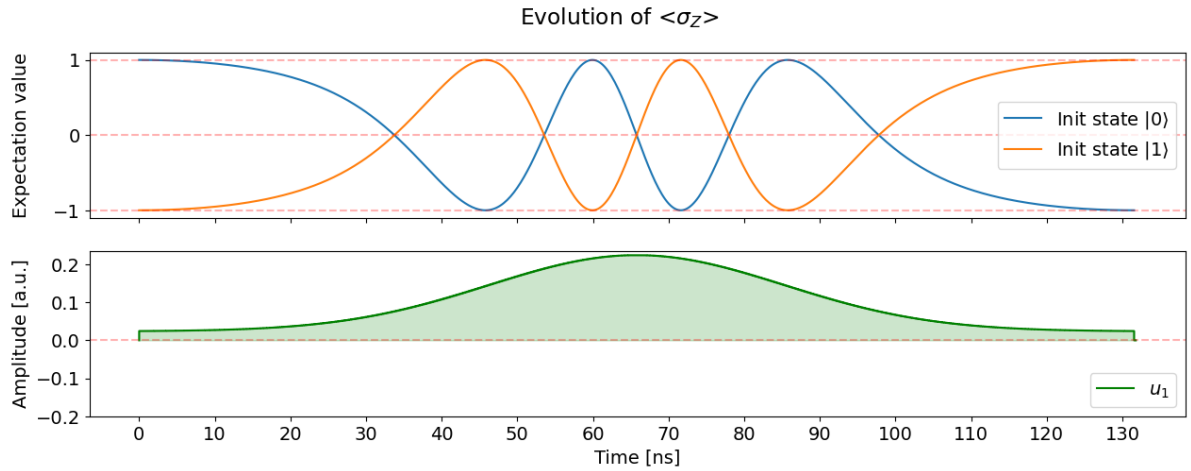


(a) Optimized pulse with $T \approx 130\,\text{ns}$ and constant initial shape resulting in $\mathcal{F}_{pro} = 0.9892$

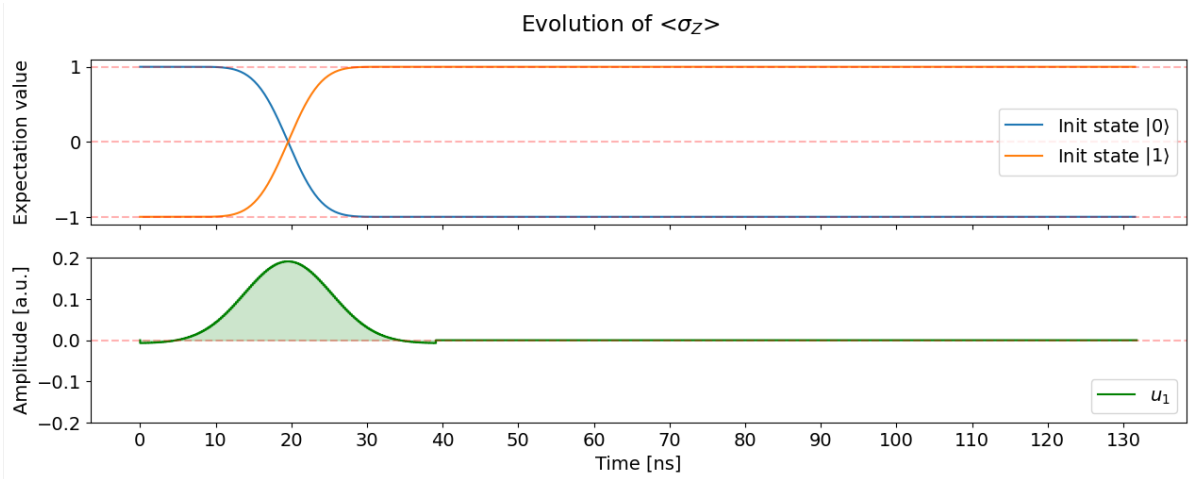


(b) Optimized pulse with $T \approx 40\,\text{ns}$ and constant initial shape resulting in $\mathcal{F}_{pro} = 0.9758$

Figure 5.10.: Simulated evolution of the expectation value $\langle \sigma_Z \rangle$ for pulses of different durations with initial constant shape implementing a X-Gate.



(a) Optimized pulse with $T \approx 130\,\text{ns}$ and Gaussian shaped initial amplitudes resulting in $\mathcal{F}_{pro} = 0.9832$



(b) Optimized pulse with $T \approx 40\,\text{ns}$ and Gaussian shaped initial amplitudes resulting in $\mathcal{F}_{pro} = 0.9838$

Figure 5.11.: Simulated evolution of the expectation value $\langle\sigma_Z\rangle$ for pulses of different durations with initial Gaussian shape implementing a X-Gate.

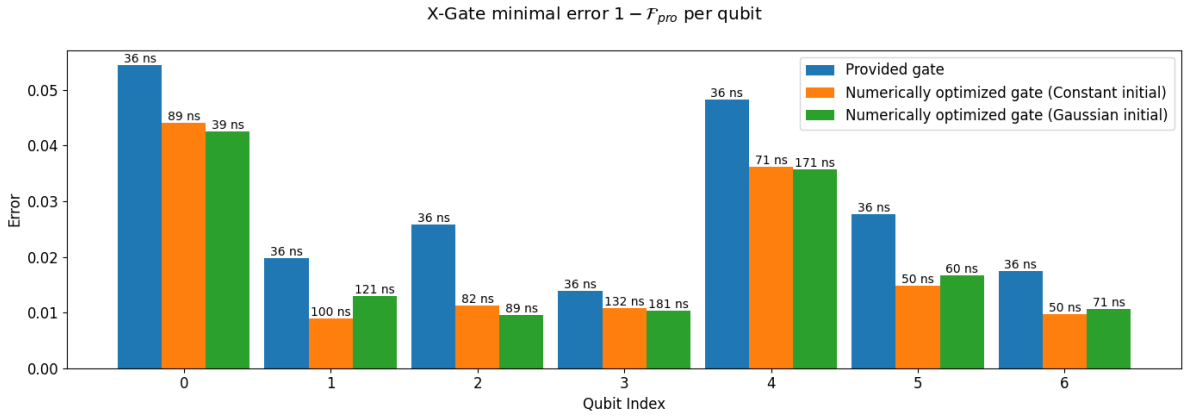


Figure 5.12.: Comparison of the error $(1 - \mathcal{F}_{pro})$ of the provided X-Gate (blue) with the optimized pulse sequences of initial constant (orange) and Gaussian shaped (green) samples. Simulation of the optimized pulses yielded error values $\leq 10^{-5}$ and are therefore not shown. The duration corresponding to the pulse with the lowest error is shown on top of each bar. The experiments were executed on *ibm_perth*.

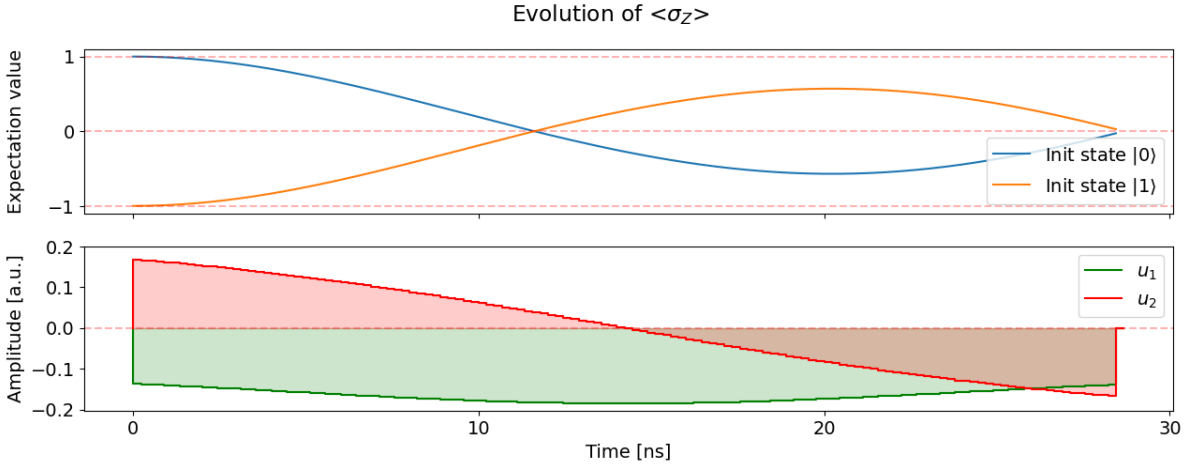


Figure 5.13.: Simulated evolution of the expectation values σ_Z for H-Gate implemented by the optimized pulse sequence of Figure 5.15a for initial states $|0\rangle$ and $|1\rangle$. The amplitudes u_1 and u_2 drive rotations about the x- and the y-axis respectively.

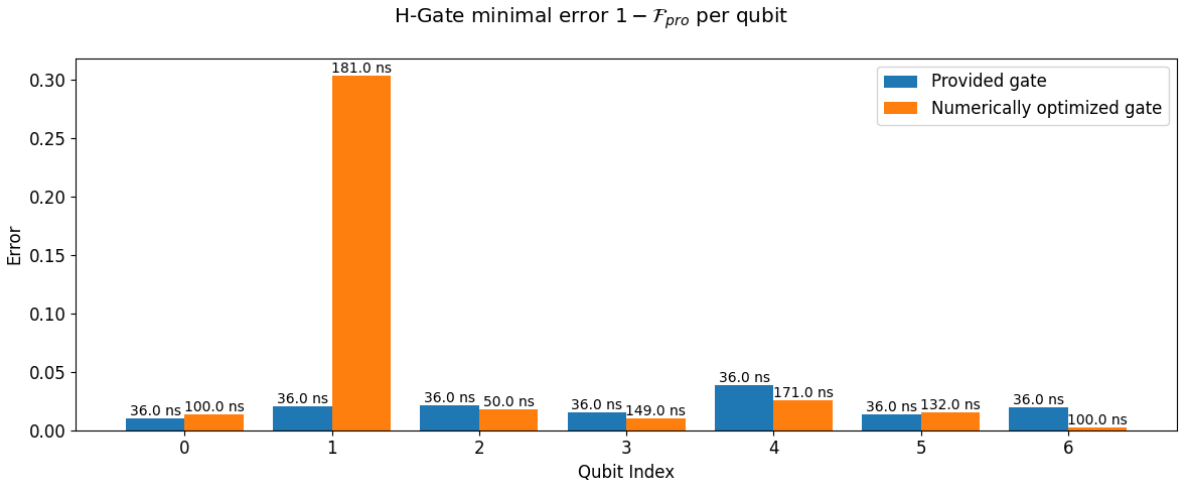
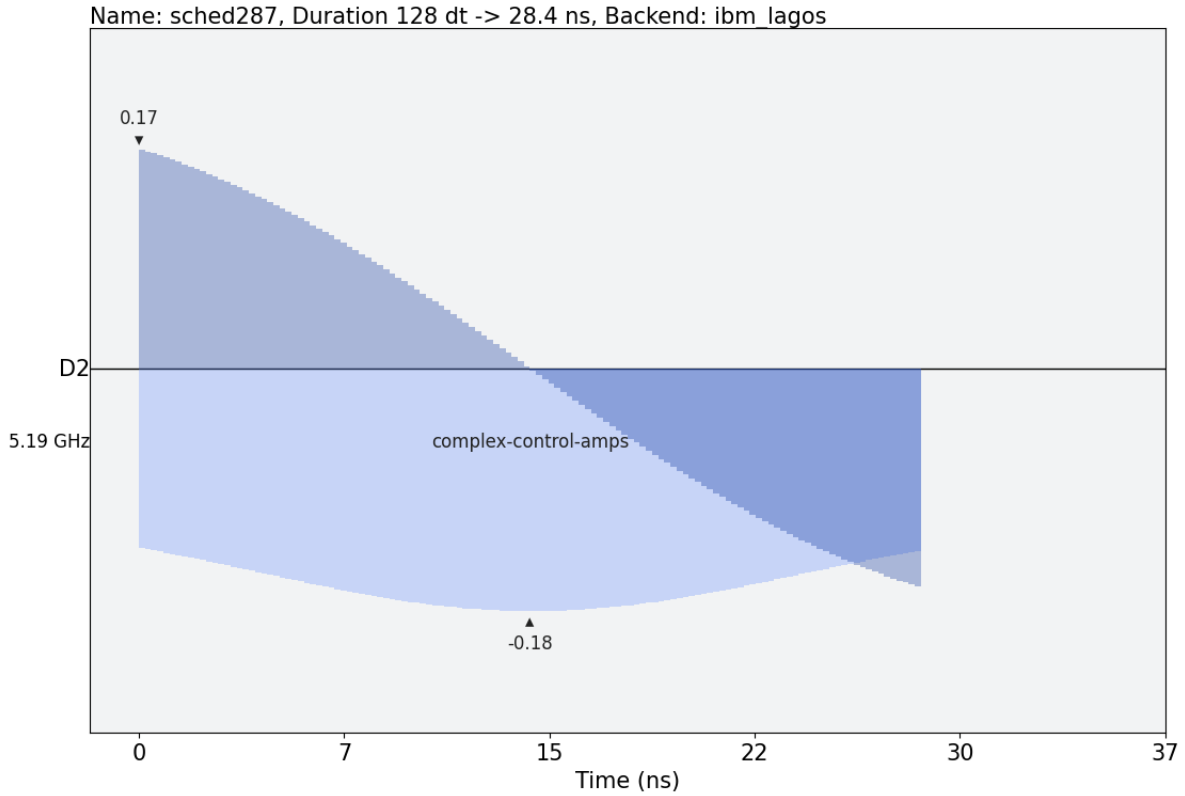
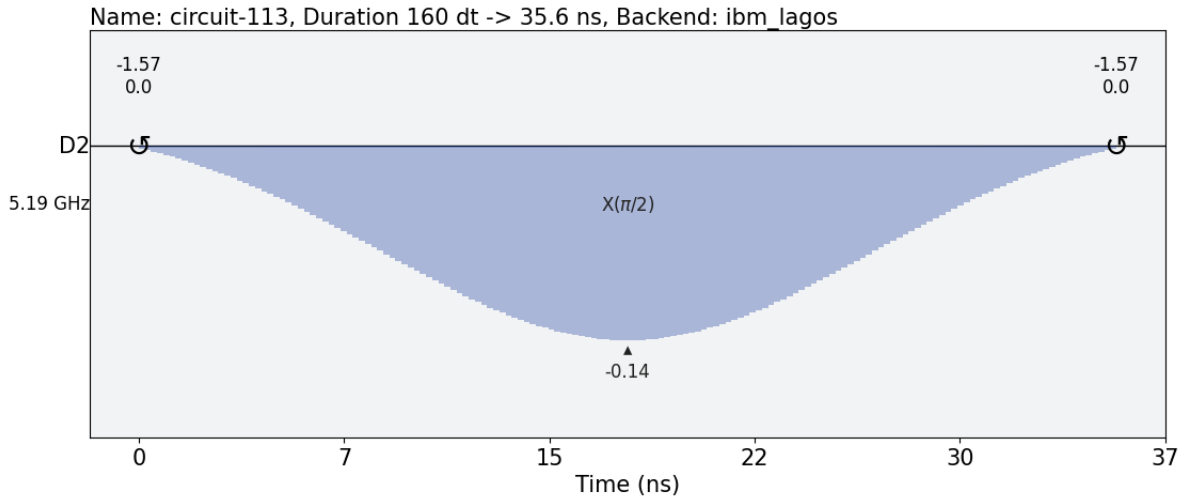


Figure 5.14.: Comparison of the error $(1 - \mathcal{F}_{pro})$ of the provided H-Gate (blue) with the optimized pulse sequences with initial samples of constant amplitude (orange). The duration corresponding to the pulse with the lowest error is shown on top of each bar. Simulation of the optimized pulses yielded error values $\leq 10^{-5}$. The experiments were executed on *ibm_lagos*.



(a) Optimized pulse sequence for a H-Gate with a duration of $T \approx 28\,\text{ns}$ achieving $\mathcal{F}_{pro} = 0.9844$.



(b) Provided H-Gate with a duration of $T \approx 36\,\text{ns}$ resulting in $\mathcal{F}_{pro} = 0.9785$.

Figure 5.15.: Optimized pulse sequence and provided H-Gate for qubit 2 on *ibm_lagos*.

6. Conclusion

6.1. Summary

During the course of this thesis, the theoretical foundation of OCT in conjunction with quantum mechanics was studied and experimentally applied on superconducting quantum computers provided by IBM Quantum. For this, a modular and extensible framework including reference implementations of the GRAPE algorithm, as well as a closed quantum system simulator was designed and implemented. The essential ideas behind the GRAPE optimization algorithm, a method to numerically optimize piecewise constant pulse sequences to achieve a target unitary evolution based on minimizing a cost function, were highlighted. This cost function evaluates the fidelity of the unitary propagation generated by the current pulse sequences against the desired target. The closed system simulator computes the final unitary by utilizing the matrix exponential of the time-independent Hamiltonian which describes the dynamics of the quantum system.

A mathematical formulation of the Hamiltonian for superconducting qubits was provided and justified from the circuitry and operating principles of quantum anharmonic oscillator. For determining the qubit specific transition and Rabi frequencies appearing in the Hamiltonian, fully automated qubit characterization experiments were implemented and executed prior to the numerical optimization. The QST and QPT experiments which enable us to assess the performance of an applied quantum channel are described. Executing the QPT experiment results in a mathematical description of the applied quantum channel, the χ–matrix, allowing us to extract a single scalar value known as process fidelity from comparison with the ideal desired channel.

Finally, we conduct QPT experiments on all qubits on a IBM Quantum 7–qubit system with numerically optimized pulse sequences implementing the single qubit X-Gate and H-Gate for a range of pulse lengths. We compare the process fidelity achieved by the optimized pulse sequences against the performance of the provided gates, demonstrating lower error rates for the X-Gate on all 7 qubits and on 4 out of 7 for the H-Gate.

6.2. Discussion and outlook

Although we have experimentally proven that optimal control can provide a path to high-fidelity quantum operations, there are still many possibilities for improving fidelities and minimizing the gate durations even further. In our optimizations we use the qubit abstraction of the Hamiltonian, which is a simplified model of the Transmon not taking into account the possibility of transitions onto higher level states. One idea to suppress these leakage effects is the addition of penalty to the cost function used in the optimization

algorithm, requiring a more realistic model of the Transmon. The analysis that led to the invention of the DRAG pulse shape [51] can be used as a starting point to gain a more thorough understanding of the possibilities to counteract such effects. Another idea to prevent leakage could be a classical post-processing approach in which the optimized amplitudes are Fourier transformed and frequency filtered before being applied to the target qubit. This could negate the influence of undesired frequency stimulation as seen in Section 5.1.1.

A second simplification that was taken in the system model is the omission of qubit-qubit coupling effects. The effect of neighboring coupled qubits on the targeted qubits need to be characterized and evaluated in regard to their influence on the achieved fidelities. Adding additional spectroscopy experiments to quantify the coupling strength between adjacent qubits and incorporating the results into the Hamiltonian prior to the optimization could potentially improve the achieved results. Moving to open system simulations modelling the dissipation and decoherence effects are also options to achieve more realistic models of the target systems.

An obvious next step is the extension of the optimization to multi-qubit gates capable of generating maximally entangled states. One possible approach might be the integration of the Cross-Resonance Hamiltonian (2.50) into the optimization scheme, introducing the required ZX interaction between two qubits. Alternative strategies utilizing the XX and YY coupling of the Jaynes-Cummings Hamiltonian (2.47) to generate entanglement could also result in the desired interactions.

Due to the exponential growth in complexity once the number of qubits in the optimization rises, computational optimization schemes in the implemented system simulator become inevitable. Propagations within a single iteration, for example during the evaluation of the step size could be parallelized to significantly decrease the runtime of the optimization algorithm. Other approaches include the usage of pre-compiled functions or the utilization of the GPU for matrix multiplications. Another idea is to use other already existing quantum simulation libraries, such as `q-optimize` or `qutip`, which already provide heavily optimized simulators also supporting open system simulations.

The techniques to extract the process fidelities using QST and QPT rely on initial state preparation which is done with the provided gate set. This induces consistency problems, as we cannot assume the initial states to match our expectations perfectly. These drawbacks with tomography experiments are covered in detail in [39] and [53], which also propose adaptions to the existing tomography experiments which should be considered in future work.

7. References

[1] Konrad Zuse and H. Wössner. *The Computer - My Life*. Berlin, Heidelberg: Springer Berlin Heidelberg, 1993. ISBN: 978-3-662-02931-2.

[2] Gordon E. Moore. "Cramming More Components onto Integrated Circuits, Reprinted from Electronics, Volume 38, Number 8, April 19, 1965, Pp.114 Ff." In: *IEEE Solid-State Circuits Society Newsletter* 11.3 (Sept. 2006), pp. 33–35. ISSN: 1098-4232. DOI: 10.1109/N-SSC.2006.4785860. URL: http://ieeexplore.ieee.org/document/4785860/ (visited on 01/28/2023).

[3] *Rechenleistung der leistungsstärksten Supercomputer weltweit im November 2022*. Statista. URL: https://de-statista-com.ezproxy.bib.hm.edu/statistik/daten/studie/193104/umfrage/rechenleistung-der-leistungsstaerksten-supercomputer-weltweit/ (visited on 01/28/2023).

[4] Richard P. Feynman. "Simulating Physics with Computers." In: *International Journal of Theoretical Physics* 21.6-7 (June 1982), pp. 467–488. ISSN: 0020-7748, 1572-9575. DOI: 10.1007/BF02650179. URL: http://link.springer.com/10.1007/BF02650179 (visited on 01/28/2023).

[5] *IBM Quantum*. IBM Quantum. URL: https://quantum-computing.ibm.com/ (visited on 01/28/2023).

[6] J. A. Jones, R. H. Hansen, and M. Mosca. "Quantum Logic Gates and Nuclear Magnetic Resonance Pulse Sequences." In: *Journal of Magnetic Resonance* 135.2 (Dec. 1998), pp. 353–360. ISSN: 10907807. DOI: 10.1006/jmre.1998.1606. arXiv: quant-ph/9805070. URL: http://arxiv.org/abs/quant-ph/9805070 (visited on 01/28/2023).

[7] Gavin K. Brennen et al. "Quantum Logic Gates in Optical Lattices." In: *Physical Review Letters* 82.5 (Feb. 1, 1999), pp. 1060–1063. ISSN: 0031-9007, 1079-7114. DOI: 10.1103/PhysRevLett.82.1060. arXiv: quant-ph/9806021. URL: http://arxiv.org/abs/quant-ph/9806021 (visited on 01/28/2023).

[8] Jerry M. Chow et al. "Complete Universal Quantum Gate Set Approaching Fault-Tolerant Thresholds with Superconducting Qubits." In: *Physical Review Letters* 109.6 (Aug. 9, 2012), p. 060501. ISSN: 0031-9007, 1079-7114. DOI: 10.1103/PhysRevLett.109.060501. arXiv: 1202.5344 [cond-mat, physics:quant-ph]. URL: http://arxiv.org/abs/1202.5344 (visited on 01/28/2023).

[9] Avner Friedman. *Foundations of Modern Analysis*. New York: Dover, 1982. 250 pp. ISBN: 978-0-486-64062-4.

[10] Richard R. Ernst, Geoffrey Bodenhausen, and Alexander Wokaun. *Principles of Nuclear Magnetic Resonance in One and Two Dimensions*. Repr. The International Series of Monographs on Chemistry 14. Oxford: Clarendon Press, 2004. 610 pp. ISBN: 978-0-19-855647-3.

[11] Roger A. Horn and Charles R. Johnson. *Matrix Analysis*. 2nd ed. Cambridge ; New York: Cambridge University Press, 2012. 643 pp. ISBN: 978-0-521-83940-2.

[12] Andreas Spörl. "Numerische Und Analytische Lösungen Für Quanteninformatisch-relevante Probleme." TU München. URL: https://mediatum.ub.tum.de/doc/651334/651334.pdf.

[13] Christopher J. Wood, Jacob D. Biamonte, and David G. Cory. *Tensor Networks and Graphical Calculus for Open Quantum Systems*. May 7, 2015. arXiv: 1111.6950 [quant-ph]. URL: http://arxiv.org/abs/1111.6950 (visited on 12/29/2022).

[14] William C. Waterhouse. "Algebraic Matrix Groups." In: *Introduction to Affine Group Schemes*. Red. by C. C. Moore. Vol. 66. Graduate Texts in Mathematics. New York, NY: Springer New York, 1979, pp. 28–35. ISBN: 978-1-4612-6219-0 978-1-4612-6217-6. DOI: 10.1007/978-1-4612-6217-6_4. URL: http://link.springer.com/10.1007/978-1-4612-6217-6_4 (visited on 01/16/2023).

[15] G. G. A. Bäuerle, E. A. de Kerf, and A. P. E. ten Kroode. *Lie Algebras: Finite and Infinite Dimensional Lie Algebras and Applications in Physics*. Studies in Mathematical Physics v. 1, 7. Amsterdam, The Netherlands : New York, N.Y: North-Holland ; Sole distributors for the U.S.A. and Canada, Elsevier Science Pub. Co, 1990. 2 pp. ISBN: 978-0-444-88776-4 978-0-444-82836-1.

[16] Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information*. 10th anniversary ed. Cambridge ; New York: Cambridge University Press, 2010. 676 pp. ISBN: 978-1-107-00217-3.

[17] Christopher J. Wood. "Initialization and Characterization of Open Quantum Systems." UWSpace, 2015. URL: http://hdl.handle.net/10012/9557.

[18] Y. Nakamura, Yu A. Pashkin, and J. S. Tsai. "Coherent Control of Macroscopic Quantum States in a Single-Cooper-pair Box." In: *Nature* 398.6730 (Apr. 1999), pp. 786–788. ISSN: 0028-0836, 1476-4687. DOI: 10.1038/19718. arXiv: cond-mat/9904003. URL: http://arxiv.org/abs/cond-mat/9904003 (visited on 01/03/2023).

[19] *Quantum-Centric Supercomputing: The next Wave of Computing — IBM Research Blog*. URL: https://research.ibm.com/blog/next-wave-quantum-centric-supercomputing (visited on 01/03/2023).

[20] *Quantum Hardware from Google*. Google Quantum AI. URL: https://quantumai.google/hardware (visited on 01/18/2023).

[21] Philip Krantz et al. "A Quantum Engineer's Guide to Superconducting Qubits." In: *Applied Physics Reviews* 6.2 (June 2019), p. 021318. ISSN: 1931-9401. DOI: 10.1063/1.5089550. arXiv: 1904.06560 [physics:physics, physics:quant-ph]. URL: http://arxiv.org/abs/1904.06560 (visited on 09/07/2022).

[22] He-Liang Huang et al. *Superconducting Quantum Computing: A Review*. Nov. 2, 2020. arXiv: 2006.10433 [quant-ph]. URL: http://arxiv.org/abs/2006.10433 (visited on 01/03/2023).

[23] Alexandre Blais et al. "Circuit Quantum Electrodynamics." In: *Reviews of Modern Physics* 93.2 (May 19, 2021), p. 025005. ISSN: 0034-6861, 1539-0756. DOI: 10.1103/RevModPhys.93.025005. arXiv: 2005.12667 [quant-ph]. URL: http://arxiv.org/abs/2005.12667 (visited on 11/07/2022).

[24] 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. ISSN: 0018-9219. DOI: 10.1109/PROC.1963.1664. URL: http://ieeexplore.ieee.org/document/1443594/ (visited on 01/04/2023).

[25] Jerry M. Chow et al. "A Simple All-Microwave Entangling Gate for Fixed-Frequency Superconducting Qubits." In: *Physical Review Letters* 107.8 (Aug. 17, 2011), p. 080502. ISSN: 0031-9007, 1079-7114. DOI: 10.1103/PhysRevLett.107.080502. arXiv: 1106.0553 [cond-mat, physics:quant-ph]. URL: http://arxiv.org/abs/1106.0553 (visited on 10/27/2022).

[26] Jonathan P. Dowling and Gerard J. Milburn. *Quantum Technology: The Second Quantum Revolution*. June 13, 2002. arXiv: quant-ph/0206091. URL: http://arxiv.org/abs/quant-ph/0206091 (visited on 01/07/2023).

[27] F. Albertini and D. D'Alessandro. "Notions of Controllability for Bilinear Multilevel Quantum Systems." In: *IEEE Transactions on Automatic Control* 48.8 (Aug. 2003), pp. 1399–1403. ISSN: 0018-9286. DOI: 10.1109/TAC.2003.815027. URL: http://ieeexplore.ieee.org/document/1220755/ (visited on 01/18/2023).

[28] Daoyi Dong and Ian R. Petersen. "Quantum Control Theory and Applications: A Survey." In: *IET Control Theory & Applications* 4.12 (Dec. 1, 2010), pp. 2651–2671. ISSN: 1751-8644, 1751-8652. DOI: 10.1049/iet-cta.2009.0508. arXiv: 0910.2350. URL: http://arxiv.org/abs/0910.2350 (visited on 03/25/2022).

[29] Garng M. Huang, T. J. Tarn, and John W. Clark. "On the Controllability of Quantum-mechanical Systems." In: *Journal of Mathematical Physics* 24.11 (Nov. 1983), pp. 2608–2618. ISSN: 0022-2488, 1089-7658. DOI: 10.1063/1.525634. URL: http://aip.scitation.org/doi/10.1063/1.525634 (visited on 01/18/2023).

[30] W. S. Warren, H. Rabitz, and M. Dahleh. "Coherent Control of Quantum Dynamics: The Dream Is Alive." In: *Science* 259.5101 (Mar. 12, 1993), pp. 1581–1589. ISSN: 0036-8075, 1095-9203. DOI: 10.1126/science.259.5101.1581. URL: https://www.sciencemag.org/lookup/doi/10.1126/science.259.5101.1581 (visited on 01/18/2023).

[31] Hideo Mabuchi and Navin Khaneja. "Principles and Applications of Control in Quantum Systems." In: *International Journal of Robust and Nonlinear Control* 15.15 (Oct. 2005), pp. 647–667. ISSN: 1049-8923, 1099-1239. DOI: 10.1002/rnc.1016. URL: https://onlinelibrary.wiley.com/doi/10.1002/rnc.1016 (visited on 01/18/2023).

[32] Steffen J. Glaser et al. "Training Schrödinger's Cat: Quantum Optimal Control: Strategic Report on Current Status, Visions and Goals for Research in Europe." In: *The European Physical Journal D* 69.12 (Dec. 2015), p. 279. ISSN: 1434-6060, 1434-6079. DOI: 10.1140/epjd/e2015-60464-1. URL: http://link.springer.com/10.1140/epjd/e2015-60464-1 (visited on 01/18/2023).

[33] David J. Tannor and Stuart A. Rice. "Control of Selectivity of Chemical Reaction via Control of Wave Packet Evolution." In: *The Journal of Chemical Physics* 83.10 (Nov. 15, 1985), pp. 5013–5018. ISSN: 0021-9606, 1089-7690. DOI: 10.1063/1.449767. URL: http://aip.scitation.org/doi/10.1063/1.449767 (visited on 01/18/2023).

[34] Shenghua Shi and Herschel Rabitz. "Quantum Mechanical Optimal Control of Physical Observables in Microsystems." In: *The Journal of Chemical Physics* 92.1 (Jan. 1990), pp. 364–376. ISSN: 0021-9606, 1089-7690. DOI: 10.1063/1.458438. URL: http://aip.scitation.org/doi/10.1063/1.458438 (visited on 01/18/2023).

[35] Navin Khaneja et al. "Optimal Control of Coupled Spin Dynamics: Design of NMR Pulse Sequences by Gradient Ascent Algorithms." In: *Journal of Magnetic Resonance* 172.2 (Feb. 2005), pp. 296–305. ISSN: 10907807. DOI: 10.1016/j.jmr.2004.11.004. URL: https://linkinghub.elsevier.com/retrieve/pii/S1090780704003696 (visited on 07/07/2022).

[36] Ingmar Immanuel Amadeus Landgraf. "Grape4Space - feasibility study for satellite position control using a gradient ascent pulse engineering algorithm." TUM München, 2020.

[37] G. Lindblad. "On the Generators of Quantum Dynamical Semigroups." In: *Communications in Mathematical Physics* 48.2 (June 1976), pp. 119–130. ISSN: 0010-3616, 1432-0916. DOI: 10.1007/BF01608499. URL: http://link.springer.com/10.1007/BF01608499 (visited on 01/18/2023).

[38] *Quantum Process Tomography — Forest-Benchmarking 0.6.0 Documentation*. URL: https://forest-benchmarking.readthedocs.io/en/latest/examples/tomography_process.html (visited on 12/30/2022).

[39] Daniel Greenbaum. *Introduction to Quantum Gate Set Tomography*. Sept. 9, 2015. arXiv: 1509.02921 [quant-ph]. URL: http://arxiv.org/abs/1509.02921 (visited on 12/30/2022).

[40] Yeong-Cherng Liang et al. "Quantum Fidelity Measures for Mixed States." In: *Reports on Progress in Physics* 82.7 (July 1, 2019), p. 076001. ISSN: 0034-4885, 1361-6633. DOI: 10.1088/1361-6633/ab1ca4. arXiv: 1810.08034 [quant-ph]. URL: http://arxiv.org/abs/1810.08034 (visited on 01/03/2023).

[41] Ramesh Bhandari and Nicholas A. Peters. *On Single Qubit Quantum Process Tomography for Trace-Preserving and Nontrace-Preserving Maps*. Feb. 3, 2015. arXiv: 1502.01016 [quant-ph]. URL: http://arxiv.org/abs/1502.01016 (visited on 01/02/2023).

[42] Nathaniel Johnston and David W. Kribs. "Quantum Gate Fidelity in Terms of Choi Matrices." In: *Journal of Physics A: Mathematical and Theoretical* 44.49 (Dec. 9, 2011), p. 495303. ISSN: 1751-8113, 1751-8121. DOI: 10.1088/1751-8113/44/49/495303. arXiv: 1102.0948 [quant-ph]. URL: http://arxiv.org/abs/1102.0948 (visited on 01/25/2023).

[43] Easwar Magesan, Jay M. Gambetta, and Joseph Emerson. "Characterizing Quantum Gates via Randomized Benchmarking." In: *Physical Review A* 85.4 (Apr. 11, 2012), p. 042311. ISSN: 1050-2947, 1094-1622. DOI: 10.1103/PhysRevA.85.042311. arXiv: 1109.6887 [quant-ph]. URL: http://arxiv.org/abs/1109.6887 (visited on 01/24/2023).

[44] Charles R. Harris et al. "Array Programming with NumPy." In: *Nature* 585.7825 (Sept. 2020), pp. 357–362. DOI: 10.1038/s41586-020-2649-2. URL: https://doi.org/10.1038/s41586-020-2649-2.

[45] Awad H. Al-Mohy and Nicholas J. Higham. "A New Scaling and Squaring Algorithm for the Matrix Exponential." In: *SIAM Journal on Matrix Analysis and Applications* 31.3 (Jan. 2010), pp. 970–989. ISSN: 0895-4798, 1095-7162. DOI: 10.1137/09074721X. URL: http://epubs.siam.org/doi/10.1137/09074721X (visited on 01/23/2023).

[46] David C. McKay et al. *Qiskit Backend Specifications for OpenQASM and OpenPulse Experiments*. Sept. 10, 2018. arXiv: 1809.03452 [quant-ph]. URL: http://arxiv.org/abs/1809.03452 (visited on 09/19/2022).

[47] Andrew W. Cross et al. "OpenQASM 3: A Broader and Deeper Quantum Assembly Language." In: *ACM Transactions on Quantum Computing* 3.3 (Sept. 30, 2022), pp. 1–50. ISSN: 2643-6809, 2643-6817. DOI: 10.1145/3505636. arXiv: 2104.14722 [quant-ph]. URL: http://arxiv.org/abs/2104.14722 (visited on 01/23/2023).

[48] A-tA-v et al. *Qiskit: An Open-Source Framework for Quantum Computing*. 2021. DOI: 10.5281/zenodo.2573505.

[49] *Building Pulse Schedules - Tutorial*. URL: https://qiskit.org/documentation/tutorials/circuits_advanced/06_building_pulse_schedules.html (visited on 01/13/2023).

[50] F. Motzoi et al. "Simple Pulses for Elimination of Leakage in Weakly Nonlinear Qubits." In: *Physical Review Letters* 103.11 (Sept. 8, 2009), p. 110501. ISSN: 0031-9007, 1079-7114. DOI: 10.1103/PhysRevLett.103.110501. arXiv: 0901.0534 [cond-mat, physics:quant-ph]. URL: http://arxiv.org/abs/0901.0534 (visited on 01/14/2023).

[51] 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. 18, 2011), p. 012308. ISSN: 1050-2947, 1094-1622. DOI: 10.1103/PhysRevA.83.012308. arXiv: 1011.1949 [cond-mat, physics:quant-ph]. URL: http://arxiv.org/abs/1011.1949 (visited on 01/14/2023).

[52] Norman F. Ramsey. “A Molecular Beam Resonance Method with Separated Oscillating Fields.” In: *Physical Review* 78.6 (June 1950), pp. 695–699. DOI: 10.1103/PhysRev.78.695. URL: https://link.aps.org/doi/10.1103/PhysRev.78.695.

[53] Seth T. Merkel et al. “Self-Consistent Quantum Process Tomography.” In: *Physical Review A* 87.6 (June 24, 2013), p. 062119. ISSN: 1050-2947, 1094-1622. DOI: 10.1103/PhysRevA.87.062119. arXiv: 1211.0322 [quant-ph]. URL: http://arxiv.org/abs/1211.0322 (visited on 12/30/2022).

Appendix

A. UML Diagrams

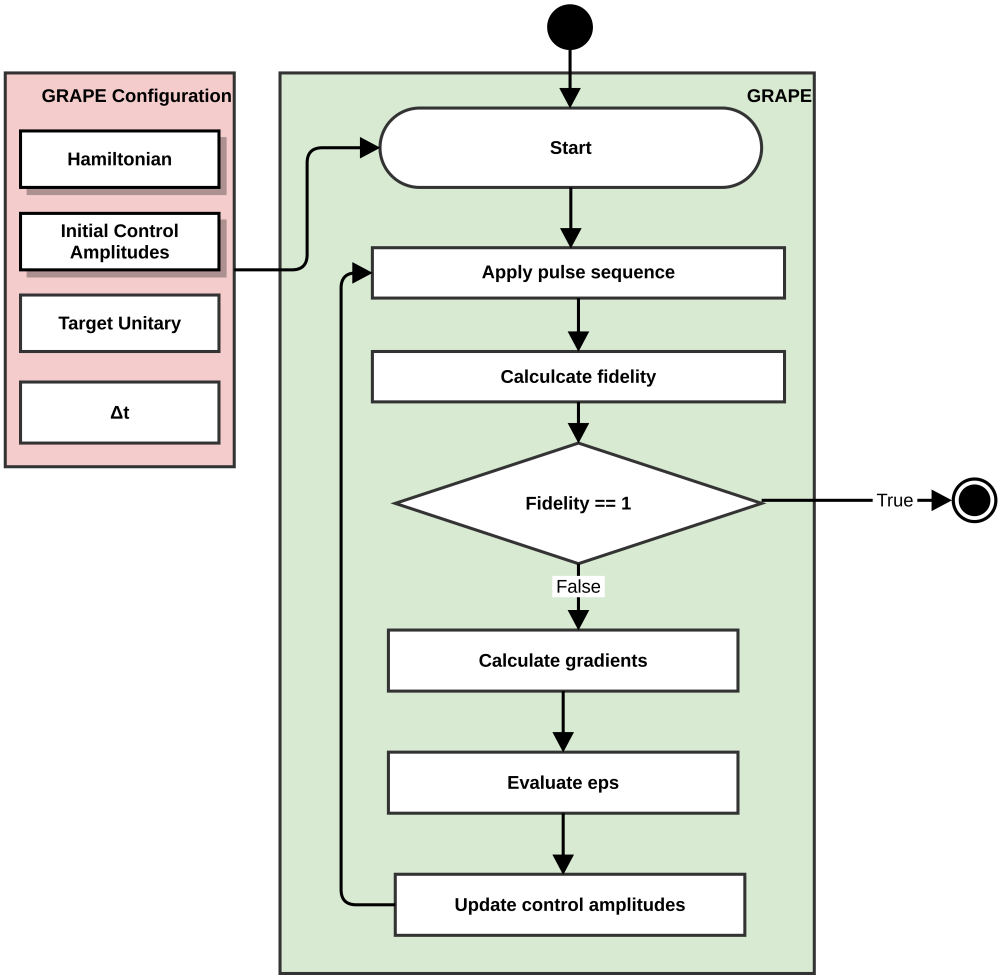


Figure 1.: GRAPE flowchart

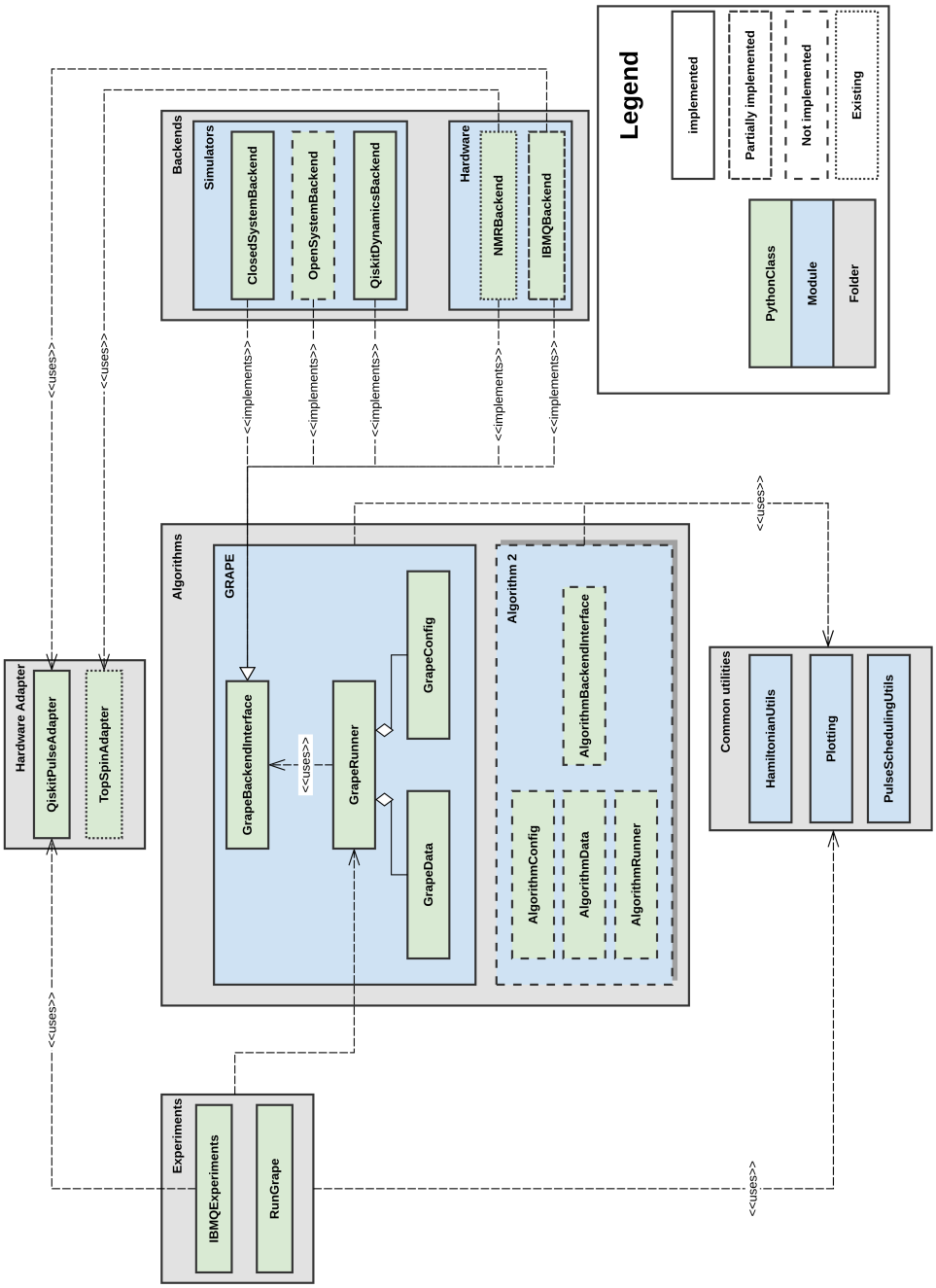


Figure 2.: Software Architecture

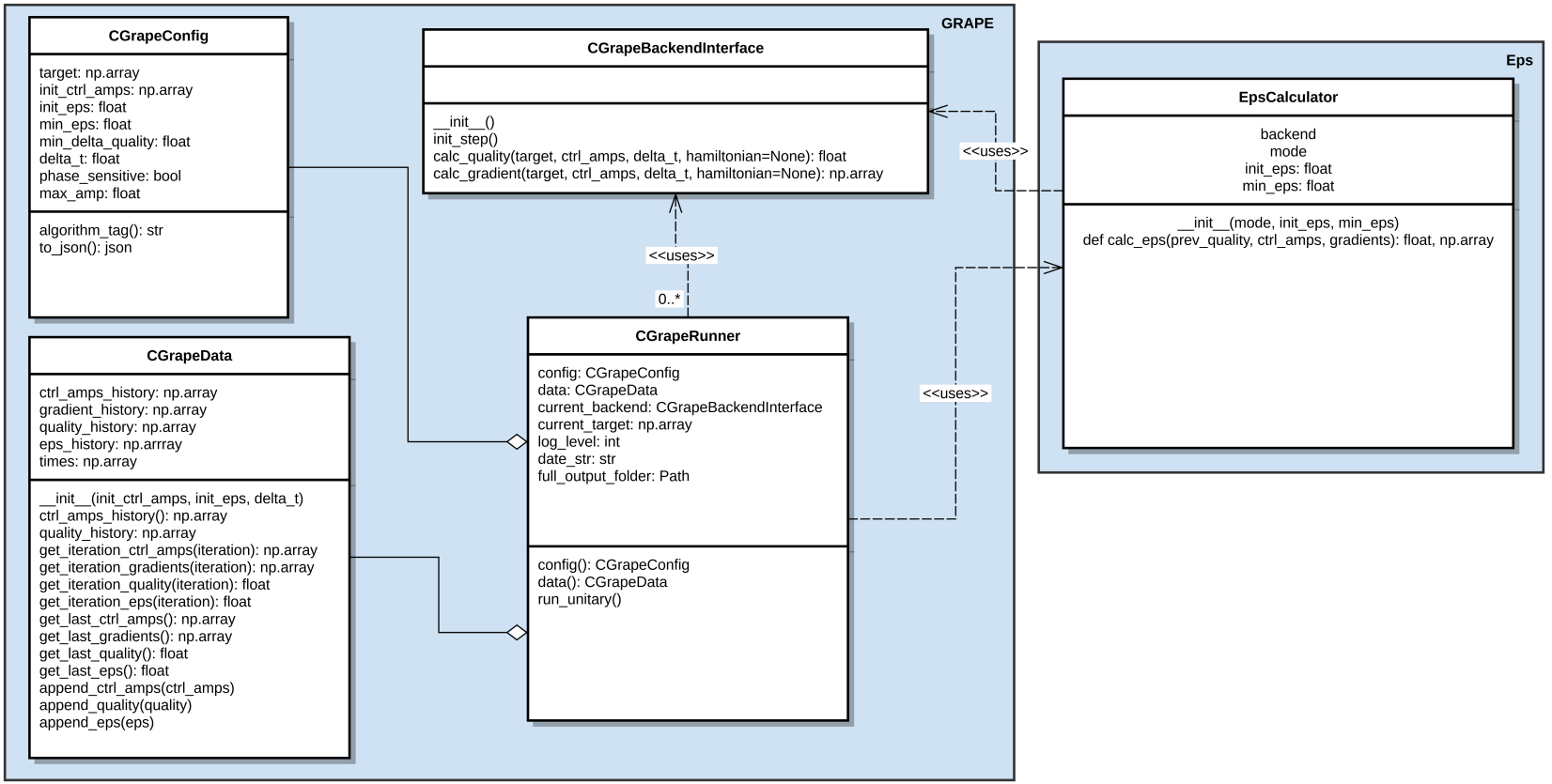


Figure 3.: Grape Algorithm Class Diagram

B. Qubit drive amplitude duration relation diagrams

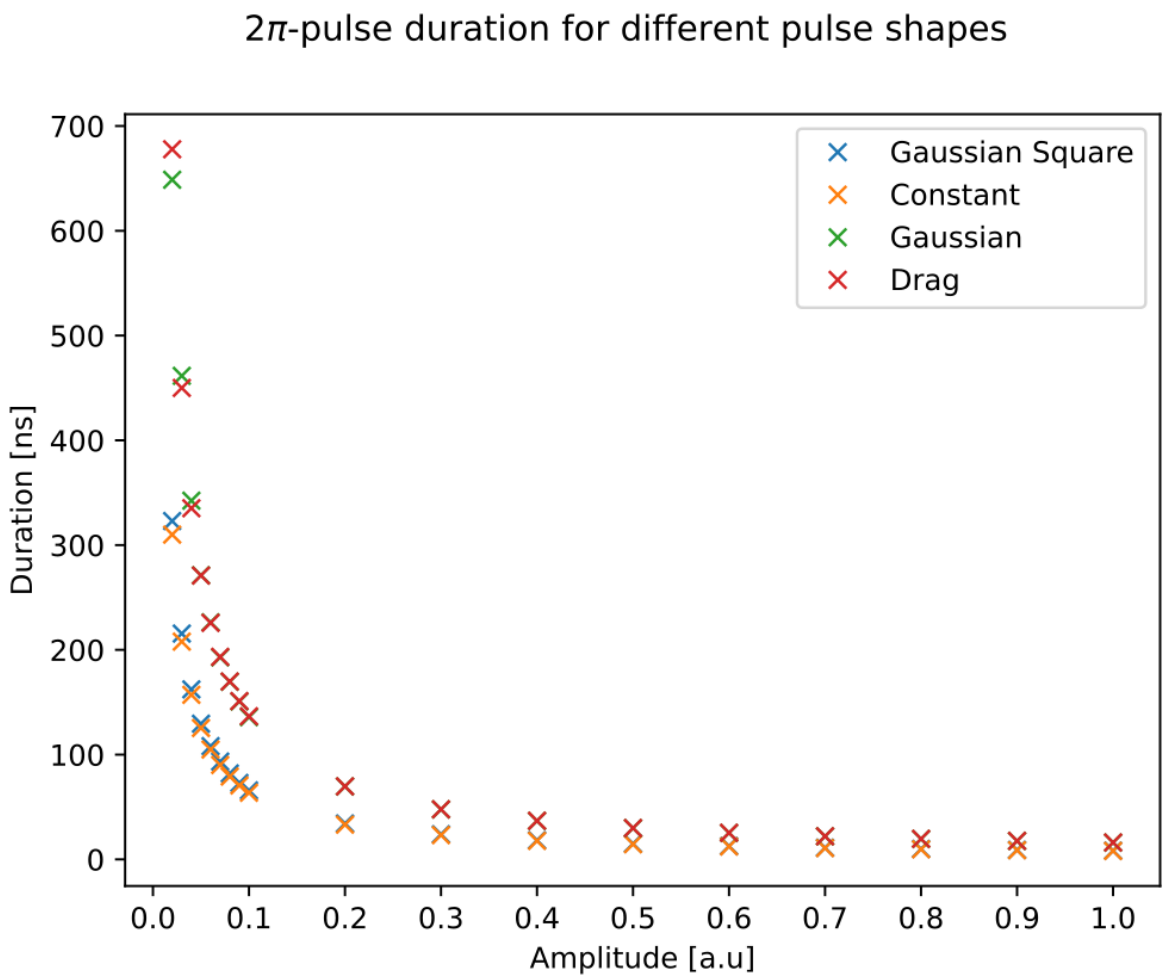


Figure 4.: Periods of sinus fits extracted from single Rabi experiments with varying duration and constant amplitude per experiment. Each data point represents a single duration for which a drive of the given amplitude results in a 2π rotation about the x-axis of the Bloch sphere.

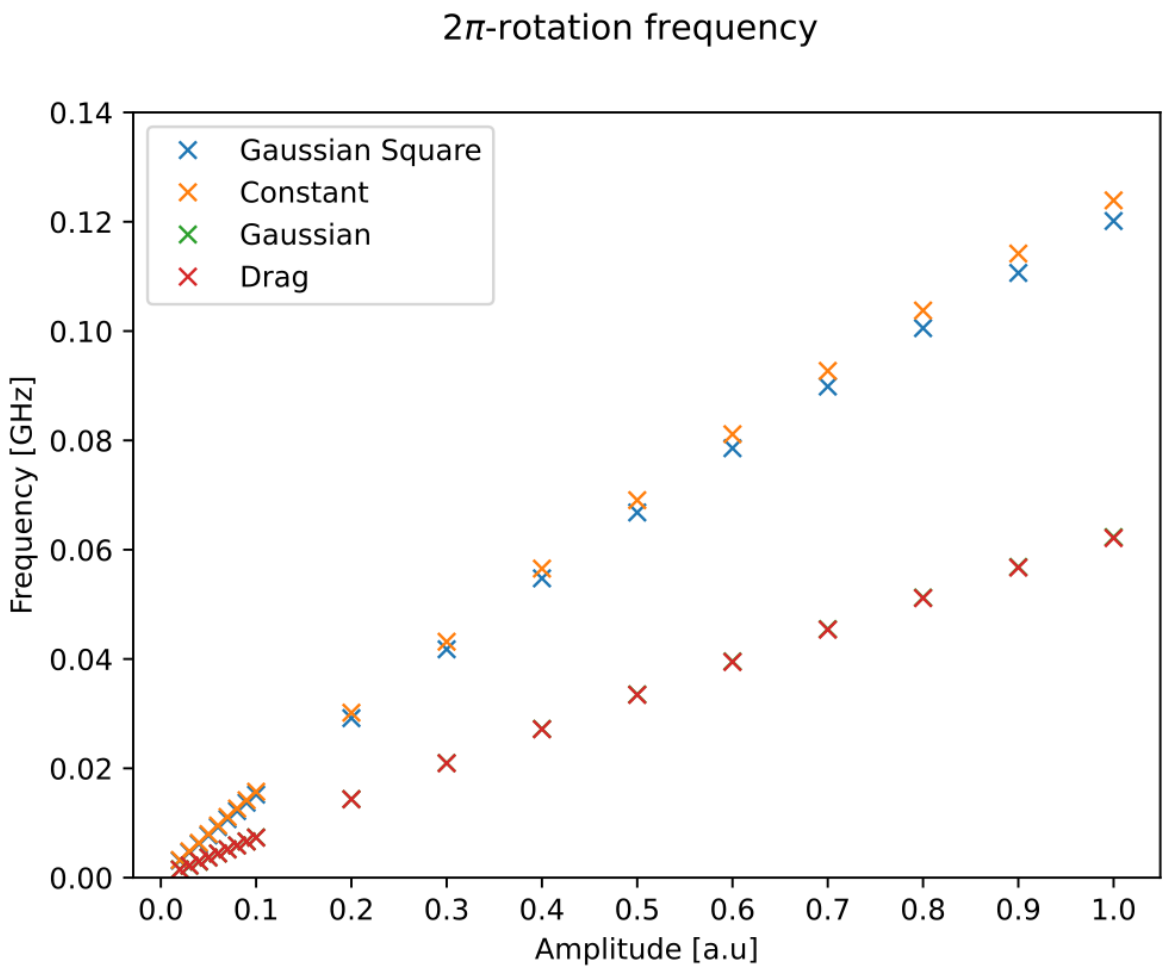


Figure 5.: Frequencies $f_{2\pi} = \frac{1}{t_{2\pi}}$ calculated from data in Figure 4

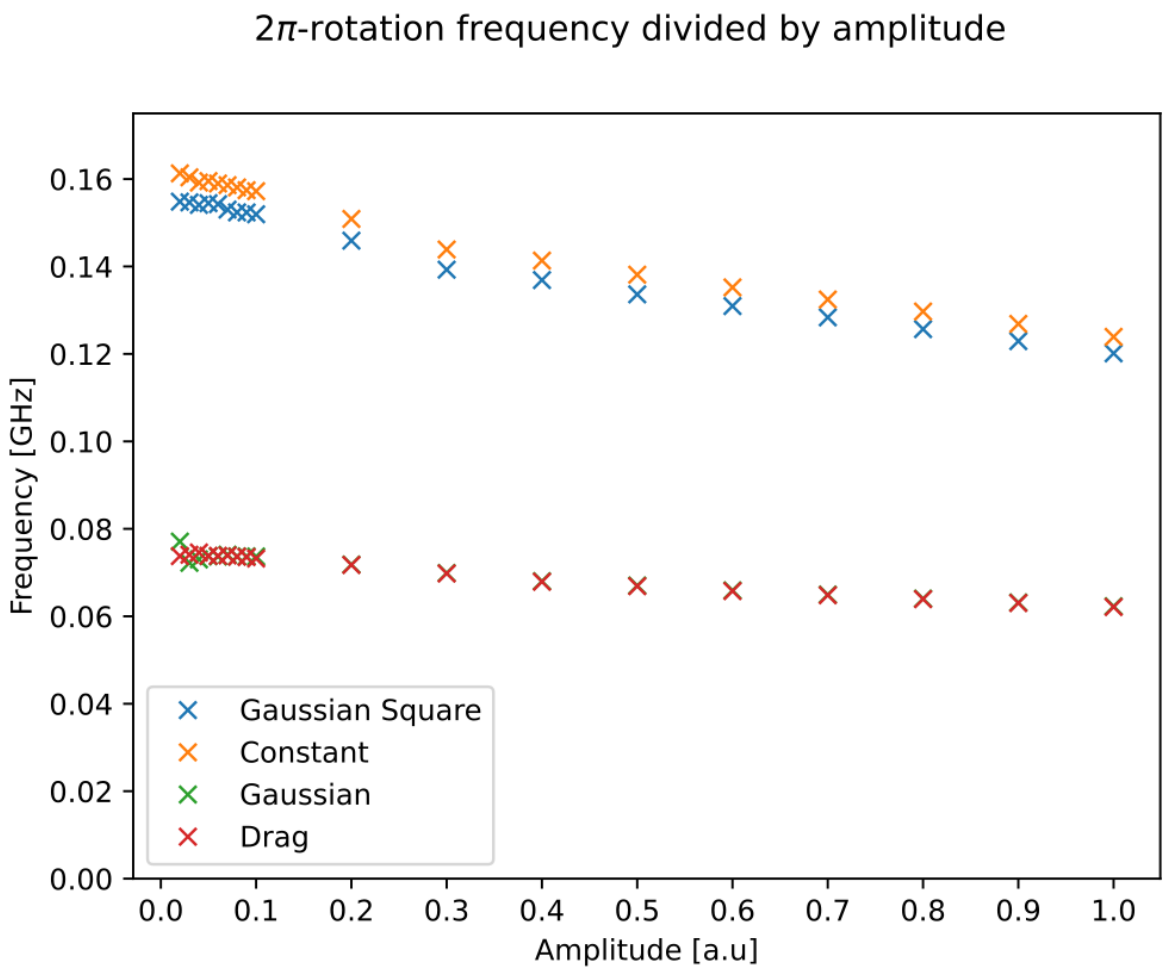


Figure 6.: Rabi frequency calculated from data in Figure 4 via (5.1)