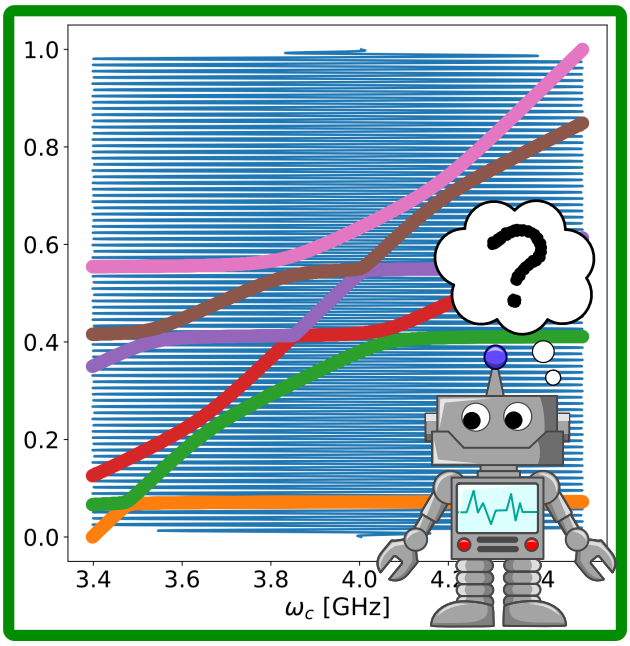
CHALMERS
UNIVERSITY OF TECHNOLOGY



A Comparison of Quantum Gate Optimization Techniques

Deep Reinforcement Learning with an Ansatz, Limited Experimental Bandwidth, and Analysis of CZ Gate Dynamics

Master's thesis in Complex Adaptive Systems

PONTUS LINDGREN

DEPARTMENT OF MICROTECHNOLOGY AND NANOSCIENCE
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2025
www.chalmers.se

Master's thesis 2025

A Comparison of Quantum Gate Optimization Techniques

Deep Reinforcement Learning with an Ansatz, Limited Experimental Bandwidth, and Analysis of CZ Gate Dynamics

Pontus Lindgren

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Microtechnology and Nanoscience
Division of Applied Quantum Physics
Chalmers University of Technology
Gothenburg, Sweden 2025

A Comparison of Quantum Gate Optimization Techniques
Deep Reinforcement Learning with an Ansatz, Limited Experimental Bandwidth, and Analysis of CZ Gate Dynamics
PONTUS LINDGREN

Supervisor: Tahereh Abad, Microtechnology and Nanoscience
Examiner: Anton Frisk Kockum, Microtechnology and Nanoscience

Master's Thesis 2025
Department of Microtechnology and Nanoscience (MC2)
Division of Applied Quantum Physics (AQP)
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: Robot thinking about optimizing a two-qubit ansatz (shown in blue) for the energy spectrum of the two-qubit set-up (shown as thick lines). The ansatz and spectrum are shown in figure 3.2a. Using deep reinforcement learning to improve the two-qubit ansatz, a minor improvement is possible, shown in figure 4.15.

Typeset in LaTeX
Printed by Chalmers Reproservice
Gothenburg, Sweden 2025

A Comparison of Quantum Gate Optimization Techniques
Deep Reinforcement Learning with an Ansatz, Limited Experimental Bandwidth, and Analysis of CZ Gate Dynamics
PONTUS LINDGREN
Department of Microtechnology and Nanoscience
Chalmers University of Technology

Abstract

Better quantum gates are likely key to enabling fault-tolerant, useful quantum computers. This thesis compares gate optimization techniques by simulating single-qubit and two-qubit gates for superconducting qubits. The primary focus is deep reinforcement learning. For single-qubit gates, the task is to optimize a π-pulse, while for two-qubit gates, the task is to optimize the Controlled-Z gate. The results indicate that using an ansatz for the gate's pulse shape can enhance the performance of deep reinforcement learning, both for single-qubit and two-qubit gates, but only significantly for single-qubit gates. A simple square-pulse ansatz approximately halves the simulation time needed to reach the coherence limit for the single-qubit gate studied. The speed-up in simulation should translate to a speed-up in experiments as well. The thesis does not find evidence that the implemented deep reinforcement learning algorithm yields better quantum gates than a state-of-the-art black-box optimizer, despite the black-box optimizer being easier to implement experimentally. For a quantum gate defined by piece-wise constant controls, a low-pass filter seems to enhance the performance, at least if the filter is considered when optimizing. This indicates that piece-wise constant controls, for example, generated with deep reinforcement learning, are not hindered by the limited bandwidth of control electronics. Finally, the study highlights the importance of ZZ coupling to understanding Controlled-Z gates.

Keywords: Quantum Optimal Control, Quantum Computing, Reinforcement Learning, Controlled-Z gate, Optimization, Gradient Ascent Pulse Engineering, Derivative Removal by Adiabatic Gate, Machine Learning

Acknowledgements

I want to express my gratitude to my supervisor, Tahereh Abad. Thanks for tailoring the project to suit my interest in quantum technology and machine learning. Your supervision turbocharged my progress. I also want to thank my examiner, Anton Frisk Kockum. Thanks for sharing your extensive research expertise and your genuine interest in my work. Together, Anton and Tahereh have made me a much better researcher. I am also grateful to the Applied Quantum Physics division for the welcoming environment. The movie nights and the lunch room discussions deserve special mention. I also wish to thank my opponent, Johanna Örgård. Your well-formulated, detailed feedback made this thesis significantly better. Last but not least, thanks to friends and family for your undying support.

Pontus Lindgren, Gothenburg, June 2025

Nomenclature

Below are the acronyms, parameters, and definitions used throughout the thesis.

Acronyms

AWG Arbitary Waweform Generator

BCH Baker-Campbell-Hausdorff

CMA-ES Covariance Matrix Adaptation Evolution Strategy

CSQR Chalmers Superconducting Qubit Repository

CZ Controlled-Z

DAC Digital-to-Analog Converter

DRAG Derivative Removal by Adiabatic Gate

DRL Deep Reinforcement Learning

GRAPE Gradient Ascent Pulse Engineering

NG Nevergrad

PWC piece-wise constant

QOC Quantum Optimal Control

SQUID Superconducting Quantum Interference Device

Parameters

t_g	Gate time

Δt	Length of piece-wise constant interval
$\hat{U}_T$	Target unitary
F	Fidelity
I	Infidelity
I_c	Infidelity from decoherence

Definitions

Hermitian operator	$\hat{H}^\dagger = \hat{H}$
Complex conjugate	$\Omega^* = \Re(\Omega) - i\Im(\Omega)$
Unitary operator	$\hat{U}^\dagger\hat{U} = \mathrm{I}$
Fidelity	A measure of the similarity of two matrices. A way to describe how good a quantum gate is.
Infidelity	$I = 1 - F$
$\|i, j, k\rangle$	Quantum state where the first qubit is in eigenstate i, the second qubit is in eigenstate j and the coupler is in eigenstate k.
Black-box optimizer	Optimization method not requiring knowledge about the gradient of the loss function.
Deep reinforcement learning	A class of algorithms used to train a neural network to output the optimal action for every input. The difference relative to a black-box optimizer is that deep reinforcement learning can output different actions for different inputs.

Contents

List of Figures

1
Introduction

If large-scale, fault-tolerant quantum computing is realized, it could have significant consequences on everything from encryption to drug design due to the promise of an algorithmic speed-up compared with today's computing [1, 2]. There are several different ways to realize quantum computing. This thesis focuses on discrete variable quantum computing implemented with superconducting circuits, the main focus of, for example, Chalmers' and Google's quantum computing efforts [3]. Alternatives exist, for example, continuous variable quantum computing with photons [4].

1.1 Discrete Variable Quantum Computing

Discrete variable quantum computing is based on qubits. A qubit is a two-level system

$$|\psi\rangle = \cos\left(\frac{\theta}{2}\right)|0\rangle + e^{i\phi}\sin\left(\frac{\theta}{2}\right)|1\rangle \tag{1.1}$$

where ϕ is the relative phase between the $|0\rangle$ and the $|1\rangle$ component of ψ and θ describes how much of $|\psi\rangle$ is made up of the pure states $|0\rangle$ and $|1\rangle$. The surface of a sphere can represent two variables. In quantum computing, the visualization of a two-level system in terms of spherical coordinates based on θ and ϕ is known as the Bloch sphere.

Multiple qubits can be combined to encode binary information as bitstrings. A simple example is the state $|\psi\rangle = |0\rangle \otimes |1\rangle$, meaning that qubit one is in $|0\rangle$ and qubit two is in $|1\rangle$. Simplifying the notation by writing $|\psi\rangle = |01\rangle$ is common. A bitstring like 01 can encode different kinds of information, for example, the number one if viewed as a decimal number. An advantage of quantum computing is that a quantum system can be in a superposition of several quantum states. This means that N qubits can represent a superposition of 2^N bitstrings.

Programs run on a discrete variable quantum computer are usually reasoned about in terms of quantum gates. A quantum gate defined by $\hat{U}_T$ changes the state of a quantum system $|\psi\rangle$ such that $|\psi_{\text{after gate}}\rangle = \hat{U}_T|\psi\rangle$, which can be understood mathematically as matrix multiplication. Gates do, in practice, not change quantum states instantaneously. An example of a quantum gate is a rotation by π on the Bloch sphere. A π-pulse corresponds to having increased θ (defined in Eq. (1.1)) by π at the end of the gate. A π-pulse is shown by the red curve in Fig. 1.1. Unfortunately, a two-level system is an approximation of qubits based on superconducting circuits. If one is not careful, the quantum system can leak into higher excitation levels, such as $|2\rangle$. This is one reason why, for example, a π-pulse can be challenging to implement

correctly. The green curve in Fig. 1.1 visualizes a suboptimal π-pulse. Imperfect quantum gates limit today's quantum computers. For superconducting qubits, gates are implemented using microwave pulses. This master's thesis investigates how to optimize those pulses.

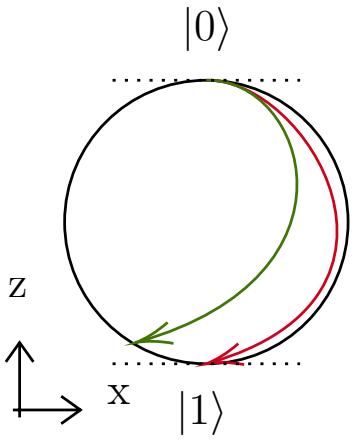


Figure 1.1: Visualization of a π-pulse on the XZ plane of the Bloch sphere. ($z = 1, x = 0$) corresponds to $|0\rangle$. The red curve visualizes an optimized π-pulse rotating the quantum state by π such that $|0\rangle \rightarrow |1\rangle$ while the green curve visualizes a suboptimal π-pulse.

Quantum gate optimization techniques can often be divided into two categories: model-based and model-free techniques. Model-based approaches rely on a theoretical model of the quantum system, while model-free techniques do not need a theoretical model. Instead, model-free approaches optimize the quantum gates by interacting with the hardware. Model-free approaches are interesting because modeling quantum computer hardware accurately enough to enable large-scale, fault-tolerant quantum computing is challenging. An advantage of model-based methods is that they are often faster to implement. This study considers two model-based approaches: Gradient Ascent Pulse Engineering and Derivative Removal by Adiabatic Gate. A way to combine model-free and model-based optimization methods is to use an ansatz. An ansatz is an educated guess used to initialize the optimization procedure. The ansatz is often based on a theoretical model of the qubit hardware. An ansatz could benefit model-free approaches like deep reinforcement learning because it could lead to faster convergence, so less experimental data needs to be collected.

1.2 Quantum Error Correction

How optimization of quantum gates could help realize large-scale, fault-tolerant quantum computing can be understood in terms of quantum error correction. The idea behind quantum error correction is to encode quantum information in multiple qubits to suppress the error rate of quantum gates exponentially. Using three qubits, a logical 1 could, for example, be defined $|1_L\rangle = |111\rangle$. Google recently showed that quantum error correction can be implemented experimentally to suppress error

rates exponentially [3]. They used the surface code [5], a specific way to realize error correction in quantum computers. A well-established model of the logical qubit's error rate ϵ_d given the error rate of the physical qubit p is

$$\epsilon_d \propto \left(\frac{p}{p_{\text{thr}}}\right)^{\frac{d+1}{2}}, \tag{1.2}$$

where p_{thr} is the threshold error-rate (a constant), and d is the distance of the surface code encoding. As a rule of thumb p_{thr} is a bit below 1 %. The number of physical qubits needed to achieve a logical qubit of distance d is $n_p = 2d^2 - 1$. Scaling the number of qubits is an engineering challenge, typically today limited to a few hundred qubits for superconducting quantum computers. This represents a trade-off between scale and fault tolerance, as a single high-performing logical qubit would require a lot of physical qubits. By decreasing the physical error rate p, one can achieve a logical qubit with the same error rate using fewer qubits, meaning that the trade-off is less restrictive. One way to decrease p is to optimize the control of the qubits by optimizing the control pulses, which this master's thesis focuses on.

1.3 Purpose

The purpose of this thesis is to compare different quantum gate optimization techniques through simulations. Several model-free and model-based techniques are compared. The primary focus is on deep reinforcement learning, comparing different ways to use an ansatz and different ways to use the state of the quantum system for gate optimization. Experimental implementation is also considered in the simulations, more specifically, the limited bandwidth of control electronics. The dynamics of the optimized quantum gates are also analyzed. Some novelty is introduced, but most of the techniques compared already exist in the literature. The simulated quantum devices in this thesis are intended to mimic the superconducting quantum computing hardware at Chalmers. Both a single-qubit device and a two-qubit device are simulated. Additionally, the thesis tries to give a broad, accessible overview of quantum gate optimization.

Comparing quantum gate optimization techniques on the same set-up is valuable, as optimization of quantum gates is a large and diverse field. [6] suggests a different way than [7, 8] to use ansatzes when optimizing quantum gates, while [9] does not use an ansatz at all. [9] uses the quantum state of the system for two-qubit gate optimization with deep reinforcement learning, while [8] does not use the state. Moreover, many different superconducting quantum computing hardware exist, [8] uses fluxonium qubits while [10] uses transmon qubits. The qubits can be connected in different ways, [10] uses fixed frequency couplers while [11] uses tuneable couplers. What kind of quantum gate that is optimized can also be different.

1.4 Quantum Gates

The two-qubit gate which this study is focused on is the Controlled-Z (CZ) gate, acting upon the computational subspace with

$$\hat{U}_T = |10\rangle\langle 10| + |01\rangle\langle 01| + |00\rangle\langle 00| - |11\rangle\langle 11|, \tag{1.3}$$

which can be expressed as a matrix,

$$U_{\mathrm{CZ}} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix}. \tag{1.4}$$

A CZ gate adds π to the phase of the state of the system if both qubits are in the excited $|1\rangle$ state, otherwise nothing changes. In other words, the gate applies a Pauli-Z matrix σ_z to the second qubit if the first qubit is in the excited $|1\rangle$ state. The CZ gate is used in quantum error correction, and the error rate of the gate limits the performance of quantum error correction [3]. This gate is run on the simulated two-qubit device.

For single-qubit gates, the focus is on the π-pulse,

$$\hat{U}_T = |1\rangle\langle 0| + |0\rangle\langle 1|, \tag{1.5}$$

which in matrix format is the Pauli-X matrix,

$$\sigma_x = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}.$$

The π-pulse gate is run on the simulated single-qubit device. A π-pulse is the quantum analogy of a Boolean NOT gate.

1.5 Choice of Units

In the entire thesis, I have used units so that the reduced Planck constant $\hbar = 1$, the elementary charge $e = \pi$, and the magnetic flux quantum $\Phi_0 = 1$. The three constants are not independent as $\Phi_0 = \frac{\pi\hbar}{e} = 1$. In some parts, $\hbar, e, \Phi_0$ are written out explicitly to help the reader. This choice of units is convenient writing-wise and makes the simulations more numerically stable. Numerically, $\hbar = 1$ is much easier to handle than $\hbar \approx 10^{-34}$. $\hbar = 1$ is the convention of all QuTiP [12] functions, a Python library I have used in the study. With $\hbar = 1$, I define the unit of the Hamiltonian of the simulated devices considered in rad/s.

The choice of units does not impact the time evolution operator $\hat{U}(t)$ as it is dimensionless. As such, the choice of units will not impact the error rate of quantum gates. What will be affected by the choice of units is the meaning of the magnitude of the controls and the time scale of the controls. Note that in SI-units the unit of $\hbar$ =Js. Hence, one can achieve $\hbar = 1$ by only tweaking the energy scale, not the time scale.

2
Theory

This chapter focuses on modeling the quantum hardware, the theory behind the two model-based quantum gate optimization approaches considered, and introduces deep reinforcement learning. No new results are derived, but more details are given than in a typical review article.

2.1 Time Evolution of Closed Quantum Systems

Performing a quantum gate in a quantum computer corresponds to evolving a quantum system in time. In this thesis, the simulated quantum computing hardware is modeled as a closed quantum system, meaning a system whose time evolution is described by unitary operators. The time evolution of closed systems can be described by the Schrödinger equation

$$i\hbar\frac{\partial}{\partial t}\left|\psi(t)\right\rangle = \hat{H}(t)\left|\psi(t)\right\rangle, \tag{2.1}$$

where the Hamiltonian $\hat{H}$ is a hermitian operator, and $|\psi(t)\rangle$ the state of the system. The solution is $|\psi(t)\rangle = \hat{U}(t)\,|\psi(0)\rangle$ where in general

$$\hat{U}(t) = \tau \exp\left(\frac{-i}{\hbar}\int_0^t \hat{H}(t)dt\right), \tag{2.2}$$

where τ is the time ordering operator and $\hat{U}(t)$ unitary. If $\hat{H}(t)$ is time independent or commutes with itself for all times, this simplifies to

$$\hat{U}(t) = \exp\left(\frac{-i}{\hbar}\int_0^t \hat{H}(t)dt\right). \tag{2.3}$$

One also gets Eq. (2.3) by approximating $\hat{U}(t)$ with a first-order Magnus expansion, a continuous version of the Baker-Campbell-Hausdorff (BCH) formula. The second-order Magnus expansion is per [13]

$$\hat{U}(t) = \exp\left(\frac{-i}{\hbar}\int_0^t \hat{H}(t)dt - \frac{1}{2(i\hbar)^2}\int_0^t dt_2 \int dt_1 \left[\hat{H}(t_1), \hat{H}(t_2)\right]\right). \tag{2.4}$$

$\hat{H}$ commuting with itself across all times implies $[\hat{H}(t_1), \hat{H}(t_2)] = 0$ for every t_1, t_2.

It is difficult to work with operators like $\hat{U}(t)$ numerically. What is instead done is to use a basis to transform an operator into a matrix. Let H be the matrix corresponding to $\hat{H}$, then $H_{ij} = \langle\psi_i|\,\hat{H}\,|\psi_j\rangle$ where $\{\psi_i\}_{i=1}^N$ is the basis set for the

computation. Any operator in a (sub)space with the basis $\{\phi_i\}_{i=1}^N$ can be described by

$$\hat{U} = \sum_{i,j=1}^{N} U_{ij} \left|\phi_i\right\rangle \left\langle\phi_j\right|, \tag{2.5}$$

where $U_{ij} = \left\langle\phi_i\right| \hat{U} \left|\phi_j\right\rangle$ is the matrix representation of $\hat{U}$. The derivation is

$$\hat{U} = \left(\sum_{i=1}^{N} \left|\phi_i\right\rangle \left\langle\phi_i\right|\right) \hat{U} \left(\sum_{j=1}^{N} \left|\phi_j\right\rangle \left\langle\phi_j\right|\right) = \sum_{i,j=1}^{N} U_{ij} \left|\phi_i\right\rangle \left\langle\phi_j\right|, \tag{2.6}$$

where it is used that $\sum_{i=1}^{N} \left|\phi_i\right\rangle \left\langle\phi_i\right| = \mathrm{I}$. From here on, out of notational convenience, operators $\hat{O}$ are sometimes written as O.

2.2 Unitary Transformations

When analyzing quantum systems, working in a nice coordinate system is often beneficial as it can make the actual physics clearer and easier to analyze analytically. A coordinate transformation

$$\left|\psi(t)\right\rangle = \hat{A}(t) \left|\Psi(t)\right\rangle. \tag{2.7}$$

is unitary if $\hat{A}$ is unitary, $\hat{A}\hat{A}^\dagger = \mathrm{I}$. As $\hat{A}(t)$ is known changing between $\left|\psi(t)\right\rangle$ and $\left|\Psi(t)\right\rangle$ is straightforward. Let $i\hbar\frac{\partial}{\partial t}\left|\Psi(t)\right\rangle = \hat{H}(t)\left|\Psi(t)\right\rangle$. Then

$$i\hbar\frac{\partial}{\partial t}\left|\psi(t)\right\rangle = \left(i\hbar\frac{\partial}{\partial t}(\hat{A})\hat{A}^\dagger + \hat{A}H(t)\hat{A}^\dagger\right)\left|\psi(t)\right\rangle. \tag{2.8}$$

Hence, dynamics defined by $(\hat{H}, \Psi)$ becomes $(\tilde{H}, \psi)$ in the coordinate system defined by $\left|\psi(t)\right\rangle = \hat{A}(t)\left|\Psi(t)\right\rangle$, where

$$\tilde{H} = i\hbar\frac{\partial}{\partial t}\left(\hat{A}\right)\hat{A}^\dagger + \hat{A}\hat{H}(t)\hat{A}^\dagger. \tag{2.9}$$

The trick is to pick a useful $\hat{A}(t)$. An example is the interaction picture, defined by $\hat{A}(t) = \exp(i\hat{H}_0 t)$. Let $\hat{H}(t) = \hat{H}_0 + \hat{H}_d(t)$ and $\hat{A}(t) = \exp(i\hat{H}_0 t)$. $\hat{A}$ implies a coordinate system rotating like the intrinsic dynamics of the system given by H_0. Then per Eq. (2.9) and the BCH lemma, for example derived in [14],

$$e^X Y e^{-X} = Y + [X,Y] + \frac{1}{2}[X,[X,Y]] + \ldots \frac{1}{n!}[\overbrace{X,[X,\ldots[X}^{nY's},Y]\ldots] + \ldots. \tag{2.10}$$

one obtains

$$\tilde{H} = \hat{A}(t)\hat{H}_d\hat{A}^\dagger(t). \tag{2.11}$$

$X = Y = \hat{H}_0$ leads to the cancellation of the $\hat{H}_0$ term by the rotating frame $\hat{A}(t) = \exp(i\hat{H}_0 t)$.

2.3 Quantum Optimal Control

As defined by [15] Quantum Optimal Control (QOC) is a branch of optimal control theory aiming to adapt optimal control to quantum mechanical systems. Optimal control is a mathematical theory aiming to design control signals $\vec{C}(t)$ that manipulate dynamical systems to achieve specific goals. Often, the goal is to optimize some figure of merit. In general, this is a constrained infinite-dimensional optimization problem. The optimization parameter is the control signal. A dynamical system influenced by a control signal $\vec{C}(t)$ can be defined by

$$\frac{\partial \vec{x}}{\partial t} = \vec{f}(\vec{x}, \vec{C}(t), t), \tag{2.12}$$

where $\vec{f}$ is a function describing the time evolution of a quantity $\vec{x}$. The dynamical system this study focuses on is the Schrödinger equation, Eq. (2.1).

The Hamiltonian of a quantum system can be divided into two parts, the intrinsic dynamics of the system given by $\hat{H}_{\text{sys}}$ and the dynamics induced by the control $\hat{H}_{\text{ctrl}}$. $\hat{H}_{\text{ctrl}}$ depends on the control signal $\vec{C}(t)$. It is often assumed that $\hat{H}_{\text{sys}}$ is time-independent, but this is an approximation of experimental hardware. In total, ideally, the system would be described by

$$\hat{H}(t)[\vec{C}(t)] = \hat{H}_{\text{sys}} + \hat{H}_{\text{ctrl}}(t)[\vec{C}(t)]. \tag{2.13}$$

This study focuses on quantum gate optimization. This means that the figure of merit to optimize is $F(\hat{U}(t_g), \hat{U}_{\text{T}})$ where $\hat{U}(t_g)$ is the time evolution operator at the end of the gate, $\hat{U}_T$ defines the gate, F is a measure of the similarity of $\hat{U}_T$ and $\hat{U}(t_g)$, and t_g is the gate time. This is not the same task as state preparation, which would be defined by $F(|\psi_T\rangle, \hat{U}(t_g)|\psi_0\rangle)$ where $|\psi_T\rangle$ is the target state and $|\psi_0\rangle$ the initial state.

Quantum gate optimization is challenging; a real system might have time-dependent intrinsic dynamics, unknown terms in $\hat{H}_{\text{sys}}$, and unknown distortions of the control signals. Similarly to [9], experimental hardware could be described by

$$\hat{H}_{\text{real}} = \hat{H}_{\text{sys}} + \tilde{H}_{\text{sys}}(t) + \hat{H}_{\text{ctrl}}(t)[\vec{D}(\vec{C}(t), t)] \tag{2.14}$$

where $\tilde{H}_{\text{sys}}(t)$ is the unknown, potentially time-dependent part of the intrinsic dynamics, and $\vec{D}(\vec{C}(t), t)$ describes the distorted control signals. Handling $\vec{D}(\vec{C}(t), t)$, $\tilde{H}_{\text{sys}}(t)$ is a challenge and is a motivating factor behind using model-free approaches to QOC. I mainly simulate $\hat{H} = \hat{H}_{\text{sys}} + \hat{H}_{\text{ctrl}}(t)[\vec{\Omega}(t)]$ but notably consider a distortion $\vec{D}(\vec{C}(t))$ modeling the effect of bandwidth limited control electronics. Simulating $\tilde{H}_{\text{sys}}$ is a topic for future studies.

To derive quantum gates based on modeling $\hat{H}_{\text{sys}}$ and $\hat{H}_{\text{ctrl}}(t)[\vec{C}(t)]$ one can measure the parameters of $\hat{H}_{\text{sys}}$ and $\hat{H}_{\text{ctrl}}(t)[\vec{C}(t)]$ and handle $\tilde{H}_{\text{sys}}(t)$ and $\hat{H}_{\text{ctrl}}(t)[\vec{D}(\vec{\Omega}(t), t)]$ by making the system robust to perturbations. A model-free approach can be optimized directly using $\hat{H}_{\text{real}}$. The risk is that good model-free gates are only obtained after testing many different control shapes on $\hat{H}_{\text{real}}$ many times, while a model-based approach, in principle, just requires measuring the system parameters. This can be

addressed by hybridizing model-free and model-based approaches by initializing the model-free approach using a model-based ansatz. This is a focus of this thesis.

The challenge in QOC is finding gates that are short, have a low error, and are robust. It is a challenge as fast gates tend to have a higher error rate than slow gates, while robustness sometimes comes at the cost of the error rate.

2.4 Gate Error Measure

Here, I motivate the choice of gate error measure. The goal in gate optimization is to optimize the control pulses to ensure that the time evolution of the system at the end of the gate $U(t_g)$ is as similar as possible to the target unitary of the gate U_T. The similarity of U_T and $U(t_g)$ can be measured with the expected fidelity F_{avg} of applying the gate to an initial state ψ_0 sampled from a distribution of initial states ρ_θ,

$$F_{\text{avg}} = \mathrm{E}_{\psi_0 \sim \rho_\theta}\left(\left|\langle\psi_0| U_T^\dagger U(t_g) |\psi_0\rangle\right|^2\right) \approx \frac{1}{N}\sum_{i=1}^{N}\left|\langle\psi_i| U_T^\dagger U(t_g) |\psi_i\rangle\right|^2 \tag{2.15}$$

where the approximation is more correct for large N, ψ_i is a sampled ψ_0. The measure is reasonable as the global phase difference between $U_T, U(t_g)$ is of little interest. Other choices exist. The problem with the general definition of expected fidelity, Eq. (2.15), is that it is expensive, as a large N is required. A remedy is to pick a distribution of initial states ρ_θ such that the expectation value can be calculated analytically. [16] shows that for a uniform distribution across all possible normalized initial states, any linear operator M fulfills

$$\mathrm{E}_{\psi_0 \sim \mathrm{Unif}}(|\langle\psi_0| M |\psi_0\rangle|^2) = \frac{1}{d(d+1)}\left(\mathrm{Tr}(MM^\dagger) + |\mathrm{Tr}(M)|^2\right) \tag{2.16}$$

where M is of dimension $d \times d$. Noteably $\mathrm{Tr}(MM^\dagger) = \sum_{i,j}|M_{ij}|^2 \in \mathrm{R}$, making the full expression real for all M. Inserting the unitary $M = U_T^\dagger U(t_g)$ one finds

$$F_{\text{avg}}(U_T, U) = \frac{1}{d(d+1)}\left(d + \left|\mathrm{Tr}\left(U_T^\dagger U(t_g)\right)\right|^2\right). \tag{2.17}$$

This would mean optimizing F_{avg} is equivalent to optimizing $|\mathrm{Tr}(U_T^\dagger U(t_g))|^2$.

Unfortunately, the fidelity measure defined by Eq. (2.17) is problematic. For gate synthesis applications, one does not care about what happens to states outside the computational subspace, except that they should not evolve into the computational subspace. This means many different U_T are equally desirable from this point of view, with the full general description given by

$$\hat{U}_T = \sum_{i,j\in\text{comp}}^{N} A_{ij} |\phi_i\rangle\langle\phi_j| + \sum_{i,j\notin\text{comp}}^{N} B_{ij} |\phi_i\rangle\langle\phi_j|, \tag{2.18}$$

where B_{ij} are arbitrary (except the unitarity criteria, which leaves a lot of degrees of freedom) as far as the gate design problem is concerned, and comp defines the computational subspace.

Another problem with Eq. (2.17) is that the initial state is likelier to be inside the computational subspace than outside the computational subspace. Eq. (2.17) is based on a uniform distribution of all possible initial states, not a uniform distribution of all initial states in the computational subspace. The result according to [16] is that Eq. (2.17) is an underestimate of the actual fidelity.

A fidelity measure F focused on the computational subspace addressing the undefined U_T problem and considering that ψ_0 is likely in the computational subspace is used in [10], suggested and derived by [16]. The difference relative to Eq. (2.17) is to sample uniformly normalized initial states solely in the computational subspace and let $M = \hat{\Pi}\hat{U}_T^\dagger\hat{\Pi}\hat{U}(t_g)\hat{\Pi}$. $\hat{\Pi}$ is the projection operator for the computational subspace,

$$\hat{\Pi} = \sum_{i \in \text{comp}} |\phi_i\rangle \langle\phi_i| , \tag{2.19}$$

where comp $= \{0, 1\}$ for a single-qubit system. Note that

$$\hat{\Pi}\hat{U}_T = \sum_{i,j \in \text{comp}}^{N} A_{ij} |\phi_i\rangle \langle\phi_j| , \tag{2.20}$$

implying that B_{ij} can be ignored if using the fidelity measure F focused on the computational subspace as a figure of merit instead of Eq. (2.17). Note that $[\hat{\Pi}, \hat{U}_T] = [\hat{\Pi}, \hat{U_T}^\dagger] = 0$. For notational convenience, redefine $\Pi U_T \to U_T, U_T^\dagger\Pi \to U_T^\dagger$, with this notation $U_T^\dagger U_T = \Pi \neq \mathrm{I}$. The end result when rewriting Eq. (2.16) using for example $\hat{\Pi}\hat{\Pi}\hat{\Pi} = \hat{\Pi}, \mathrm{Tr}(MM^\dagger) = \mathrm{Tr}(M^\dagger M)$, is

$$F\left(\hat{U}_T, \hat{U}(t_g)\right) = \frac{1}{d_c(d_c+1)}\left(\mathrm{Tr}\left(\hat{\Pi}\hat{U}^\dagger(t_g)\hat{\Pi}\hat{U}(t_g)\hat{\Pi}\right) + \left|\mathrm{Tr}\left(\hat{U}_T^\dagger\hat{\Pi}\hat{U}(t_g)\hat{\Pi}\right)\right|^2\right) \tag{2.21}$$

where $\hat{\Pi}$ is the projection operator of the computational subspace (comp $= \{0, 1\}$ if qubit), d_c the dimension of the computational subspace. $d_c = 2$ for the single-qubit case; $d_c = 4$ for the two-qubit case. Observe that M is not, in general, unitary. This means $\mathrm{Tr}(MM^\dagger) \neq d$. F given by Eq. 2.21 is the fidelity measure F used in this study, both for the single-qubit and two-qubit cases. Often, the focus is on minimizing the infidelity $I = 1 - F$ instead of maximizing the fidelity F.

2.5 Decoherence and Leakage

As a quantum computer must be controllable, it will interact with the environment. The interaction with the environment leads to a loss of coherence. This is a reason why faster gates are preferable. Properly modeling interactions with the environment is computationally expensive, requiring, for example, the Lindblad master equation. Therefore, this study uses simplified models of decoherence. A model of the decoherence infidelity due to interactions with the environment is given by [17],

$$I_c = \frac{d}{2(d+1)} t_g \sum_{k=1}^{N}\left(1/T_1^k + 1/T_2^k\right) \tag{2.22}$$

where $d = 2^N$ is the number of computational states, N the number of computational qubits, t_g is the gate time, T_1^k the relaxation time and T_2^k the dephasing time of qubit k. The total gate infidelity, considering decoherence,

$$I_{\text{total}} = 1 - F + I_c. \tag{2.23}$$

Unless explicitly stated, the infidelity in the report is $I = 1 - F$, not the total infidelity.

Eq. (2.22) captures the decoherence if the system mainly stays in the computational subspace. For this study, this means Eq. (2.22) is applicable to the π pulse (where $N = 1$) but not the CZ gate as it is often implemented by (on-purpose) leaving the computational subspace. A CZ gate can be realized in several different ways. For the CZ gate realization focused on in this thesis, the formula from [18] is more relevant,

$$I_c = \left(\frac{1}{2T_1^1} + \frac{3}{10T_1^2} + \frac{31}{40T_2^1} + \frac{3}{8T_2^2} \right) t_g \tag{2.24}$$

where T_1^k, T_2^k are the coherence times (relaxation and dephasing) of qubit k. Notably, Eq. (2.24) is not symmetric, which makes sense as the CZ gate realization treats the two qubits differently. The CZ gate realization is explained in detail in section 3.3.2. In an attempt to mimic the Chalmers device, both the single-qubit and two-qubit systems were modeled with $T_1 \approx 80$ µs, $T_2 \approx 40$ µs. See Table B.1 and Table B.2 for the system parameters used.

Unfortunately, shorter pulses often imply high-amplitude control signals. This will also be bad for the gate infidelity, as intense pulses will have a higher bandwidth, meaning there is a risk that more leakage occurs. Leakage means that the quantum system leaves the computational subspace. This can happen if the bandwidth of the signal is so wide that more transitions than intended are activated. An example of leakage would be if the probability of finding a qubit in the second excited state $|2\rangle$ at the end of the gate is non-zero, $|\langle 2|\psi(t_g)\rangle|^2 > 0$.

2.6 Robustness

The true dynamics of the system studied will not be perfectly known and might change over time as the system interacts with the environment. The uncertainty is both in the parameters of the system (for example, the resonance frequency of the qubit) and, if using a theoretical model, the accuracy of the model. As such, frequent recalibrations are commonly required. In [8], recalibration for one hour is interleaved with one hour of benchmarking. This uncertainty and parameter drift make robustness, good performance despite perturbations, a beneficial property.

The problem is that robustness might lead to decreased performance. [19] shows that stochastic gradient descent leads to more robustness but worse performance compared with gradient descent when applied in a quantum optimal control task, while [20] shows that a model-free approach beats a model-based approach when the deviation between the model and the simulated set-up becomes large. This is to be expected; a pulse optimized to work well on a specific Hamiltonian might be very sensitive to small deviations in the Hamiltonian. One could hence expect a trade-off between robustness and performance.

2.7 Modeling

This study focuses on two models: a single transmon and two transmons connected via a tunable coupler. Here, the analytical models used for the simulations are introduced and motivated. This section closely follows what already exists in the literature. The section is primarily based on [11, 21, 22]. The section explains why I let the single-qubit simulation be defined by

$$\hat{H}_{\text{1-qb}} = \hat{H}_{\text{sys}} + \hat{H}_{\text{ctrl}} = \delta(t)\hat{a}^\dagger\hat{a} + \frac{\alpha}{2}\hat{a}^\dagger\hat{a}^\dagger\hat{a}\hat{a} + \frac{\Omega_x(t)}{2}\left(\hat{a} + \hat{a}^\dagger\right) + i\frac{\Omega_y(t)}{2}\left(\hat{a}^\dagger - \hat{a}\right) \quad (2.25)$$

where $\delta(t) = \omega_q - \omega_d$, ω_q is the qubit frequency, ω_d is the drive frequency, α is the anharmonicity, $\hat{a}$ is the annihilation operator, $\hat{a}^\dagger$ is the creation operator, and $\Omega_x(t), \Omega_y(t)$ is components of the control signal $\xi(t) = \Omega_y(t)\cos(\omega_d t) + \Omega_x(t)\sin(\omega_d t)$. The section also explains why the two-qubit simulation was made with

$$\hat{H}_{\text{2-qb}} = \hat{H}_c + \sum_{i=\{1,2\}} \hat{H}_i - g_{ic}(\hat{a}_i - \hat{a}_i^\dagger)(\hat{a}_c - \hat{a}_c^\dagger) \quad (2.26)$$

where g_{ic} is the coupling strength between qubit i and the coupler and $\hat{H}_1, \hat{H}_2, \hat{H}_c$ are

$$\hat{H}_k = \omega_k \hat{a}_k^\dagger \hat{a}_k + \frac{\alpha_k}{2}\hat{a}_k^\dagger\hat{a}_k^\dagger\hat{a}_k\hat{a}_k, \quad (2.27)$$

where ω_k is the frequency of qubit $(k = 1, 2)$ or coupler $k = c$. The two-qubit system is controlled via $\omega_c(t) = \omega_c^0\sqrt{|\cos(\pi\Phi(t))|}$ where $\Phi(t)$ is the flux signal sent to the system and ω_c^0 is ω_c when $\Phi(t) = 0$. The two-qubit simulation uses the parameters in Table B.1, while the single-qubit simulation uses the parameters in Table B.2. The simulation parameters are based on the experimental set-up at Chalmers.

2.7.1 Intrinsic Dynamics of a Transmon

This section summarizes how the model of the intrinsic dynamics of a single transmon $\hat{H}_{\text{sys}}$ is derived. The equivalent circuit diagram of an isolated transmon is given by Fig. 2.1. This is the basic building block of the recent 105-qubit flip-chip Chinese processor [23] as well as Chalmers' quantum computing efforts. The transmon is an appealing choice for superconducting circuit qubits as the design is insensitive to charge noise.

Using circuit quantization, explained in more details in [21], one obtains that the transmon shown in Fig. 2.1 is described by

$$\hat{H}_{\text{sys}} = 4E_C\hat{n}^2 - E_J\cos(\hat{\phi}) \quad (2.28)$$

where $\hat{n} = \frac{\hat{Q}}{2e}$ is the reduced charge, $\hat{Q}$ is the charge of the capacitor of the transmon, $\hat{\phi} = \frac{2\pi\hat{\Phi}_T}{\Phi_0}$ is the reduced flux, $\hat{\Phi}_T$ is the flux of the transmon, E_J is the Josephson energy (which can be influenced by designing the critical current of the junction), and the charging energy $E_C = \frac{e^2}{2C_\Sigma}$, where $C_\Sigma = C_J + C_S$ is the total capacitance of the transmon. The transmon's insensitivity to charge noise comes from designing the device such that $E_J \gg E_C$, as shown by [22]. This can be achieved by making

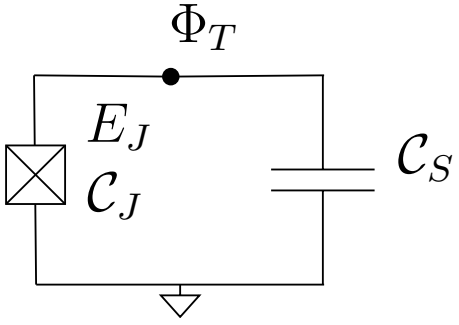


Figure 2.1: Equivalent circuit diagram of a transmon. The crossed box is the Josephson junction. A Josephson junction consists of two superconductors separated by a thin barrier. The Josephson junction is characterized by the Josephson energy E_J and a capacitance C_J. C_S is a shunt capacitance and Φ_T the flux of the transmon.

C_S large. Wanting to express the system in the standard harmonic oscillator basis $\{n\}_{i=0}^{\infty}$, one defines the annihilation and creation operators $\hat{a}, \hat{a}^\dagger$ such that

$$\hat{n} = \left(\frac{E_J}{32E_C}\right)^{1/4} i\left(\hat{a} - \hat{a}^\dagger\right) \tag{2.29}$$

and

$$\hat{\phi} = \left(\frac{2E_C}{E_J}\right)^{1/4} \left(\hat{a} + \hat{a}^\dagger\right). \tag{2.30}$$

With this definition one obtains the familar annihilation and creation operator properties $\hat{a}\,|n\rangle = \sqrt{n}\,|n-1\rangle$, $\hat{a}^\dagger\,|n\rangle = \sqrt{n}\,|n+1\rangle$, $[\hat{a}, \hat{a}^\dagger] = 1$. This is a convention; other ways to define $\hat{a}, \hat{a}^\dagger$ exist.

Eq. (2.28) can be simplified via a Taylor expansion and perturbation theory. Expand the cosine non-linearity to fourth order in $\hat{\phi}$ ($\hat{\phi}$ is small since $E_J \gg E_C$),

$$E_J \cos\left(\hat{\phi}\right) \approx E_J - \frac{E_J}{2}\hat{\phi}^2 + \frac{E_J}{24}\hat{\phi}^4 \tag{2.31}$$

and disregard all constant terms (for example, the E_J term) as they only impact the global phase. Observe that $\hat{\phi}^4 \propto (\hat{a} + \hat{a}^\dagger)^4$. [22] suggests keeping only terms $\hat{B}$ in $(\hat{a} + \hat{a}^\dagger)^4$ such that $|n\rangle$ is an eigenstate ($\hat{B}\,|n\rangle = B_n\,|n\rangle$). [22] motivates this with perturbation theory. Consider $H = H_0 + V$ with $\hat{V} = -\frac{E_J}{24}\hat{\phi}^4$ being the perturbation. The first-order correction of the energy $E_n^{(1)} = \langle n|\,V\,|n\rangle$. This means that to first-order in energy, only terms in V where $|n\rangle$ is an eigenstate matter. If one keeps only such terms and disregards any constant terms, one can show that $(\hat{a} + \hat{a}^\dagger)^4 \to 6\hat{a}^\dagger\hat{a}^\dagger\hat{a}\hat{a} + 12a^\dagger\hat{a}$. This results in, as claimed in [21],

$$\hat{H}_{\text{sys}} = \omega_q\hat{a}^\dagger\hat{a} + \frac{\alpha}{2}\hat{a}^\dagger\hat{a}^\dagger\hat{a}\hat{a}. \tag{2.32}$$

where $\alpha = -E_C = \omega_{12} - \omega_q < 0$ ($\omega_{ij} = E_j - E_i$ where $\hat{H}_{\text{sys}}\,|i\rangle = E_i\,|i\rangle$) is the anharmonicity of the lowest energy level outside the qubit subspace and the qubit frequency

$$\omega_q = \omega_{01} = \sqrt{8E_CE_J} - E_C \tag{2.33}$$

Note that $E_J \gg E_C$ leads to the transmon's insensitivity to charge noise but is problematic as $E_J \gg E_C$ is usually obtained by making $E_C = |\alpha|$ small, while a large anharmonicity $|\alpha|$ leads to less leakage. There is, hence, a trade-off between sensitivity to charge noise and low leakage. Fortunately, the charge insensitivity is exponentially suppressed with increasing E_J/E_C [22], while the anharmonicity $|\alpha|$ only decreases linearly with E_C.

Finally rewrite $\hat{H}_{\text{sys}}$ in the rotating frame

$$\hat{A} = e^{i(\omega_d t+\phi)\hat{a}^\dagger\hat{a}}, \tag{2.34}$$

with the detuning $\delta = \omega_q - \omega_d$

$$\hat{H}_{\text{sys}} = \delta\hat{a}^\dagger\hat{a} + \frac{\alpha}{2}\hat{a}^\dagger\hat{a}^\dagger\hat{a}\hat{a}. \tag{2.35}$$

The rewrite uses the BCH lemma Eq. (2.10) and $[\hat{a}^\dagger\hat{a}, \hat{a}^\dagger\hat{a}^\dagger\hat{a}\hat{a}] = 0$ (shown in appendix A.2).

2.7.2 Control of a Transmon

So far, the focus has been on $\hat{H}_{\text{sys}}$ of the Transmon; here, modeling $\hat{H}_{\text{ctrl}}$ is discussed. The transmon is controlled via a weak capacitative coupling with a microwave drive line. The set-up is described by Fig. 2.2.

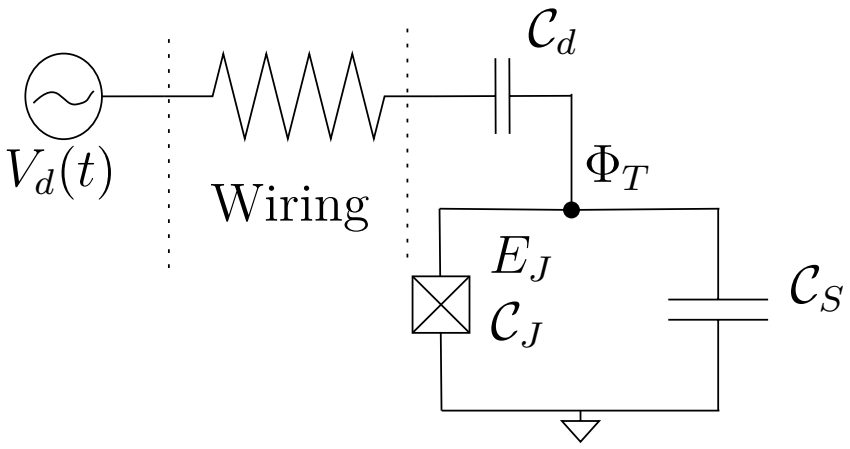


Figure 2.2: The full single-qubit set-up. The transmon is controlled via a microwave drive line capacitively coupled to the transmon. The wiring dampens the voltage signal $V_d(t)$. The chip's temperature is typically less than 60 mK while the signal generator is at room temperature. Φ_T is the flux of the transmon at the node.

The aim of this section is to find $\hat{H}_{\text{ctrl}}$ of the set-up in Fig. 2.2 in the same frame $\hat{A}$ as $\hat{H}_{\text{sys}}$. It is explained how one can use the rotating-wave approximation and a rotating frame to derive that a transmon driven by a weak capacitive coupling is described by

$$\hat{H}_{\text{ctrl}} = \frac{\Omega_x(t)}{2}\left(\hat{a} + \hat{a}^\dagger\right) + i\frac{\Omega_y(t)}{2}\left(\hat{a}^\dagger - \hat{a}\right) \tag{2.36}$$

where the voltage signal used to drive the transmon

$$V_d(t) \propto \xi(t) = \Omega_y(t)\cos(\omega_d t + \phi) + \Omega_x(t)\sin(\omega_d t + \phi). \tag{2.37}$$

The aim of this thesis is to optimize the controls Ω_x, Ω_y, and ω_d. The focus is on $\xi(t)$, not $V_d(t)$, as the proportionality constant must be measured experimentally.

Assuming weak coupling $C_d \ll C_\Sigma$, the system shown in Fig. 2.2 can (according to [21]) be described by

$$\hat{H}_{\text{ctrl}} = \frac{\xi(t)\hat{Q}}{Q_{ZPF}} \tag{2.38}$$

where $\hat{Q}/Q_{ZPF} = -i(\hat{a} - \hat{a}^\dagger)$, Q_{ZPF} is a constant (with dimension charge) describing the zero-point fluctuations of the system. Algebraically, when defining $\Omega = \Omega_y + i\Omega_x$, one finds that

$$\hat{H}_{\text{ctrl}} = -i\left(\frac{\Omega(t)}{2}e^{-i(\omega_d t+\phi)} + \frac{\Omega^*(t)}{2}e^{i(\omega_d t+\phi)}\right)\left(\hat{a} - \hat{a}^\dagger\right). \tag{2.39}$$

Using the BCH lemma Eq. 2.10, $[\hat{a}^\dagger\hat{a}, \hat{a}] = -\hat{a}$, $[\hat{a}^\dagger\hat{a}, \hat{a}^\dagger] = \hat{a}^\dagger$, one can derive that the $\hat{H}_{\text{sys}}$ rotating frame

$$\hat{A} = e^{i(\omega_d t+\phi)\hat{a}^\dagger\hat{a}}, \tag{2.40}$$

transforms $\hat{a} \to \hat{a}e^{-i(\omega_d t+\phi)}$, $\hat{a}^\dagger \to \hat{a}^\dagger e^{i(\omega_d t+\phi)}$. This results in

$$\hat{H}_{\text{ctrl}}/(-i) = \frac{\Omega(t)}{2}e^{-i2(\omega_d t+\phi)}\hat{a} + \frac{\Omega^*(t)}{2}\hat{a} - \frac{\Omega^*(t)}{2}e^{i2(\omega_d t+\phi)}\hat{a}^\dagger - \frac{\Omega(t)}{2}\hat{a}^\dagger \tag{2.41}$$

The next step is to apply the rotating-wave approximation. The rotating-wave approximation is an approximation that, if valid, enables discarding fast-rotating terms. The basic intuition behind the rotating-wave approximation is that terms that rotate faster than any other rate or frequency can be time-averaged to zero. In this case, the rotating-wave approximation implies disregarding the $e^{\pm i2(\omega_d t+\phi)}$ terms. The result is

$$\hat{H}_{\text{ctrl}} = \frac{\Omega_x(t)}{2}\left(\hat{a} + \hat{a}^\dagger\right) + i\frac{\Omega_y(t)}{2}\left(\hat{a}^\dagger - \hat{a}\right). \tag{2.42}$$

Rigorously motivating the rotating-wave approximation is not trivial, even though it is used without detailed justification by other studies of single-qubit gates, for example, [21, 24]. There are parameter regimes where the rotating-wave approximation is invalid for single-qubit gates [25]. The key, according to [25], is to ensure that the frequency of the fast rotations is much larger than the drive strength. In the single-qubit case, the drive strength corresponds to Ω_x and Ω_y. The rotating-wave approximation is hence justified if $2\omega_d \gg \Omega_x(t), \Omega_y(t)$. For the single-qubit system considered in this thesis, $2\omega_d \gg \Omega_x, \Omega_y$ seems reasonable as ω_d is typically a few GHz while Ω_x and Ω_y are typically tens of MHz. The rotating-wave approximation will be less accurate for small gate times t_g. Throughout this study, the rotating-wave approximation based $\hat{H}_{\text{ctrl}}$ Eq. 2.42 is used, as is commonly done in the literature.

The control signal $\xi(t)$ is according to [21], assuming weak coupling, given by

$$\xi(t) = \frac{C_d V_d(t) Q_{ZPF}}{C_\Sigma D} \tag{2.43}$$

where D is any extra damping of the control signal, C_d the capacitance between the transmission line and the qubit, $C_\Sigma = C_J + C_s$ the total capacitance of the transmon and Q_{ZPF} the zero-point flucations of the system. Weak coupling means $C_d \ll C_\Sigma$. What voltage $V_d(t)$ (Ω_x, Ω_y) corresponds to depends the experimental constants Q_{ZPF}, C_D, C_Σ and D. The magnitudes of Ω_x and Ω_y in this study are expressed in MHz as the unit choice $\hbar = 1$ implies all Hamiltonians $\hat{H}$ are expressed in rad/s.

2.7.3 Two-qubit Set-up

The experimental set-up at Chalmers consists of qubits connected via tunable couplers. The simplest such system available is described in detail in [11], and the circuit diagram is shown in Fig. 2.3. This is the set-up used when simulating two-qubit gates.

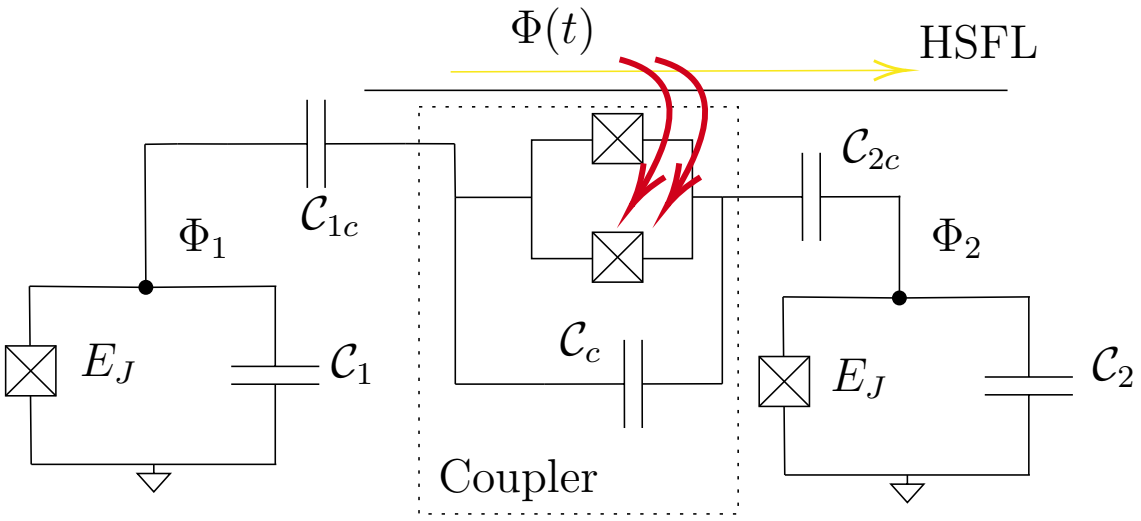


Figure 2.3: The two-qubit set-up considered in the thesis. Two qubits are controlled via a tunable coupler. The frequency of the coupler $\omega_c(t)$ is controlled by sending flux $\Phi(t)$ (marked in red) through the loop with two Josephson junctions of the coupler. The loop with two Josephson junctions is known as a Superconducting Quantum Interference Device (SQUID). The flux is generated by sending a current (marked in yellow) through a high-speed flux line (HSFL). C_1 and C_2 are the total capacitances of the two transmons.

There are two fixed-frequency transmons connected via a transmon with a tunable frequency. The purpose of the tunable qubit is to control the interaction between the fixed frequency qubits; its state is not used for computations in itself. The three transmons are capacitively coupled.

According to [21] a system consisting of two qubits H_i and a coupler H_c is described by

$$H = \hat{H}_1 + \hat{H}_2 + \hat{H}_c + \hat{H}_{12} + \hat{H}_{1c} + \hat{H}_{2c}. \tag{2.44}$$

There is not supposed to be any direct coupling between the two qubits per the circuit diagram in Fig. 2.3, as such $\hat{H}_{12} = 0$. The interaction Hamiltonian can per

[21] be modeled as

$$\hat{H}_{ic} = -g_{ic}(\hat{a}_i - \hat{a}_i^\dagger)(\hat{a}_c - \hat{a}_c^\dagger). \tag{2.45}$$

Assuming $C_{ic} \ll C_i, C_c$

$$g_{ic} = 4e^2 \frac{C_{ic}}{C_i C_g} n_i n_c, \tag{2.46}$$

where $\hat{n}_j = i n_j (\hat{a}_j - \hat{a}_j^\dagger)$ and $j = i, c$. Note that there are different conventions; [11] uses

$$\hat{H}_{ic} = g_{ic}(\hat{a}_i^\dagger + \hat{a}_i)(\hat{a}_c^\dagger + \hat{a}_c). \tag{2.47}$$

The coupler can be described as a transmon with a tunable frequency. The equivalent circuit is shown in Fig. 2.3. The frequency becomes tunable via the use of a Superconducting Quantum Interference Device (SQUID). A SQUID consists of two Josephson junctions connected in parallel. The frequency is tuned by sending an external flux through the SQUID. The dynamics is according to [21] given by

$$\hat{H}_{\text{Coupler}} = 4E_C \hat{n}^2 - 2E_J \left| \cos\left(\frac{\pi \Phi(t)}{\Phi_0} \right) \right| \cos\left(\hat{\phi}\right) \tag{2.48}$$

where $\Phi(t)$ is the applied external flux, and the magnetic flux quantum $\Phi_0 = 1$ in the chosen units. The external flux is controlled by a current sent through a flux line. This is the same Hamiltonian $\hat{H}$ as the transmon, except replacing E_J with $\tilde{E}_J = 2E_J \left| \cos(\frac{\pi \Phi(t)}{\Phi_0}) \right|$. Replace $E_J \to \tilde{E}_J$ in Eq. (2.33) and use that $E_J \gg E_C$ to obtain

$$\omega_c(t) = \sqrt{8\tilde{E}_J E_C} - E_C \approx \sqrt{8\tilde{E}_J E_C}. \tag{2.49}$$

The end result is, as first derived in [22],

$$\omega_c(t) = \omega_c^0 \sqrt{\left| \cos\left(\frac{\pi \Phi(t)}{\Phi_0} \right) \right|}, \tag{2.50}$$

where $\omega_c(t) = \omega_c^0$ if $\Phi(t) = 0$. The approximation made in Eq. (2.49) is not valid if $\Phi(t) = (\frac{1}{2} + n)\Phi_0$ for any $n \in \mathrm{Z}$, in that case $\omega_c(t)$ should be offset by a constant.

A strong reason to avoid $\Phi(t) = (\frac{1}{2} + n)\Phi_0$ is sensitivity to flux noise. For $-0.5 < \Phi(t) < 0.5$

$$\frac{\partial \omega_c}{\partial \Phi} = \frac{-\pi \omega_c^0 \sin(\pi \Phi(t))}{2\sqrt{\cos(\pi \Phi(t))}}. \tag{2.51}$$

This means, for example

$$\lim_{\Phi \to \pm 0.5} \left| \frac{\partial \omega_c}{\partial \Phi} \right| = \infty. \tag{2.52}$$

Hence, $\Phi(t)$ close to ± 0.5 will imply sensitivity to noise in the flux signal. It is therefore preferred if $-0.5 \ll \Phi(t) \ll 0.5 \ \forall t$.

The total two-qubit Hamiltonian

$$\hat{H}_{\text{2-qb}} = \hat{H}_c + \sum_{i=\{1,2\}} \hat{H}_i - g_{ic}(\hat{a}_i - \hat{a}_i^\dagger)(\hat{a}_c - \hat{a}_c^\dagger) \tag{2.53}$$

where $\hat{H}_1, \hat{H}_2$ and $\hat{H}_c$ are transmons modeled, like previously, with

$$\hat{H}_i = \omega_i \hat{a}_i^\dagger \hat{a}_i + \frac{\alpha_i}{2} \hat{a}_i^\dagger \hat{a}_i^\dagger \hat{a}_i \hat{a}_i, \tag{2.54}$$

and the system is controlled by $\omega_c(t)$ via $\hat{H}_{\text{ctrl}} = \omega_c(t)\hat{a}_c^\dagger \hat{a}_c$.

In this thesis, I have not looked into having single-qubit drives in addition to controlling $\omega_c(t)$. This would correspond to

$$\hat{H}_{\text{ctrl}} = \omega_c(t)\hat{a}_c^\dagger \hat{a}_c + \sum_{k=1,2} \frac{\Omega_x^k(t)}{2} \left(\hat{a}_k + \hat{a}_k^\dagger\right) + i\frac{\Omega_y^k(t)}{2} \left(\hat{a}_k^\dagger - \hat{a}_k\right). \tag{2.55}$$

This could have advantages, but many electromagnetic signals at the same time might lead to significant cross-talk, meaning that different control signals interfere with each other, distorting the signals. It also leads to a higher-dimensional optimization problem, which might lead to slower convergence. This is something future studies could look into.

2.8 Optimization Methods in Quantum Optimal Control

There are many different ways to optimize quantum gates. This study applies Derivative Removal by Adiabatic Gate (DRAG), Gradient Ascent Pulse Engineering (GRAPE), deep reinforcement learning, and black-box optimized piece-wise constant controls for the single-qubit case and deep reinforcement learning combined with a black-box optimized ansatz to the two-qubit case. There are other approaches, for example, model learning investigated in [20, 26], but they are out of scope for this thesis. Model learning means finding a machine learning model to replace the Hamiltonian $\hat{H}$ for describing the dynamics of the quantum system. Here, the optimization methods are introduced. The details regarding how a two-qubit CZ gate is realized are explained in section 3.3.2 of the next chapter, outlining, for example, how the implementation relates to the level structure of the two-qubit Hamiltonian.

2.8.1 Derivative Removal by Adiabatic Gate

DRAG is a method based on analytically deriving a suitable parametrization for single-qubit gates and then optimizing that parametrization. It is specific to single-qubit gates. It is introduced by [27] and expanded upon in [24]. The fundamental idea is to optimize the controls to ideally achieve

$$U_{\text{ideal}} = U_{\text{qb}} \oplus U_{\text{rest}} \tag{2.56}$$

in some reference frame. This would mean that there is no connection between the first two energy levels (the qubit) and the rest of the levels. This means no leakage. If there is no coupling between different energy levels in $\hat{H}$, then $U(t) = U_{\text{ideal}}$. As such, the goal is to ensure

$$\hat{H} = H_{\text{qb}} \oplus H_{\text{rest}}. \tag{2.57}$$

The challenge is finding a reference frame that removes the connection between the computational subspace and the rest of the energy levels. The DRAG derivation given by [24] starts from a standard non-linear oscillator (SNO) with d levels

$$\hat{H}_{\text{SNO}} = \sum_{j=1}^{d-1} \left(j\delta(t) + \alpha_j\right) \Pi_j + \lambda_{j-1} \left(\frac{\Omega_x(t)}{2} \sigma_{j-1,j}^x + \frac{\Omega_y(t)}{2} \sigma_{j-1,j}^y \right) \tag{2.58}$$

where $\Pi_j = |j\rangle\langle j|$, $\sigma_{i,j}^x = |i\rangle\langle j| + |j\rangle\langle i|$, $\sigma_{j,k}^y = -i\,|j\rangle\langle k| + i\,|k\rangle\langle j|$, λ_i measures the strength of the transmission $|j\rangle \to |j+1\rangle$, and α_i is the anharmoncity of the $|i\rangle \to |i+1\rangle$ transition.

This becomes the single transmon model

$$\hat{H} = \delta(t)\hat{a}^\dagger\hat{a} + \frac{\alpha}{2}\hat{a}^\dagger\hat{a}^\dagger\hat{a}\hat{a} + \frac{\Omega_x(t)}{2}\left(\hat{a} + \hat{a}^\dagger\right) + i\frac{\Omega_y(t)}{2}\left(\hat{a}^\dagger - \hat{a}\right) \tag{2.59}$$

when $d \to \infty$, $\lambda_i = \sqrt{i}$ and $\alpha_n = \frac{\alpha}{2}n(n-1)$ for $n \in \mathrm{N}_0$. The single-qubit simulation is de facto a SNO as truncation is needed numerically, $d = 7$ is what is used in this study.

The approach of, for example, [24] is to find a unitary transformation $\hat{A}(t)$ so that the qubit becomes decoupled from the rest of the energy levels in that frame. To ensure that the effect of the gate is the intended in the lab frame, we require $\hat{A}(0) = \hat{A}(t_g) = \mathrm{I}$. The general problem is addressed by [24] while [27] just suggests a specific $\hat{A}(t)$. [24] suggests,

$$\begin{aligned} \Omega_x(t) &= \Omega_G(t) \\ \Omega_y(t) &= -\frac{\dot{\Omega}_G(t)}{2\alpha} \\ \delta(t) &= -\frac{\Omega_G^2(t)}{4\alpha} \end{aligned} \tag{2.60}$$

where

$$\Omega_G(t) = M\frac{\exp\left(-\frac{(t-t_g/2)^2}{2\sigma^2}\right) - \exp\left(-\frac{t_g^2}{8\sigma^2}\right)}{\sqrt{2\pi\sigma^2}\mathrm{erf}\left(\frac{t_g}{\sqrt{8}\sigma}\right) - t_g\exp\left(-\frac{t_g^2}{8\sigma^2}\right)} \tag{2.61}$$

is the standard Gaussian pulse commonly employed for single-qubit gates. $\mathrm{erf}(x) = \frac{2}{\sqrt{\pi}}\int_0^x e^{-t^2}dt$. In the case of a π-pulse, $M = \pi$. $\sigma = t_g/2$ is used in [27] and in this study. σ is a trade-off between having the signal be smooth despite truncating such that $\Omega_G(0) = \Omega_G(t_g) = 0$ and Ω_G being tight in the frequency domain. The derivation of the Gaussian pulse is given in Appendix A.1. A Gaussian pulse is preferred over, for example, a square pulse because it has a more favorable bandwidth-to-intensity relationship than less smooth pulses. This means there is less leakage using a Gaussian shape instead of, for example, a square pulse. There is ongoing research on how to improve DRAG, a recent example is [28], which suggests optimizing the shape of $\Omega_x(t)$ and using higher order derivatives. In the methodology used to derive DRAG, there is no prescription regarding the shape of $\Omega_x(t)$; there are several valid shapes.

2.8.2 Gradient Ascent Pulse Engineering

If there is an analytical model of the system's Hamiltonian $\hat{H}$, the controls are assumed to be piece-wise constant, the figure of merit is differentiable, and the system is approximated to be closed, one can optimize quantum gates using gradients calculated analytically. This approach is called GRAPE. The idea is interesting because gradient-based optimization methods are good at handling large parameter spaces. In this thesis, I have implemented a simple version of GRAPE from scratch. GRAPE is only applied to the single-qubit set-up. There are many more sophisticated approaches, see for example [29]. The goal is to derive $\frac{\partial F}{\partial c_{jk}}$ so that gradient ascent can be applied to update the controls $\vec{C}$ with gradient descent

$$c_{jk} \to c_{jk} + \epsilon \frac{\partial F}{\partial c_{jk}}. \tag{2.62}$$

ϵ is the learning rate, in order to minimize F. In the thesis, the focus is on infidelity $I = 1 - F$. The control update Eq. (2.62) can be rewritten in terms of I. The derivation of $\frac{\partial F}{\partial c_{jk}}$ consists of two-steps, finding $\frac{\partial U(t_g)}{\partial c_{jk}}$, then $\frac{\partial F}{\partial c_{jk}}$.

GRAPE assumes piece-wise constant controls to derive the analytical gradient. This means that the system must be described by

$$\hat{H}(t) = \hat{H}_{\text{sys}} + \sum_j c_{jk}\hat{H}_j, \forall t \in [t_k, t_{k+1}] \tag{2.63}$$

where $\hat{H}_j$ are the different control Hamiltonians, and the control signal is piece-wise constant, the j-th control at the k-th time step $c_j(t) = c_{jk}, \forall t \in [t_k, t_{k+1}]$. In the single-qubit case $c_1(t) = \Omega_x(t)$ and $\hat{H}_1 = (\hat{a} + \hat{a}^\dagger)/2$, for example. The end points of piece-wise constant time intervals are defined by $t_{k+1} = t_k + \Delta t, \forall k$ where Δt is a constant.

Start the GRAPE derivation by approximating $\hat{U}(t)$. In general, $[\hat{H}(t_1), \hat{H}(t_2)] \neq 0$ for piece-wise constant pulses. Do a first-order Magnus expansion approximation of $\hat{U}(t)$, to obtain

$$\hat{U}(t) = \exp\left(\frac{-i}{\hbar}\int_0^t \hat{H}(t)dt\right). \tag{2.64}$$

As H is piece-wise constant $\int_0^t \hat{H}(t)dt = \Delta t \sum_k \hat{H}(t_k)$. Then apply the Baker-Campbell-Hausdorff expansion [14], to second order

$$e^{\Delta t A + \Delta t B} \approx e^{\Delta t A} e^{\Delta t B} e^{\frac{\Delta t^2}{2}[A,B]}. \tag{2.65}$$

As $\Delta t^2 \ll \Delta t$ keep only the first order Δt terms. This line of reasoning makes it reasonable to approximate

$$\hat{U}(t_g) \approx \Pi_{k=1}^N \hat{U}_k = \hat{U}_N \hat{U}_{N-1} \dots \hat{U}_1 \tag{2.66}$$

where

$$\hat{U}_k = e^{-i\Delta t(\hat{H}_{\text{sys}} + \sum_j c_{jk}\hat{H}_j)} \tag{2.67}$$

and $t_g = N\Delta t$. This approximation is commonly applied in literature [30, 10]. The next step is approximating for small Δt

$$\frac{\partial U_k}{\partial c_{jk}} \approx -i\Delta t H_j U_k. \tag{2.68}$$

It is an approximation because $[H_{\text{sys}}, H_j] \neq 0, [H_i, H_j] \neq 0$ if $i \neq j$ in general. The validity of the approximation is motivated in [30]. The end result is that, because U_i is independent of c_{jk} if $k \neq i$,

$$\frac{\partial U(t_g)}{c_{jk}} = -i\Delta t U_N \dots U_{k+1} H_j U_k \dots U_1. \tag{2.69}$$

This can be made neater by defining the backpropagation operator $R_k = U_1^\dagger \dots U_{k-1}^\dagger U_k^\dagger$. Observe that $R_k U(t_k) = \mathrm{I}$. Insert to obtain

$$\frac{\partial U(t_g)}{\partial c_{jk}} = -i\Delta t U(t_g) R_k H_j R_k^\dagger. \tag{2.70}$$

One can define a lot of figures of merit $F(U_T, U(t_g))$ that are differentiable with respect to c_{jk}. For the fidelity measure used in this study

$$F(\hat{U}_T, \hat{U}(t_g)) = \frac{1}{d_c(d_c+1)} \left(\mathrm{Tr}\left(\hat{\Pi}\hat{U}^\dagger(t_g)\hat{\Pi}\hat{U}(t_g)\hat{\Pi}\right) + \left|\mathrm{Tr}\left(\hat{U}_T^\dagger \hat{\Pi}\hat{U}(t_g)\hat{\Pi}\right)\right|^2 \right), \tag{2.71}$$

one obtains for the infidelity $I = 1 - F$

$$\begin{aligned} \frac{\partial I}{\partial c_{jk}} = &- \frac{1}{d_c(d_c+1)} \left(\mathrm{Tr}\left(\Pi \frac{\partial U^\dagger(t_g)}{\partial c_{jk}} \Pi U(t_g) \Pi\right) + \mathrm{Tr}\left(\Pi U^\dagger(t_g) \Pi \frac{\partial U(t_g)}{\partial c_{jk}} \Pi\right) \right. \\ &\left. + 2\Re\left(\mathrm{Tr}\left(M\right)\right) \Re\left(\mathrm{Tr}\left(\frac{\partial M}{\partial c_{jk}}\right)\right) + 2\Im\left(\mathrm{Tr}\left(M\right)\right) \Im\left(\mathrm{Tr}\left(\frac{\partial M}{\partial c_{jk}}\right)\right) \right). \end{aligned} \tag{2.72}$$

$M = U_T^\dagger \Pi U(t_g) \Pi$, $\frac{\partial U(t_g)}{\partial c_{jk}}$ is given in Eq. (2.70), the computational subspace dimension $d_c = 2$ for the single-qubit case, $z = \Re(z) + i\Im(z)$.

2.8.3 Deep Reinforcement Learning

Here, deep reinforcement learning in general is introduced, and the specific deep reinforcement learning algorithm used is described and interpreted. How an ansatz can be introduced is treated in chapter 3.

Reinforcement learning is a method to control a system in an optimal way using machine learning. The control process is viewed sequentially; the reinforcement learning agent performs a sequence of actions, making the system transition through a sequence of states. Reinforcement learning requires no knowledge of the dynamics of the process. A reinforcement learning agent uses an estimate of the current state of the system s_i to select an action a_i to perform on the system. The system returns a reward r_{i+1} and an estimate s_{i+1} of the state the system transitioned into. This action-feedback loop is visualized in Fig. 2.4.

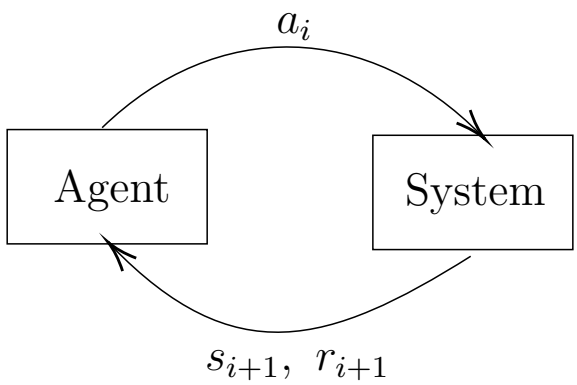


Figure 2.4: The action-feedback loop of reinforcement learning. The reinforcement learning agent selects an action a_i based on the current estimate of the system s_i and performs that action on the system. The system returns a reward r_{i+1} and an estimate of the state the system transitioned into s_{i+1}. The agent uses s_{i+1} to select the next action a_{i+1}.

The goal of a reinforcement learning agent is to learn what action a_i to take in an arbitrary state s_i. This can be thought of as finding the optimal policy $\pi_\theta : s_i \to a_i$ where θ is the parameters that are learned. A policy $\pi_\theta(a|s)$ describes the probability of picking action a if the system is estimated to be in the state s and the policy is parametrized by θ. In Deep Reinforcement Learning (DRL), θ is the weights and biases of a neural network. This can be reformulated as an optimization problem

$$\min_\theta(L(\theta)) = 1 - \mathrm{E}_{\tau\sim\pi_\theta}[R(\tau)] = 1 - \int d\tau P_{\pi_\theta}(\tau)R(\tau), \tag{2.73}$$

where L is the loss, τ is a trajectory, the reward $R(\tau)$ is in this case the fidelity F and $P_{\pi_\theta}(\tau)$ is the probability to obtain the trajectory τ given the policy. I formulate the gate optimization problem such that the reward is only given at the end of the trajectory. Hence, a trajectory τ is defined by a sequence of states and actions ending with a reward,

$$\tau = (s_0 a_0 s_1 a_1 \dots s_{T-1} a_{T-1} s_T, R(\tau)). \tag{2.74}$$

The connection to quantum gate optimization can be made by letting a_i be the piece-wise constant controls applied to the system for the interval $t \in [i\Delta t, (i+1)\Delta t]$. The result is piece-wise constant controls, like GRAPE. Letting DRL output piece-wise constant controls is what is done in [9, 10]. The controls $\vec{C}$ can be viewed as a matrix where the columns are the actions applied at each interval, $\vec{C} = (a_0 \dots a_{T-1})$.

This study uses an on-policy DRL algorithm closely resembling the well-known REINFORCE algorithm, because [9] claims it is a good choice. There are other DRL algorithms, such as deep deterministic policy gradients used in [10]. On-policy means that only trajectories obtained when following the current policy are considered when updating θ.

The theory presented here is inspired by [31, 9, 32]. The high-level idea is to calculate $\nabla_\theta L(\theta)$ so that the parameters of the policy θ can be updated with gradient descent.

Use

$$\nabla_\theta \log(P_{\pi_\theta}(\tau)) = \frac{\nabla_\theta P_{\pi_\theta}(\tau)}{P_{\pi_\theta}(\tau)} \tag{2.75}$$

to show

$$\begin{aligned} \nabla_\theta L(\theta) = -\int d\tau \nabla_\theta P_{\pi_\theta}(\tau) R(\tau) = \\ -\int d\tau P_{\pi_\theta}(\tau) \nabla_\theta \log(P_{\pi_\theta}(\tau)) R(\tau) = -\mathrm{E}_{\tau\sim\pi_\theta}[R(\tau)\nabla_\theta \log(P_{\pi_\theta}(\tau))]. \end{aligned} \tag{2.76}$$

Use

$$P_{\pi_\theta}(\tau) = p(s_0) \prod_{i=0}^{T-1} \pi_\theta(a_i|s_i) p(s_{i+1}|s_i, a_i), \tag{2.77}$$

to rewrite using Monte-Carlo estimation

$$\begin{aligned} \nabla_\theta L(\theta) = -\mathrm{E}_{\tau\sim\pi_\theta}[R(\tau)\nabla_\theta \log(P_{\pi_\theta}(\tau))] \approx \\ -\frac{1}{N} \sum_{i=1}^{N} R(\tau_i) \sum_{t=0}^{T-1} \nabla_\theta \log(\pi_\theta(a_t^i|s_t^i)). \end{aligned} \tag{2.78}$$

Even though [9] does not, it is common to use a baseline $b(s_t)$ to decrease the variance of $R(\tau)\nabla_\theta \log(P_{\pi_\theta}(\tau))$ [32]. This is important as random variables with large variance need larger N to yield an acceptable Monte-Carlo estimate of the expectation value. For any distribution $P_\theta(x)$ parametrized by θ and random variable x, $\int P_\theta(x)\nabla_\theta \log(P_\theta(x))dx = 0$. According to [32], $\int P_\theta(x)\nabla_\theta \log(P_\theta(x))dx = 0$ can be used to show that for any baseline

$$\mathrm{E}_{\tau\sim\pi_\theta}\left[R(\tau)\nabla_\theta \log(P_{\pi_\theta}(\tau))\right] = \mathrm{E}_{\tau\sim\pi_\theta}\left[\sum_{t=0}^{T-1} \nabla_\theta \log(\pi_\theta(a_t|s_t))(R(\tau) - b(s_t))\right]. \tag{2.79}$$

Motivated by numerical experiments, it is common to use $b(s_t) = V^\pi(s_t)$ where $V^\pi(s_t)$ is the expected reward of following the policy π_θ starting from s_t. This is an intuitive choice. If $R(\tau) = V^\pi(s_t)$, the goodness of τ is what one would expect on averege. Hence, it should not contribute to $\nabla_\theta L(\theta)$. In general, $V^\pi(s_t)$ needs to be approximated, for example, using a neural network. Instead I use the mean reward $\overline{R} = \frac{1}{N}\sum_{i=1}^{N} R(\tau_i)$ as a baseline, $b = \overline{R}$, a constant. This zero centers the rewards.

I decided to parametrize the policy as a Gaussian function. The mean μ is output by a neural network while the standard deviation σ is the exploration noise, a hyperparameter that is kept constant. The input to the neural network is the state of the system at the time step s_t. μ and σ are vectors with the same dimension as the number of controls. In the single-qubit case, there are three controls $\delta(t)$, $\Omega_x(t)$, and $\Omega_y(t)$ so $\mu, \sigma \in \mathbb{R}^3$. This is not the only choice; one could, for example, discretize the action space. [9] discretizes the action space while [10] uses a Gaussian parametrization.

In practice, $\nabla_\theta L$ is calculated by backpropagation of the policy gradient loss

$$L^{\text{policy}}(\theta) = -\frac{1}{N} \sum_{i=1}^{N} \left(R(\tau_i) - \overline{R}\right) \sum_{t=0}^{T-1} \log(\pi_\theta(a_t^i|s_t^i)) \tag{2.80}$$

where, per the Gaussian parametrization,

$$\log(\pi_\theta(a_t|s_t)) = -\frac{1}{2}\sum_{k=1}^{N_k}\left(\frac{a_t^k - \mu(\theta, s_t)}{\sigma_k}\right)^2 - \log\left(\sqrt{2\pi\sigma_k^2}\right), \tag{2.81}$$

with N_k being the number of control variables at each time step. Note that when calculating the gradient, only $\mu(\theta, s_t)$ depends on θ, all other variables are viewed as constants. Backpropagation of the policy loss $L^{\text{policy}}(\theta)$ is very efficient. In this thesis, the most time-consuming step in the deep reinforcement learning algorithm is simulating the time propagator for each trajectory τ_i, $\{\hat{U}(t_g)\}_{i=1}^N$. For the single-qubit case, I have used $N = 50$, and for the two-qubit case, $N = 40$. Observe that $L^{\text{policy}} \neq L$ but $\nabla_\theta L(\theta) = \nabla_\theta L^{\text{policy}}$ when $N \to \infty$.

Here, it is described how Eq. (2.80) can be understood intuitively. Note that $\log(P_{\pi_\theta}(\tau_i)) = \log(P_0) + \sum_{t=0}^{T-1}\log(\pi_\theta(a_t^i|s_t^i))$ where P_0 depends on $p(s_0), p(s_{t+1}|s_t, a_t)$. Observe that

$$L^{\text{policy}}(\theta) = -\frac{1}{N}\sum_{i=1}^{N}\Big(R(\tau_i) - \overline{R}\Big)\log(P_{\pi_\theta}(\tau_i)) + \text{Constant}. \tag{2.82}$$

The only thing that can be influenced is $\log(P_{\pi_\theta}(\tau_i)) \leq 0$ as it is the only thing that depends on θ when minimizing L^{policy}. If the trajectory τ_i is better than the average trajectory $-\Big(R(\tau_i) - \overline{R}\Big) < 0$, implying $\log(P_{\pi_\theta}(\tau_i))$ should be as large as possible to minimize L^{policy}. Large $\log(P_{\pi_\theta}(\tau_i))$ implies large $P_{\pi_\theta}(\tau_i)$. Hence, minimization of L^{policy} will maximize the probability of sampling trajectories τ_i that are better than average. Similarly, it will minimize the probability of sampling trajectories that are worse than average.

Note that this reinforcement learning algorithm is not necessarily a state-of-the-art algorithm. A constant standard deviation σ might lead to noisy results for low infidelities. Performance could perhaps be enhanced by using a more advanced deep reinforcement learning algorithm, such as proximal policy optimization [33]. The idea of the reinforcement learning algorithm implementation is partly to be a vehicle for investigating the effect of including an ansatz in deep reinforcement learning.

2.8.4 Black-box Optimization of Piece-wise Constant Controls

For quantum gate optimization, an alternative to deep reinforcement learning is black-box optimization. Instead of sequentially generating the actions a_i based on the current state s_i, the full action sequence $(a_0 \dots a_{T-1})$ is generated immediately. Instead of optimizing the weights of a neural network as in deep reinforcement learning, $a_0 \dots a_{T-1}$ is optimized directly. The goal is to minimize the infidelity I. This can be done with many different algorithms, such as Covariance Matrix Adaptation Evolution Strategy (CMA-ES). Black-box optimization means that no analytical formula for $\frac{\partial I}{\partial a_i}$ is assumed, unlike GRAPE. This makes black-box optimization of piece-wise constant controls a model-free approach, like deep reinforcement learning. Like deep reinforcement learning and GRAPE, the actions are assumed to

correspond to piece-wise constant actions. The batch size $N = 1$, meaning that just a single control is tried for each optimization step.

3
Methods

Here, the simulation and the implemented techniques are covered in detail, the connection to the experimental set-up is made, and the gate ansatzes are explained. See my GitHub repository [34] for more details. I have devised the single-qubit simulation myself, but the two-qubit simulation is heavily based on Chalmers Superconducting Qubit Repository (CSQR), a software package developed at Chalmers. Both simulations are heavily based on QuTiP [12], a package for simulating quantum systems.

3.1 Generating Control Signals Experimentally

I attempt to consider hardware limitations when optimizing the single-qubit gate; further studies could extend this, for example, with regards to noise. A rough idea of how the optimized control signals can be generated experimentally is given.

The signal generation set-up considered in this study is described by Fig. 3.1. The set-up is an IQ-mixer feed signals from an Arbitary Waveform Generator (AWG) and a local oscillator. The set-up and description of the set-up are from [35]. The thesis does not claim that this is the only way to control qubits, nor that it is the best way.

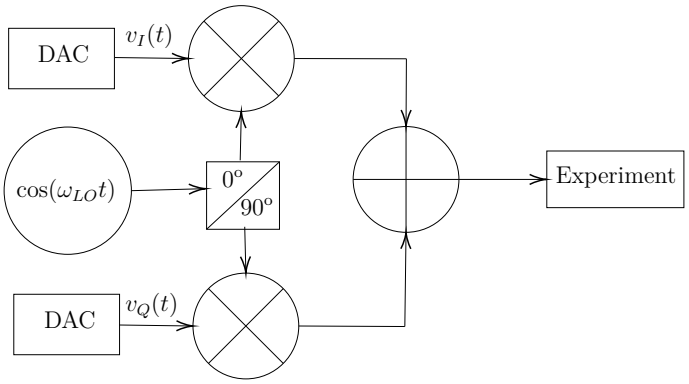


Figure 3.1: Working principle of an IQ-mixer. The AWG outputs two arbitrary band-limited signals $v_I(t), v_Q(t)$ via two independent digital-to-analog converters. They are mixed with a local oscillator sent through a 90° splitter, adding $\pi/2$ to the LO signal sent to $v_Q(t)$. The two signals are combined through a microwave combiner and sent to the experiment. The purpose of the IQ-mixer is to upconvert the frequency to go beyond the band limit of the AWG. The figure is based on [35].

3.1.1 Single-qubit Control

For single-qubit control, the task is to generate $V_d(t) \propto \xi(t) = \Omega_y(t)\cos(\omega_d t + \phi) + \Omega_x(t)\sin(\omega_d t + \phi)$. How $\Omega_x(t), \Omega_y(t), \omega_d$ impacts the single-qubit set-up is defined by Eq. (2.25). For notational convenience, let $\phi = 0$ as the global phase does not matter for dynamics. An AWG outputs a signal, for example of the form $s_{\text{AWG}} = g(t)\sin(\omega_{\text{IF}} t + \phi')$ where $v(t)$ is a pulse envelope. This can be done via a Digital-to-Analog Converter (DAC). The main limitation is that s_{AWG} is limited by the bandwidth of the AWG, limiting ω_{IF} to well below 1 GHz; hence, upconverting the frequency is required, as superconducting qubits typically have frequencies $\frac{\omega_q}{2\pi} \in [3, 6]$ GHz. This can be done by mixing the AWG signal with the signal from a local oscillator with signal $s_{\text{LO}} = \cos(\omega_{\text{LO}} t)$ using an IQ-mixer. See [35] or Fig. 3.1 for experimental details. A way presented in [35] to experimentally realise the control signal $\xi(t) = \Omega_y(t)\cos(\omega_d t + \phi) + \Omega_x(t)\sin(\omega_d t + \phi)$ is to use two digital-to-analog converters to generate $v_I(t)$ and $v_Q(t)$ where

$$\begin{bmatrix} v_I \\ v_Q \end{bmatrix} = \begin{bmatrix} \cos(\omega_{\text{IF}} t) & \sin(\omega_{\text{IF}} t) \\ -\sin(\omega_{\text{IF}} t & \cos(\omega_{\text{IF}} t) \end{bmatrix} \begin{bmatrix} \Omega_y \\ \Omega_x \end{bmatrix}. \tag{3.1}$$

The effect of an IQ-mixer is to scalar product $[v_I, v_Q]$ with $[\cos(\omega_{\text{LO}} t), \sin(\omega_{\text{LO}} t)]$. The end result is, after applying some trigonometric identities,

$$\xi(t) = \Omega_y(t)\cos((\omega_{\text{IF}} + \omega_{\text{LO}})t) + \Omega_x(t)\sin((\omega_{\text{IF}} + \omega_{\text{LO}})t). \tag{3.2}$$

Note that $\omega_d = \omega_{\text{IF}} + \omega_{\text{LO}}$. Here, it is assumed that the IQ-mixer is behaving ideally; in reality, there will be distortions.

The key to controlling the detuning $\delta(t) = \omega_q - \omega_d$ is $\omega_d = \omega_{\text{IF}} + \omega_{\text{LO}}$ as ω_q is fixed in the Chalmers set-up. The device used at Chalmers, [35], has frequency resolution 0.25 Hz for ω_{IF}, 1 Hz for ω_{LO}. How fast ω_{IF} and hence the detuning $\delta(t)$ can be modified is a bit uncertain, even though some control is possible according to experimentalists at Chalmers. That a time-dependent ω_d is possible, is not surprising as frequency modulation of signals is commonplace. Hence, time-dependent $\delta(t)$ should be possible. This study assumes identical constraints on $\Omega_x, \Omega_y, \delta(t)$. An alternative to controlling ω_{IF} is to put $\delta(t)$ in $\Omega_x(t), \Omega_y(t)$;

$$\begin{aligned} V_d(t) \propto \Omega_y(t)\cos((\omega_q + \delta(t))t) + \Omega_x(t)\sin((\omega_q + \delta(t))t) = \\ \Omega_y(t)\cos(\omega_q t)\cos(\delta(t)t) - \Omega_y(t)\sin(\omega_q t)\sin(\delta(t)t) + \\ \Omega_x(t)\sin(\omega_q t)\cos(\delta(t)t) + \Omega_x(t)\cos(\omega_q t)\sin(\delta(t)t). \end{aligned} \tag{3.3}$$

Hence

$$\begin{aligned} \Omega'_x &= \Omega_x\cos(\delta(t)t) - \Omega_y\sin(\delta(t)t) \\ \Omega'_y &= \Omega_y\cos(\delta(t)t) + \Omega_x\sin(\delta(t)t). \end{aligned} \tag{3.4}$$

3.1.2 Two-qubit Control

The most general form of flux $\Phi(t) = \theta_{DC} + g(t)\cos(\omega_{\cos} t + \phi(t)) + f(t)\sin(\omega_{\sin} t)$, used to control the two-qubit set-up, can likely be generated similarly to the single-qubit control. It is assumed that the current generated by the control electronics is

proportional to the flux. The flux can be rewritten

$$\Phi(t) = \theta + f(t)\cos(\phi(t))\cos(\omega_{\cos}t) - f(t)\sin(\phi(t))\sin(\omega_{\cos}t) + g(t)\sin(\omega_{\sin}t). \quad (3.5)$$

This is the same format as the single-qubit $V_d(t)$, for example $\Omega_x \leftrightarrow -f(t)\sin(\phi(t))$, except for the addition of an offset θ and $g(t)\sin(\omega_{\sin}t)$, something that should be possible to achieve experimentally, for example with an additional IQ-mixer. To generate the signals with an IQ-mixer, one should ensure that $f(t)\cos(\phi(t))$, $g(t)$ and $-f(t)\sin(\phi(t))$ primarily consists of frequency components below the experimental bandlimit.

3.1.3 Experimental Limitations

The use of an AWG implies constraints on the control signals. Experimentally, piece-wise constant control signals are limited by the inverse sampling rate of the AWG used to make the pulse [10]. Assuming 10^9 samples/s as given by one of the devices used at Chalmers [36], the interval length of the piece-wise constant signal $\Delta t = 1$ ns is chosen. There are devices with a higher sampling rate than [36].

An experimental device will also have a non-zero rise time due to a limited bandwidth. This means that finite pulses will be distorted as finite in time implies infinite in frequency per Heisenberg's uncertainty principle. This leads to distortion of the control signal $\vec{D}(\vec{C}(t))$. This is worth looking into when optimizing, as piece-wise constant signals, used for example in GRAPE and deep reinforcement learning, will be a lot more distorted than, for example, DRAG. This aspect of the device [36] is modeled with a second-order Bessel 750 MHz low-pass filter, implemented using SciPy's signal processing module. I decided, for simplicity, to apply the filter directly to Ω_x, Ω_y, and $\delta(t)$ such that

$$\Omega_i^{\text{filtered}}(t) = \text{Low-pass}(\Omega_i(t)). \quad (3.6)$$

$\delta(t)$ is assumed to have the same bandwidth limitations as Ω_x, Ω_y. This captures the smoothing effect of limited bandwidth on piece-wise constant pulses. The low-pass filter is not applied to the two-qubit controls.

Experimentally, the bandwidth limit is applied to the output of the digital-to-analog converter in the AWG. Hence, it would be more realistic to apply the low-pass filter to $v_I(t), v_Q(t)$. $v_I(t), v_Q(t)$ are defined in Eq. (3.1). This complicates $\xi(t)$ as one can no longer simply replace $\Omega_x \to \Omega_x^{\text{Low-pass}}$. It seems likely based on preliminary numerical experiments that if $\omega_{\text{IF}} \ll 750$ MHz, applying the low-pass filter to Ω_x, Ω_y, and $\delta(t)$ is a good approximation of low-pass filtering $v_I(t)$ and $v_Q(t)$. This is something future studies could consider.

Handling signal distortions is significant to achieve high-fidelity quantum gates [37]. This is significant from the point of view of analytical ansatzes, as they are often derived without considering signal distortions.

The effect of a more general noisy pulse distortion is considered briefly by adding piece-wise constant Gaussian noise in addition to the low-pass filter,

$$\Omega_i^{\text{Noisy}} = \text{Low-pass}\left((1+n)\,\Omega_i^{\text{PWC}}(t)\right), n \sim \mathcal{N}\left(0, 10^{-2}\right), \quad (3.7)$$

where $\mathcal{N}(\mu, \sigma)$ is a Gaussian distribution with mean μ and standard deviation σ. More general distortions of control signals than low-pass are likely in an experimental set-up; this line of research is a topic for other studies.

3.2 Deep Reinforcement Learning

Here, details about how the deep reinforcement learning agent is implemented are given and discussed. Focus is on how the state s input to the neural network is described and the two ways to include an ansatz that have been compared.

The deep reinforcement learning algorithm is implemented with JAX [38] and optimized using the ADAM [39] optimizer. A lot of hyperparameters exist, and they have been tuned differently for the single-qubit and the two-qubit cases. Tuning hyperparameters is a challenge, as simulations are time-consuming, especially for the two-qubit system. For the thesis, I have used the VERA computer cluster, but more tuning could perhaps improve the performance of deep reinforcement learning relative to, for example, DRAG. The performance of reinforcement learning algorithms is known to be sensitive to hyperparameters [40]. The neural network is, like in [9], rather small. For the single-qubit case, a single hidden neural network layer of size 10 is used, while for the two-qubit results shown in the thesis, two hidden layers, each of size 5, are used. For the single-qubit case, the batch size N is 50, while in the two-qubit case it is 40. Like in [9], tanh is used as the activation function for hidden layers.

3.2.1 The State s_i Input

Many different kinds of state inputs s_i to the deep reinforcement neural network could be conceived. I have considered two different approaches, one that includes information about the unitary time-evolution operator of the system at step i, and one that does not use any information about the quantum state of the system. Including information from $\hat{U}(t_i)$ is similar to the approach used in [9, 10], while not doing so is similar to the approach used in [7, 8].

The unitary

$$U(t_i) = \sum_m \sum_n U_{mn} \left|m\right\rangle \left\langle n\right| \tag{3.8}$$

is encoded in the state by the real and imaginary value of $\{U(t_i)_{mn}\}$ where $m \in \{0, 1, 2\}$ and $n \in \{0, 1\}$. By picking m and n in this way, information about leakage from the computational subspace to the first excited level $|2\rangle$ at time t_i is given to the deep reinforcement learning agent. This is highly similar to the approach used in [10]. Using just a part of the unitary by keeping m and n small is beneficial, as it is easier to measure experimentally. For the unitary containing s_i I use

$$s_i = [\Re(U_{00}), \Im(U_{00}) \ldots, \Re(U_{21}), \Im(U_{21}), ai + b], \tag{3.9}$$

where a and b are constants that can be chosen freely and $z = \Re(z) + i\Im(z)$. For the state without unitary information, I use

$$s_i = [ai + b]. \tag{3.10}$$

For the two-qubit case, only the approach without $\hat{U}(t_i)$ is used, as the single-qubit case indicated that not including $\hat{U}(t_i)$ in s_i might be preferable.

a and b can be chosen to normalize $ai + b$ to some chosen interval. In the single-qubit case, $ai + b$ is in the interval [-1,1], while in the two-qubit case, $ai + b$ is in [-3, 15]. The choice of interval is something future studies could look into more.

The intuition for why deep reinforcement learning for both s_i designs could be beneficial relative to directly optimizing piece-wise constant controls with a black-box optimizer is that $ai + b$ encodes that there is a relationship between the controls at different time points. In the single-qubit case, this should make it easier to learn that $\Omega_x(t_i)$ has a different relationship to $\Omega_x(t_{i+1})$ than $\Omega_x(t_{i+8})$. Another advantage is that $ai + b$ encodes that controls with the same i, for example $\Omega_x(t_i)$, $\Omega_y(t_i)$, and $\delta(t_i)$, are related.

3.2.2 Including an Ansatz

A focus of this study is to look into the effect of including an ansatz in deep reinforcement learning when optimising quantum gates. That can be done in different ways. One way is to add the ansatz directly to the output of the DRL neural network, so that the controls

$$C(t) = C_{\text{ansatz}}(t) + C_{\text{NN}}(t). \tag{3.11}$$

This ansatz plus NN approach can be contrasted with the ansatz-free case $C(t) = C_{\text{NN}}(t)$. The inspiration for this approach is that it is very simple to implement. One can ensure that the fidelity of the initialized $C(t)$ is about as good as the ansatz by initializing the output of the neural network $C_{\text{NN}}(t)$ to be small.

[6] proposes something similar to the ansatz plus NN approach. A difference is that the ansatz they use is DRAG, while I use a square-shaped ansatz. [6] restricts themself to single-qubit gates. They also use a black-box optimization method, CMA-ES, not reinforcement learning, to optimize the piece-wise constant controls added to the ansatz. They also do not consider control of the detuning $\delta(t)$ and implement the approach in experiments. The existence of this study was something I was not aware of when implementing the ansatz plus NN approach.

[7] suggests pre-training the neural network with supervised learning, effectively initializing the neural network so that $C_{\text{NN}}(t) = C_{\text{ansatz}}(t)$. It is investigated in detail by [41] and applied experimentally by [8]. Another paper following that approach is [42], but applied to robotics. In some sense, this can be thought of as transfer learning, teaching the agent the knowledge in the ansatz. The pre-training is implemented in this study by minimizing the pre-training loss

$$L_{\text{pre-training}} = \frac{1}{N}\sum_{i=1}^{N}\left(C_{\text{NN}}(t_i) - C_{\text{ansatz}}(t_i)\right)^2 \tag{3.12}$$

with ADAM before starting deep reinforcement learning optimization. With this approach, no ansatz is added to the output of the network.

3.3 Ansatzes

Here, the ansatzes C_{ansatz} supplied to the deep reinforcement learning algorithm are described and analyzed. How they are extended by deep reinforcement learning is also mentioned.

3.3.1 Single-qubit Ansatz

Assuming no leakage, $\Omega_y(t) = 0, \delta(t) = 0$, a π-pulse can be achieved by

$$\int_0^{t_g} \Omega_x(t)dt = \pi. \tag{3.13}$$

For details, see section A.1. The simplest solution is $\Omega_x^0(t) = \pi/t_g$, a square signal. I limited the single-qubit ansatz to the square signal

$$C_{\text{ansatz}}(t) = [\Omega_x^0(t), \Omega_y^0(t), \delta^0(t)] = [\pi/t_g, 0, 0] \forall t \in [0, t_g]. \tag{3.14}$$

The deep reinforcement learning agent outputs

$$C_{\text{NN}}(t) = [\Omega_x^{\text{NN}}(t), \Omega_y^{\text{NN}}(t), \delta^{\text{NN}}(t)]. \tag{3.15}$$

For the ansatz plus NN approach to including an ansatz in deep reinforcement learning, the controls sent to the simulated qubit, or the low-pass filter if a low-pass filter is considered, are

$$C(t) = [\Omega_x^0(t) + \Omega_x^{\text{NN}}(t), \Omega_y^0(t) + \Omega_y^{\text{NN}}(t), \delta^0(t) + \delta^{\text{NN}}(t)]. \tag{3.16}$$

The square signal is also used to initialize GRAPE.

3.3.2 Two-qubit Ansatz

A standard approach to implement a two-qubit CZ gate, as suggested by [43], is to drive the $|110\rangle \leftrightarrow |200\rangle$ transition. The implementation on purpose leaves the computational subspace. The transition is activated by driving at a frequency close to $\omega_{110\leftrightarrow200} = |E_{110} - E_{200}|$. This thesis focuses on this way of implementing a CZ gate. The ansatz for the two-qubit control flux $\Phi(t)$ is therefore

$$\Phi(t) = \theta + \delta_{\cos} \cos(\omega_{\cos} t) + \delta_{\sin} \sin(\omega_{\sin} t), \tag{3.17}$$

where $\theta, \delta_{\cos}, \omega_{\cos}, \delta_{\sin}, \omega_{\sin}$ are optimization parameters. The driving frequencies $\omega_{\cos}$ and $\omega_{\sin}$ is parameterized by $(i = \cos, \sin)$ $\Delta\omega_i$ where $\omega_i = \omega_{110\leftrightarrow200} + \Delta\omega_i$. To activate $|110\rangle \leftrightarrow |200\rangle$, $|\Delta\omega_i|$ must be small. I have also tested optimizing $(\Delta\omega_{\cos}, \Delta\omega_{\sin})$ without that constraint, leading to different ways of implementing a CZ gate. The ansatz (Eq. (3.17)) is an extension of [11, 43] where

$$\Phi(t) = \theta + \delta_{\cos} \cos(\omega_{\cos} t + \varphi) \tag{3.18}$$

is used, but I have not been able to show a significant advantage of having non-zero $\omega_{\sin}$ and $\delta_{\sin}$. $\delta_{\sin} = 0$ was therefore sometimes enforced. More investigation is perhaps needed, but the choice of offset θ seems more significant.

At the beginning and end of the gate, the ansatz $\Phi(t)$ is smoothed by a cosine shape such that

$$\Phi(t) = \begin{cases} \theta + \frac{\delta_{\cos}}{2}\left(1 - \cos\left(\pi\frac{t}{t_r}\right)\right)\left(\cos(\omega_{\sin}t) + \delta_{\sin}\sin\left(\omega_{\sin}t\right)\right) & 0 \leq t \leq t_r \\ \theta + \delta_{\cos}\cos(\omega_{\cos}t) + \delta_{\sin}\sin\left(\omega_{\sin}t\right) & t_r \leq t \leq t_p + t_r \\ \theta + \frac{\delta_{\cos}}{2}\left(1 - \cos\left(\pi\frac{t-t_p}{t_f}\right)\right)\left(\cos(\omega_{\cos}t) + \delta_{\sin}\sin\left(\omega_{\sin}t\right)\right) & t_p + t_r \leq t \leq t_g, \end{cases} \tag{3.19}$$

where t_r is the rise time, t_f the fall time, t_p is the "plateau" time, the gate time $t_g = t_r + t_p + t_f$. I have constrained the ansatz to $t_r = t_p$. The idea behind smoothing is that smooth signals tend to have a better intensity-to-bandwidth relationship. For the results shown in this thesis $\delta_{\sin} \ll \delta_{\cos}$, therefore $\Phi(t)$ and $\frac{\partial\Phi}{\partial t}$ becomes almost continuous. Note that $t_r < 1$ ns risks pushing $1 - \cos\left(\pi\frac{t}{t_r}\right)$ beyond the bandlimit of [36], the AWG used at Chalmers.

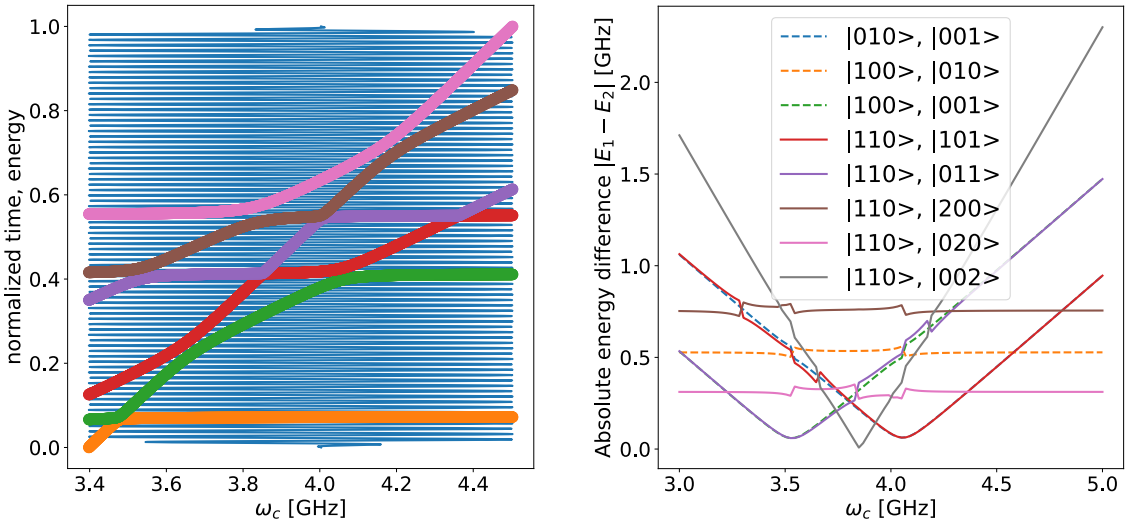
Deep reinforcement learning is introduced to the ansatz by making θ piece-wise constant instead of constant and introducing a piece-wise constant phase ϕ when $t_p + t_r \geq t \geq t_r$ such that

$$\Phi(t) = \theta + \delta_{\cos}(t)\cos(\omega_{\cos}t + \phi(t)) + \delta_{\sin}\sin\left(\omega_{\sin}t\right). \tag{3.20}$$

Deep reinforcement learning is combined with the ansatz by letting $\delta_{\cos}(t) = \delta^0_{\cos} + \delta^{\mathrm{NN}}_{\cos}(t)$ where $\delta^0_{\cos}$ is the constant $\delta_{\cos}$ of the ansatz Eq. (3.19) and $\delta^{\mathrm{NN}}_{\cos}(t)$ the deep reinforcement learning output. Similarly, $\phi(t) = \phi^{\mathrm{NN}}(t)$. The length of the piece-wise constant interval is $\Delta t = 1$ ns. The deep reinforcement learning agent also outputs a small change δt to t_r and t_f. Generating δt with deep reinforcement learning is implemented in a similar way to the other time-dependent controls.

The $|110\rangle \leftrightarrow |200\rangle$ transition reasoning leaves $t_r, t_p, \delta_{\cos}, \delta_{\sin}, \theta$ of the ansatz Eq. 3.19 undefined. I have optimized them numerically, but tried to constrain the numerical optimizer to intervals where interesting results might be expected based on the energy spectrum of the system. This is based on qualitative reasoning. I am not aware of a rigorous theory for what value, for example, θ should have to achieve an optimal CZ gate.

The energy spectrum of the two-qubit system is shown in Fig. 3.2 from two different perspectives. The energy spectrum is given by the eigenenergies of the two-qubit Hamiltonian $\hat{H}_{\text{2-qb}}(\omega_c)$ (defined in Eq. (2.26)). Based on the energy difference spectrum shown in Fig. 3.2b, it seems like picking θ such that the average $\overline{\omega}_c \approx 3.8$ GHz might be beneficial as the energy difference $|E_{110} - E_{200}|$ of the $|110\rangle \leftrightarrow |200\rangle$ is rather separated from other transitions. Notably, Fig. 3.2a shows that for the optimized 108.3 ns ansatz, the optimized ansatz oscillates over a large range of coupler frequencies $\omega_c(t)$ corresponding to many energy differences, even though $|E_{110} - E_{200}|$ is rather constant. In my experience, all numerically optimized ansatzes seem to exhibit the large oscillations in $\omega_c(t)$ behavior. Understanding the dynamics of such large oscillations in $\omega_c(t)$ is challenging. From numerical experiments, it seems that the choice of θ is very important for the infidelity.



(a) Normalized two excitation spectrum (thick lines) with normalized ansatz

(b) Energy difference for different transitions

Figure 3.2: Energy spectrum of the two-qubit system shown from two different perspectives. Fig. 3.2a shows the two-excitation energy spectrum (marked with thick lines) with an optimized ansatz with $t_g = 108.3$ ns. The two-excitation energy spectrum is the spectrum corresponding to $\{|ijk\rangle\}$ where $i+j+k = 2$. The optimized flux $\Phi(t)$ ansatz implies large oscillations in $\omega_c(t)$. The time and energy scale have been normalized. Fig. 3.2b shows the energy difference $|E_{ijk} - E_{klm}|$ of transitions $|ijk\rangle \leftrightarrow |klm\rangle$ within the single-excitation (marked with dashed curves) and two-excitation manifolds.

3.4 Simulating $\hat{U}(t_g)$

In order to calculate fidelity, the projection of the time-evolution operator $\hat{U}(t_g)$ in the computational subspace, $\hat{\Pi}\hat{U}(t_g)\hat{\Pi}$ is required. There are many ways to calculate $\hat{U}(t_g)$ in a simulation. I consider two different approaches in the thesis. The two approaches seem to result in highly similar infidelities; the relative difference in infidelity seems to be less than about 1%, see Fig. C.1 for details.

The first approach to calculate $\hat{U}(t_g)$ is matrix exponentiation, which can be motivated by a first-order Magnus expansion and a first-order Baker-Campbell-Hausdorff expansion,

$$\hat{U}(t) \approx \exp\left(\frac{-i}{\hbar}\int_0^t \hat{H}(t)dt\right) \approx \prod_{t_i=t_0}^{t} \exp\left(\frac{-i}{\hbar}\hat{H}(t_i)\Delta t\right). \tag{3.21}$$

For details, see the GRAPE theory, section 2.8.2. This can only be done in a simulation and has no experimental analogy. Advantages are that one ensures that $\hat{U}(t_g)$ is unitary, the full $\hat{U}(t_g)$ is calculated, and it is fast for piece-wise constant controls.

Another approach to calculate $\hat{U}(t_g)$ is to propagate the computational basis set using the Schrödinger equation. The obtained quantum states, after truncating to

the computational space, give $\hat{\Pi}U(\hat{t}_g)\hat{\Pi}$ directly. $\hat{U}(t_g)$. This is roughly what is done experimentally when measuring $\hat{U}(t_g)$ with quantum process tomography. To do this, I use QuTiP [12], which solves the Schrödinger equation using ZVODE. An advantage of this method is that one never needs to calculate the full $U(t_g)$.

Note that there is a different way to obtain infidelity, randomized benchmarking. This does not require calculating $\hat{U}(t_g)$. Randomized benchmarking is often the method of choice in experiments, as the quantum process tomography needed to obtain $\hat{U}(t_g)$ is computationally expensive in experiments. The idea is to repeat the same gate many times, interleaved with random single-qubit gates corresponding to different rotations on the Bloch sphere. Analytically, one can calculate what the inverse of the random sequence of gates should be. Apply that gate at the end of the sequence. If everything works as expected, the procedure should return the starting state. If not, an error occurred. By measuring the error rate as the number of gates applied varies, one can isolate the error per gate from the state preparation and measurement error. The approach corresponds to sampling over all possible contexts in which the gate could be applied, as opposed to over all states in which it could be applied. For more details, see [44]. As this study is based on simulations, it was decided to use quantum process tomography as it is less computationally expensive than randomized benchmarking when simulating quantum systems with a classical computer.

3.4.1 Calibration of $\hat{U}(t_g)$ for Two-qubit Gates

For two-qubit CZ gates, any phase gained by $|100\rangle$ and $|010\rangle$ due to the time-evolution operator $\hat{U}(t_g)$ can be calibrated away, increasing the fidelity of the gate. In this subsection, the notation is simplified by implicitly assuming the coupler to be in the ground state. Hence $|100\rangle$ is simplified into $|10\rangle$. [21] explains how virtual Z gates are used when calculating the fidelity of the CZ gate. The key is to get rid of the unwanted phase accumulations ϕ_{10} and ϕ_{01} of the $|10\rangle$ and $|01\rangle$ states, where ϕ_{10} is defined by $\hat{U}(t_g)|10\rangle = e^{i\phi_{10}}|10\rangle$ and ϕ_{01} is defined by $\hat{U}(t_g)|01\rangle = e^{i\phi_{01}}|01\rangle$. [45] shows that it is possible to effectively apply arbitrary rotations around the z-axis of the Bloch sphere

$$R_Z^{(i)}(\theta) = e^{i\frac{\theta}{2}\hat{\sigma}_z^{(i)}} = \cos\left(\frac{\theta}{2}\right)\hat{I} - i\sin\left(\frac{\theta}{2}\right)\hat{\sigma}_z^{(i)} \tag{3.22}$$

where

$$\sigma_z = |0\rangle\langle 0| - |1\rangle\langle 1| \tag{3.23}$$

to any qubit i in no time, without any errors, by varying the frame of reference. Additionally, the global phase can be modified freely. This means $e^{i\theta/2}R_Z(\theta)|0\rangle = |0\rangle$, $e^{i\theta/2}R_Z(\theta)|1\rangle = e^{i\theta}|1\rangle$. Assume unintended phases added to $|10\rangle$ and $|01\rangle$,

$$\hat{U}(t_g) = e^{i\phi_{10}}|10\rangle\langle 10| + e^{i\phi_{01}}|01\rangle\langle 01| + |00\rangle\langle 00| + e^{i(\phi_{11}+\phi_{10}+\phi_{01})}|11\rangle\langle 11|. \tag{3.24}$$

Then

$$e^{\frac{\phi_{10}+\phi_{01}}{2}}R_Z^{(1)}(\theta_1)R_Z^{(2)}(\theta_2)\hat{U}(t) = e^{i(\phi_{10}+\theta_1)}|10\rangle\langle 10| + e^{i(\phi_{01}+\theta_2)}|01\rangle\langle 01| + |00\rangle\langle 00| + e^{i(\phi_{11}+\phi_{10}+\phi_{01}+\theta_1+\theta_2)}|11\rangle\langle 11| \tag{3.25}$$

Let $\theta_1 = -\phi_{10}$, $\theta_2 = -\phi_{01}$. Then

$$\hat{U}(t) = |10\rangle\langle 10| + |01\rangle\langle 01| + |00\rangle\langle 00| + e^{i\phi_{11}}|11\rangle\langle 11|, \tag{3.26}$$

which is closer to the target gate

$$\hat{U}_T = |10\rangle\langle 10| + |01\rangle\langle 01| + |00\rangle\langle 00| - |11\rangle\langle 11|. \tag{3.27}$$

This procedure increases the fidelity of the two-qubit gate. Adopting GRAPE to consider the calibration will require some work; this is not done in this study. As such, GRAPE is only applied to single-qubit gates.

3.5 Derivative Removal by Adiabatic Gate

In order to numerically optimize DRAG, coefficients are introduced. DRAG coefficients are optimized numerically based on the simulated single-qubit system. Following the recipe of [24] Eq. (2.60) is parametrized as

$$\begin{aligned} \Omega_x(t) &= \alpha\Omega_G(t) \\ \Omega_y(t) &= -\beta\frac{\dot{\Omega}_G(t)}{2\alpha} \\ \delta(t) &= -\gamma\frac{\Omega_G^2(t)}{4\alpha} + \delta_0 \end{aligned} \tag{3.28}$$

where $[\alpha, \beta, \gamma, \delta_0]$ are dimensionless fitting parameters to be optimized and $\Omega_G(t)$ is a gaussian pulse shape defined by Eq. (2.61). The fitting parameters are initialized to [1, 1, 0, 0]. Note that for the hardware parameters considered, $\frac{\Omega_G^2(t)}{4\alpha} \gg 1$, hence δ_0 is likely not so signficant. Note that there are several advanced techniques that could enhance the performance of DRAG [21, 28].

3.6 Black-box Optimization

For direct optimization of piece-wise constant controls, the two-qubit ansatz, and to optimize DRAG coefficients, I use Nevergrad [46], an open-source state-of-the-art gradient-free optimizer. The key idea of Nevergrad is to select the best optimizer for a given problem automatically based on benchmark data. It can even combine different algorithms. Nevergrad uses a portfolio of optimization algorithms, such as CMA-ES and differential evolution.

4
Results

Here, the results of the single-qubit and two-qubit simulations are presented, analyzed, and interpreted. In the single-qubit section, for example, evidence that an ansatz can speed up deep reinforcement learning optimization is presented. In the two-qubit section, evidence that deep reinforcement learning can slightly improve an ansatz is shown.

4.1 Single-qubit Results

Here, I look into the single-qubit π-pulse in detail, comparing the different deep reinforcement learning configurations considered with the other optimization methods. I also consider the effect of including a low-pass filter to mimic the limited bandwidth of experimental control electronics. Decoherence is modeled with Eq. (2.22) where $T_1 = 80$ µs, $T_2 = 40$ µs.

4.1.1 Deep Reinforcement Learning

In this section, I investigate the effect of including an ansatz in DRL, compare time-evolution-free s_i with using the time-evolution operator in s_i, compare the pretraining and ansatz plus NN approach to include an ansatz in DRL, and investigate the effect of including a low-pass filter. Due to DRL being computationally expensive, this section is limited to the gate time $t_g = 8$ ns. GRAPE and DRAG results for $t_g = 8$ ns are included for comparison with DRL. The DRAG results are when the Gaussian, derivative, detuning, and detuning offset coefficients ($[\alpha, \beta, \gamma, \delta_0]$ in Eq. (2.60)) are numerically optimized with Nevergrad. The results for DRAG and GRAPE are covered in more detail in section 4.1.2 and section 4.1.3.

Fig. 4.1 compares DRAG, GRAPE, deep reinforcement learning without an ansatz (NN), and the ansatz plus NN approach to include an ansatz in deep reinforcement learning. The results are for the time-evolution-free s_i. Fig. 4.1 seems to indicate that using an ansatz can make deep reinforcement learning converge faster, as the decoherence infidelity is reached in about 100-200 optimization steps with an ansatz, while without an ansatz, it takes about 250-350 optimization steps. Hence, using an ansatz seems to enhance deep reinforcement learning for quantum gate optimization. Nevergrad [46] is a state-of-the-art black box optimizer, for example, preserving a memory of previous iterations. That is why the infidelity only decreases for DRAG, as the DRAG coefficients are optimized with Nevergrad. The noisiness of the deep reinforcement learning infidelities could perhaps be explained by using a constant

exploration noise σ. See section 2.8.3 for details about the exploration noise. Future studies could look into making σ decrease with the number of optimization steps. Fig. 4.1 also indicates that all four methods can reach similar minimum infidelity, but how fast they do so and with what certainty varies.

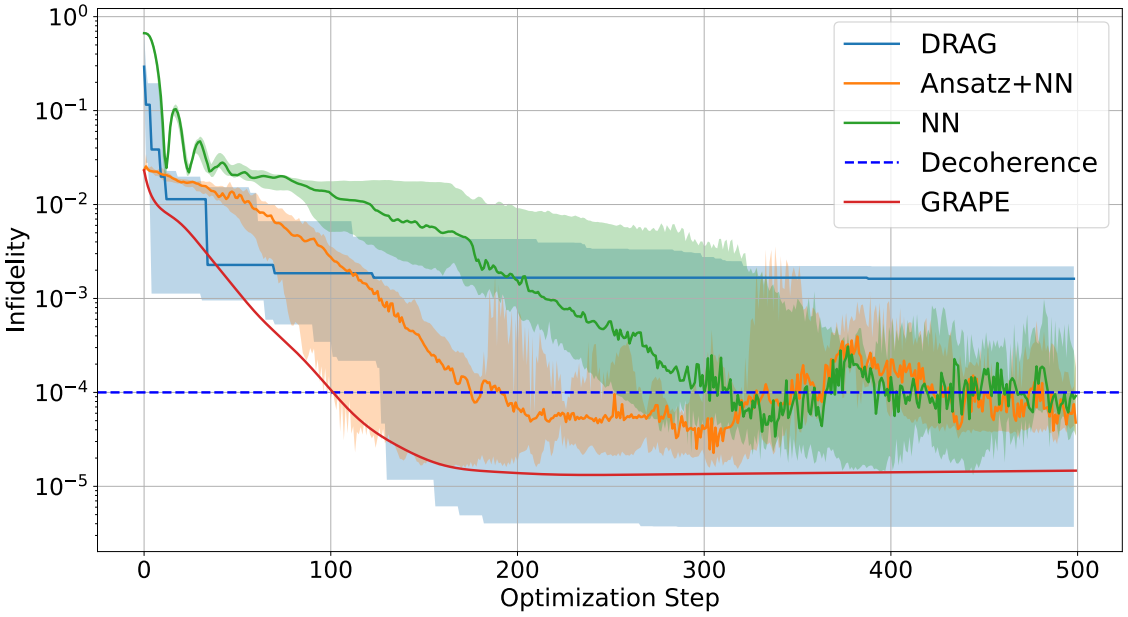
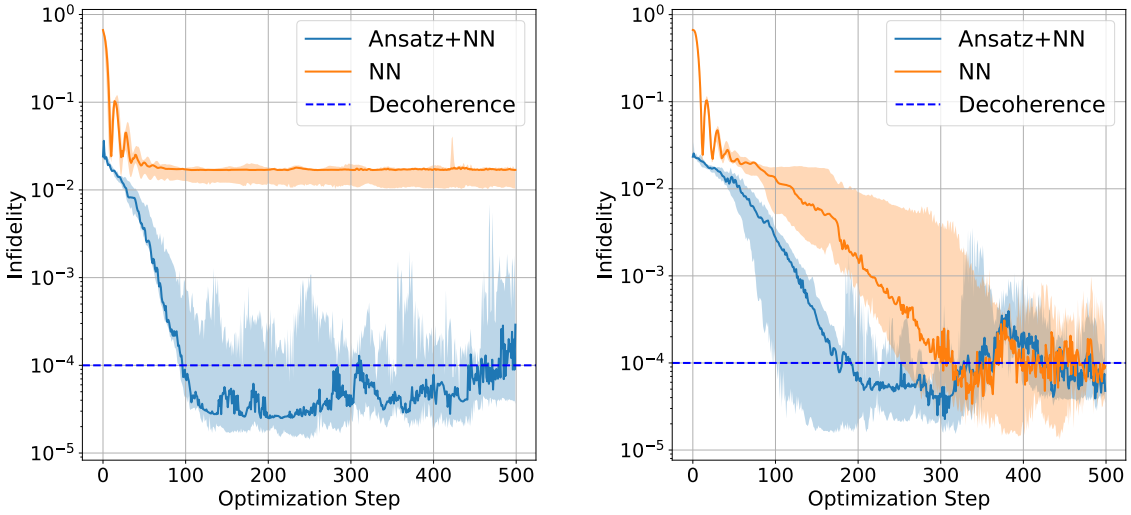


Figure 4.1: Infidelity of optimized single-qubit π-pulse for $t_g = 8$ ns for DRAG, GRAPE, the ansatz plus NN deep reinforcement learning approach, and deep reinforcement learning without an ansatz (NN). The decoherence infidelity is also shown. The DRL infidelities are for time-evolution-free DRL state input s_i. The shaded area is the minima and maxima over randomized seeds, but with the same hyperparameters. The curve is the median. For DRAG and the deep reinforcement learning methods, I use five random seeds. GRAPE is not a stochastic algorithm. For each optimization method, I use 500 optimization steps.

Fig. 4.2 shows the infidelity of deep reinforcement learning without an ansatz and the ansatz plus NN approach for inputs s_i to the deep reinforcement learning neural network that uses the time-evolution operator (Fig. 4.2a) and that does not use the time-evolution operator (Fig 4.2b). Fig. 4.2 shows that there seems to be no major advantage in putting information from the time-evolution operator in s_i. Having the time-evolution operator might even have a significant negative impact, as deep reinforcement learning without an ansatz performs significantly worse in Fig. 4.2a. I do not rule out, for example, that deep reinforcement learning with much larger neural networks could show a positive impact of including the time-evolution operator $U(t_i)$ in the state input s_i, but for the reinforcement learning implementation considered, having the time-evolution operator in s_i does not seem beneficial.



(a) Info from time-evolution in s_i **(b)** Time-evolution-free s_i

Figure 4.2: Infidelity of optimized single-qubit $t_g = 8$ ns π-pulse optimized with deep reinforcement learning without an ansatz and the ansatz plus NN approach when some of the time-evolution operator $U(t_i)$ is included in the deep reinforcement learning input s_i (Fig. 4.2a) and when s_i is time-evolution-free (Fig. 4.2b). For both cases, five random seeds are used. The decoherence marked is for $T_1 = 80$ μs, $T_2 = 40$ μs. There seems to be no significant advantage in having information from the time-evolution operator in the state s_i given as input to deep reinforcement learning.

That including information about the time-evolution operator in s_i does not yield a performance boost means the time-evolution-free s_i approach is likely preferable, as it is much easier to implement experimentally. Time-evolution-free s_i is easier to implement experimentally because measuring the time-evolution operator at time t_i will collapse the quantum state. [9] estimates the state at each step by sampling 256 times for each trajectory in the batch. The fidelity measurement is done with 1024 samples. This means not measuring the state at in between steps when there are n steps in total is $(1024 + 256n)/1024$ times faster, assuming, rather reasonably per [47], the time per sample is dominated by readout. For single-qubit gates consisting of $n = 8$ piece-wise constant intervals, in this thesis corresponding to $t_g = 8$ ns, this implies a three times speed-up. With the interval length of this thesis, a 100 ns two-qubit gate would be 26 times faster to optimize using the time-evolution-free s_i. From here on, all deep reinforcement learning results are for the time-evolution-free s_i.

Fig. 4.3 compares the pre-training concept used in [8, 42, 7] with the ansatz plus NN approach similar to the approach used in [6]. Fig. 4.3 indicates that pre-training and ansatz plus NN perform similarly; the two methods have a similar minimum infidelity and noisy fluctuations. Pre-training might have a slight advantage; it reaches the coherence limit slightly faster, but it could also plausibly be an effect of using just a few random seeds. An advantage of the ansatz plus NN approach is that no new hyperparameters are introduced; the initialization method is simpler than pre-training. It is not so surprising that pre-training and ansatz plus NN

yield similar results, as pre-training strives to make $C_{\text{NN}}(t) = C_{\text{ansatz}}(t), \forall t$ before starting optimization with deep reinforcement learning. Notably, there are several ways to minimize $|C_{\text{NN}}(t) - C_{\text{ansatz}}(t)|^2$ so pre-training can yield more or less good initializations. Comparing pre-training and ansatz plus NN for more complicated ansatzes than a square signal could be interesting.

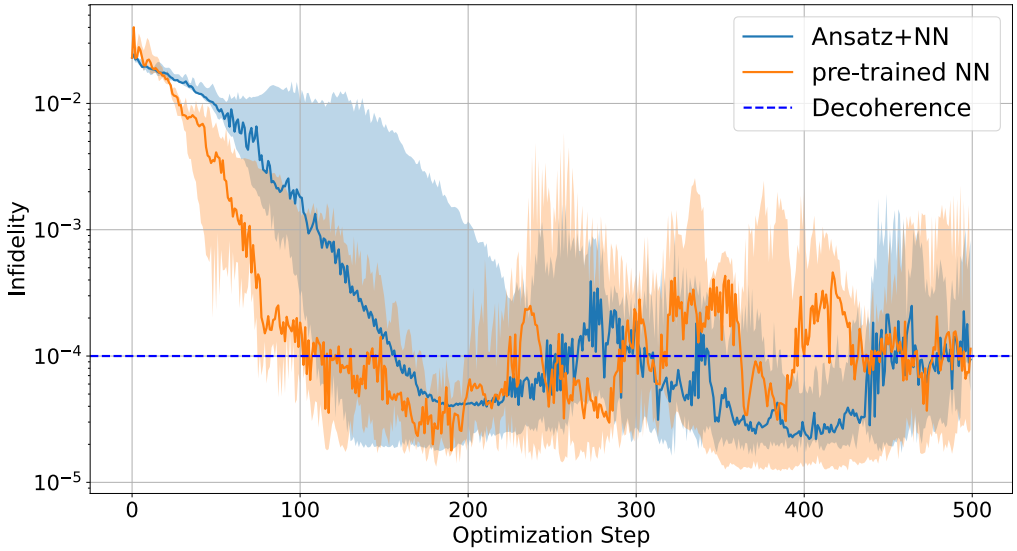
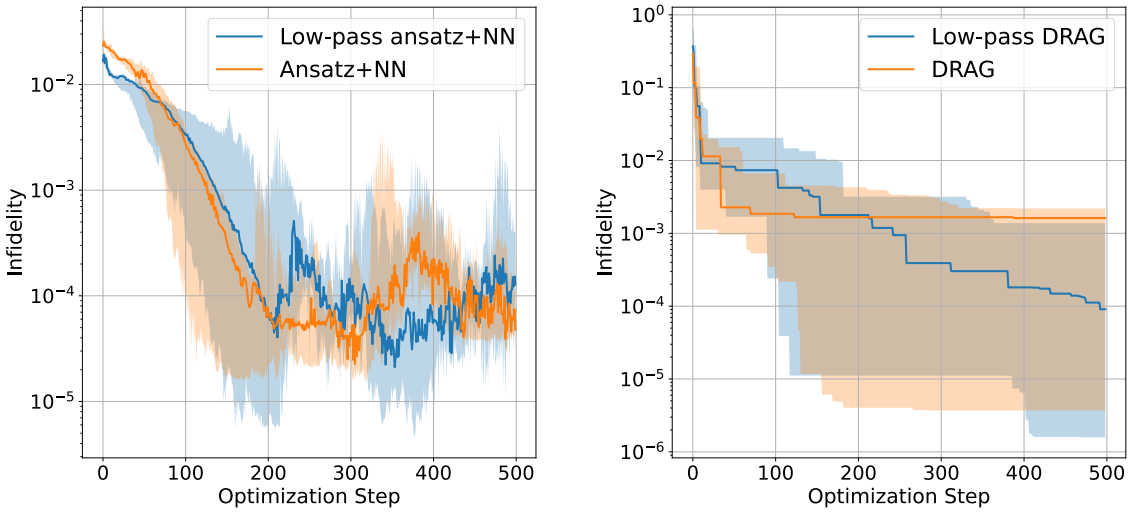


Figure 4.3: Infidelity of optimized single-qubit π-pulse for $t_g = 8$ ns for pre-training and the ansatz plus NN deep reinforcement learning approaches. The shaded area is the minima and maxima over randomized seeds, but with the same hyperparameters. The line is the median. Five random seeds are used for the deep reinforcement learning methods.

The effect of the low-pass filter (explained in section 3.1.3) when applied to DRAG and the ansatz plus NN DRL approach is shown in Fig. 4.4. DRAG and the Ansatz plus NN DRL approach are in Fig. 4.4 aware of the low-pass filter, meaning that they are optimized using the infidelities obtained from a control pulse that has passed through the filter. Fig. 4.4 indicates that the low-pass filter does not seem to have a significant impact on DRAG as the shaded areas have a large overlap, but the low-pass filter seems able to lower the infidelity of deep reinforcement learning below 10^{-5}. How likely the low-pass filter is to lower the infidelity of DRL is uncertain, as only a limited number of seeds are used in Fig. 4.4, and the median in 4.4a is similar for the filtered and unfiltered ansatz plus NN approach. However, Fig. 4.3 uses different seeds from Fig. 4.4a, and the infidelity of the unfiltered ansatz plus NN approach also in Fig. 4.3 remains above 10^{-5} for $t_g = 8$ ns. There is no example in this thesis of deep reinforcement learning without a low-pass filter going below infidelity 10^{-5}. It makes sense that ansatz plus NN is impacted but not DRAG, as ansatz plus NN DRL yields piece-wise constant (PWC) controls, hence not very smooth, while DRAG is already fairly smooth, and as such, the filter should have less effect. That smoother controls yield better results is reasonable, as smoother signals have a more beneficial intensity-to-bandwidth relationship. Notably, the ansatz plus NN approach is noisy with and without a filter.



(a) Ansatz plus NN DRL **(b)** DRAG

Figure 4.4: Infidelity of DRAG and the ansatz plus NN approach to deep reinforcement learning when filtered with a low-pass filter. The task is to optimize a single-qubit π-pulse with $t_g = 8$ ns. The marked curve is the median, the shaded area is the minima and maxima over randomized seeds. The figure indicates that the low-pass filter can push the ansatz plus NN infidelity down a bit. For both DRAG and DRL, five different random seeds are used. Both DRL and DRAG are aware of the filter when optimizing.

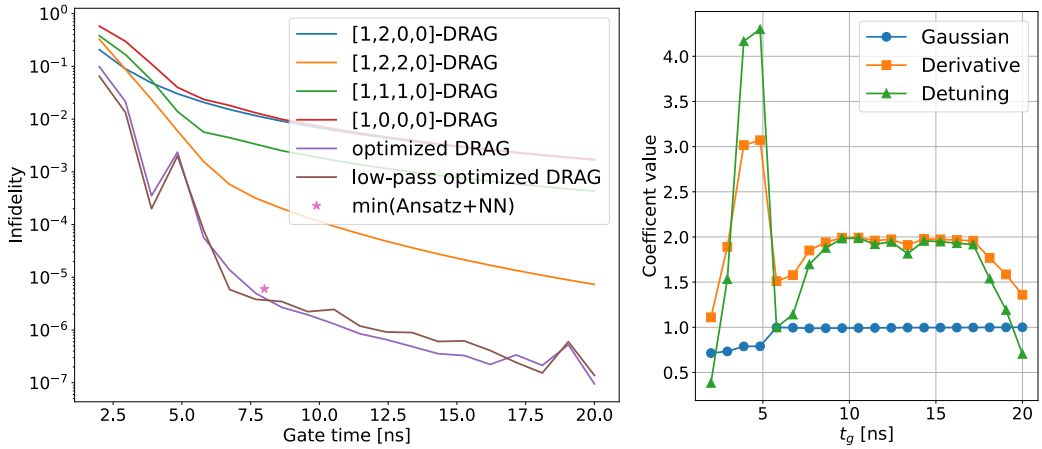
4.1.2 Derivative Removal by Adiabatic Gate

A very common approach to single-qubit optimization is DRAG. Here, DRAG's infidelities, control shapes, and gate dynamics are analyzed in detail.

Fig. 4.5a shows the infidelities of DRAG when optimizing the DRAG coefficients with Nevergrad, ([Gaussian, derivative, detuning, detuning offset] corresponding to $[\alpha, \beta, \gamma, \delta_0]$ in Eq. (3.28)). Fig. 4.5b shows what coefficients optimized DRAG corresponds to. The detuning offset δ_0 is not shown in Fig. 4.5b despite fluctuating a lot because replacing $\delta_0 \to 0$ $\forall t_g$ has a negligible impact on the infidelity. It is not surprising that δ_0 does not matter as a large number upscales the detuning coefficient γ. The relative change in infidelity I by picking $\delta_0 = 0$ is

$$\frac{|I_{\delta_0=0} - I_{\delta_0}|}{I_{\delta_0}} \ll 10^{-5}. \tag{4.1}$$

The infidelities of the optimized DRAG coefficients when a low-pass filter is applied to them are also shown (the brown curve in Fig. 4.5a). This corresponds to DRAG being unaware of the filter. In Fig. 4.5a, the approximate best infidelity of the ansatz plus NN deep reinforcement learning approach is also marked. The optimized DRAG shown in Fig. 4.5a is the minimum infidelity of four random seeds initialized to [1,1,0,0] with 500 optimization steps each. Fig. 4.5 indicates that introducing a low-pass filter makes little difference to DRAG and that optimized DRAG might perform slightly better than the low-pass filtered NN plus ansatz DRL approach.



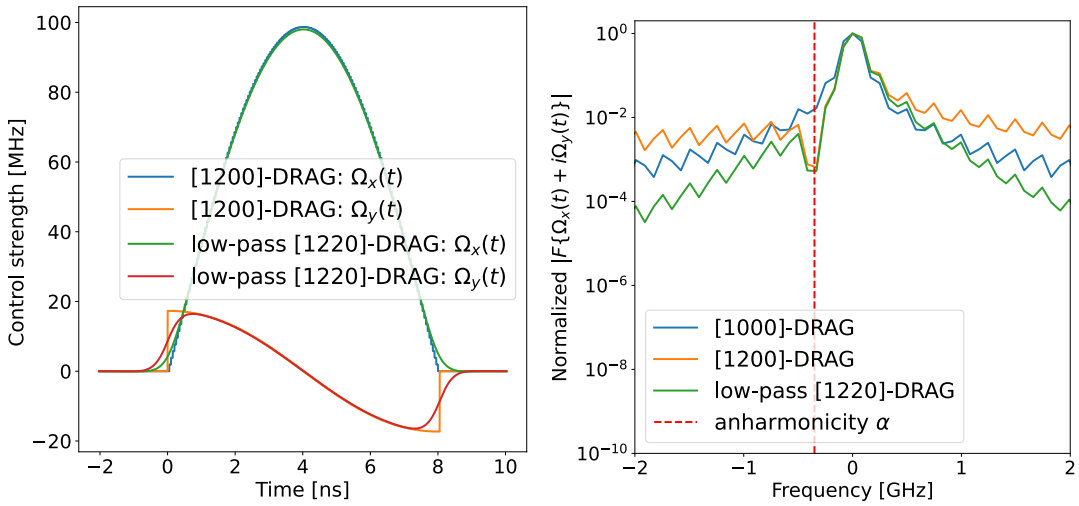
(a) Infidelity of DRAG **(b)** Optimal coefficents

Figure 4.5: Infidelity of DRAG coefficients and the optimized DRAG coefficients for different gate times t_g. Gaussian, derivative, detuning, and detuning offset are defined by $[\alpha, \beta, \gamma, \delta_0]$ in Eq. (3.28). For example, [1000]-DRAG corresponds to a pure Gaussian. The infidelity labeled min(Ansatz+NN) is $I = 6 \times 10^{-6}$, which is approximately the minimum when a low-pass filter is applied to ansatz plus NN deep reinforcement learning. For the brown curve in Fig. 4.5a DRAG is unaware of the low-pass filter.

It is somewhat surprising [1,1,1,0] (per Fig. 4.5a) is not optimal, despite the claim in [24] that [1,1,1,0] would be optimal. However, the numerically optimized DRAG coefficients shown in Fig. 4.5b align well with [21]. [21] indicates that $[1, 2, \gamma, \delta_0]$ is optimal if γ, δ_0 are optimized to minimize dephasing errors. In Fig. 4.5b, there are some fluctuations depending on t_g, but close to $[1, 2, 2, \delta_0]$ seems to be mostly optimal. Notably, the small difference between the numerically optimized DRAG coefficients and $[1, 2, 2, 0]$ in Fig. 4.5b seems to make a difference in Fig. 4.5a where the numerically optimized DRAG is almost two orders of magnitude lower in infidelity than [1,2,2,0]-DRAG. Fig. 4.5b also shows that numerically optimized DRAG seems to diverge from $[1, 2, 2, \delta_0]$ for $t_g > 16$ ns. This is difficult to explain, but not of much practical interest, as shorter gate times are preferable due to decoherence. That DRAG optimized with a state-of-the-art optimizer, here Nevergrad [46], performs on par with the ansatz plus NN deep reinforcement learning approach is not surprising, as DRAG has been analytically shown to perform well for single-qubit gates and is optimized with a state-of-the-art optimizer.

The shape of some different DRAG controls and the Fourier spectrum of $\Omega_x + i\Omega_y$ for $t_g = 8$ ns is given by Fig. 4.6. Fig. 4.6a shows the $t_g = 8$ ns controls in the time domain, but for the low-pass filter, some additional time has been added to model the finite rise time. As expected from [48], there is a local minimum in the frequency spectrum of $\Omega_x + i\Omega_y$ (Fig. 4.6b) at the anharmonicity α. This frequency corresponds to the energy of the $|1\rangle \leftrightarrow |2\rangle$ transition, meaning that leakage is suppressed. The system is expressed in a rotating frame; as such, the Fourier spectrum at zero, (the magnitude of the Fourier transform $F\{\Omega_x(t) + i\Omega_y(t)\}$ at zero, $|F\{\Omega_x(t) + i\Omega_y(t)\}|(0))$, corresponds to the $|0\rangle \leftrightarrow |1\rangle$ transistion. As expected, the peak in Fig. 4.6b is at zero, meaning that mostly the $|0\rangle \leftrightarrow |1\rangle$ transition is

driven.



(a) $\Omega_x(t)$ and $\Omega_y(t)$ in time-domain **(b)** Fourier spectrum of $\Omega_x(t) + i\Omega_y(t)$

Figure 4.6: Analysis of $\Omega_x(t), \Omega_y(t)$ for different DRAG coefficients $t_g = 8$ ns with no filter and a 750 MHz low-pass filter. Fig 4.6a shows the controls in the time-domian while Fig. 4.6b shows the Fourier transform of $\Omega_x + i\Omega_y$. The low-pass filter makes little difference as the original shape is already fairly smooth. Fig. 4.6b shows that the low-pass filter, as expected, suppresses high frequencies.

The Fourier spectrum of $\Omega_x + i\Omega_y$ does not explain the impact of detuning $\delta(t)$. For DRAG γ defines the detuning. To understand the infidelity difference between [1220]-DRAG and [1200]-DRAG, look at Fig. 4.7 showing the dynamics on the Bloch sphere for $t_g = 8$ ns. The Bloch sphere is a way to visualize a two-level system, here $|\psi(t)\rangle = C_0(t)\,|0\rangle + C_1(t)\,|1\rangle$, using spherical coordinates. Fig. 4.7 shows that [1200]-DRAG drifts to the side while [1220]-DRAG performs a seemingly perfect π-pulse. [1220]-DRAG is much more ideal than [1200]-DRAG, despite them having the same $\Omega_x(t) + i\Omega_y(t)$ frequency spectrum. Hence, the Fourier spectrum of $\Omega_x + i\Omega_y$ shown in Fig. 4.6b is not enough to predict infidelity. The takeaway is that it is not enough to ensure that $\Omega_x + i\Omega_y$ has a local minimum at the anharmonicity α, the detuning matters too. The details of this error, attributed to AC stark shift in [49], should be studied in more detail; there seems to be a drift in a specific direction rather than an oscillation. As such, it might be possible to compensate using a constant detuning, too, which is what is done in [49].

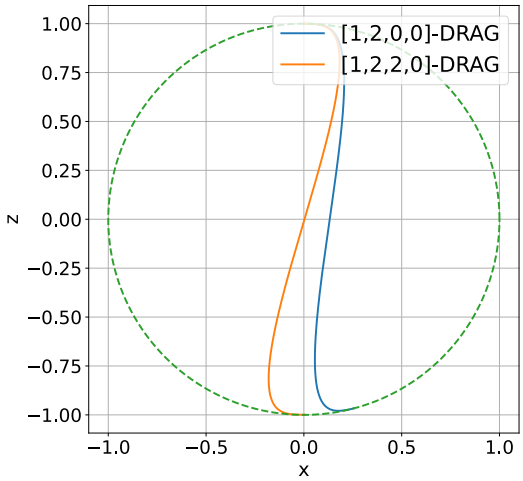


Figure 4.7: Projection of the time evolution of $|\Psi, t = 0\rangle = |0\rangle$ for [1220]-DRAG and [1200]-DRAG onto the XZ-plane of the Bloch sphere. Ideally, a π-pulse $\sigma_x |0\rangle = |1\rangle$, corresponding to $(x = 0, z = 1) \rightarrow (x = 0, z = -1)$. [1220]-DRAG is much closer to an ideal π-pulse than [1200]-DRAG.

4.1.3 Piece-wise Constant Controls

DRAG outputs a continuous control shape, hence something very different from the piece-wise constant control of the deep reinforcement method. Here, the piece-wise constant control results obtained by GRAPE and direct optimization of piece-wise controls with Nevergrad for several different gate times t_g are presented and discussed. The effect of the low-pass filter on the controls is also investigated, and the controls are compared with the deep reinforcement learning results for $t_g = 8$ ns.

Piece-wise constant controls optimized by Nevergrad (PWC NG), and GRAPE for different gate times t_g, with and without a low-pass filter, are shown in Fig. 4.8. All optimization methods are initialized from the single-qubit square ansatz. Note that PWC NG has a lower cost than deep reinforcement learning per optimization step due to a smaller batch size N. Fig 4.8, for example, shows that it can make a difference if the optimizer knows that a filter is applied. The evidence is that low-pass aware PWC NG consistently beats low-pass unaware PWC NG. A method being aware of the filter means that the filter is part of the optimization loop, while if it is unaware, the optimizer never sees the effect of the filter when optimizing. The implementation of GRAPE does not consider filters; as such, low-pass GRAPE is unaware of the filter. It should be possible to implement filter-aware GRAPE, see [29]. Notably, all methods pass the decoherence limit at similar gate times t_g, meaning they are similarly coherence-limited.

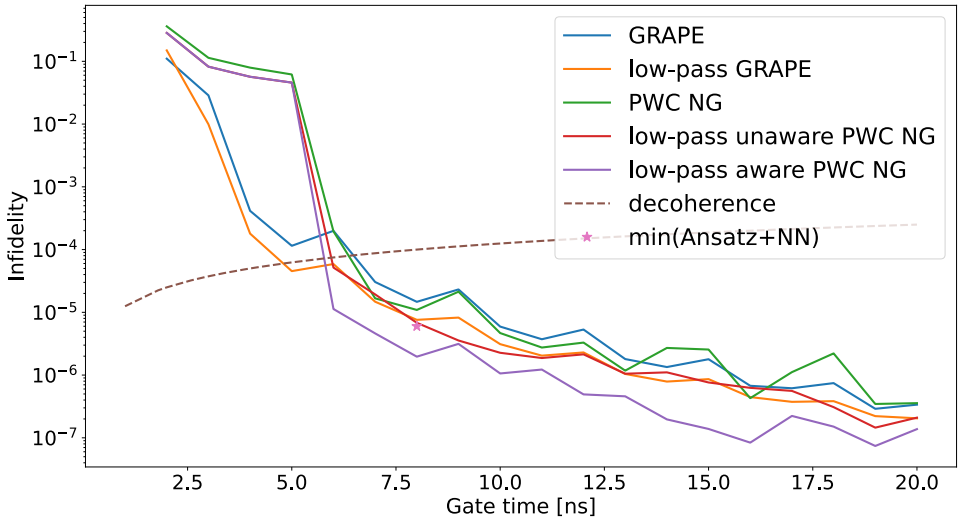


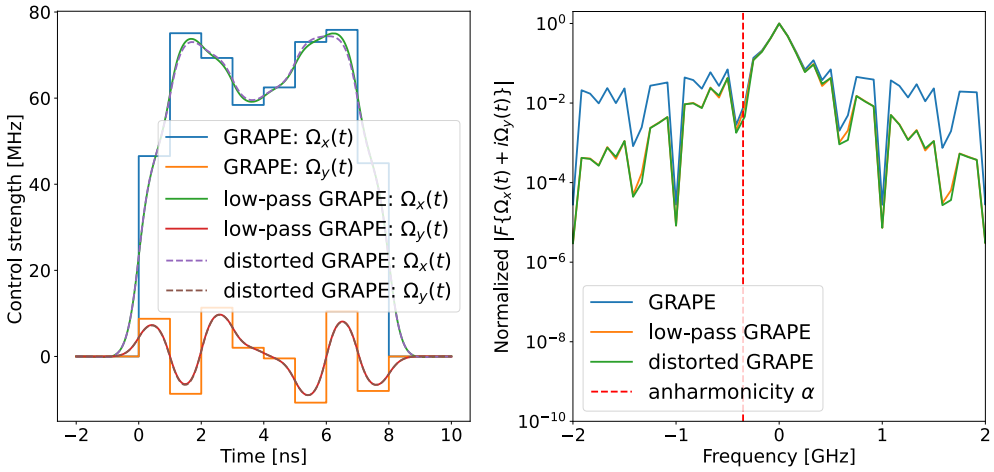
Figure 4.8: Figure showing the infidelity of the controls obtained from GRAPE and directly optimizing piece-wise constant controls with Nevergrad for different t_g. For the PWC NG method, the minima of five random seeds are shown. Both methods perform 500 optimization steps. The approximate minima $I = 6 \times 10^{-6}$ of the lowpass-filtered ansatz plus NN DRL approach is also shown. The decoherence marked is for $T_1 = 80$ µs, $T_1 = 40$ µs. GRAPE is unaware of the low-pass filter when optimizing.

It should be noted that black-box optimization with Nevergrad might scale badly to two-qubit gates, as two-qubit gates likely need many more parameters. How bad is a subject for future studies. Despite this risk, Fig. 4.8 demonstrates that directly optimizing the quantum gate controls with a black-box optimizer is an interesting alternative to deep reinforcement learning, as deep reinforcement learning does not seem to have an advantage in infidelity. For $t_g = 8$ ns optimization with Nevergrad even yields a slightly lower infidelity. Nevergrad also yields low fidelities consistently for many different t_g, even though it does not work so well for very short t_g. Additionally, Nevergrad is considerably faster than deep reinforcement learning per optimization step as the batch size N is much smaller.

Fig. 4.8 mostly supports the claim from the deep reinforcement learning that applying a low-pass filter to a piece-wise constant control lowers the infidelity, seemingly regardless of whether or not the optimization process takes the filter into account. There are two cases where piece-wise constant controls optimized with Nevergrad without the filter beat PWC Nevergrad (NG) unaware of the filter, $t_g = 7$ ns and $t_g = 16$ ns. Reasonably, the lowest infidelities are obtained when the optimizer is aware of the filter. The low-pass filter seems to decrease rather than increase the infidelity of piece-wise constant controls. This is significant, as that means piece-wise constant parametrized controls might work well in experiments where limited bandwidth distortions can be expected.

Fig. 4.9 exemplifies the effect of the low-pass filter on the optimized GRAPE pulse. In the figure, the piece-wise Gaussian distorted GRAPE pulse is also marked. See Eq. (3.7) in the methods section for details about the Gaussian distortion. Fig. 4.9b shows a local minimum in the frequency spectrum of $\Omega_x + i\Omega_y$ at the anharmonicity

α, like for DRAG shown in Fig. 4.6b. Fig. 4.9a shows that the Gaussian distortion is barely noticeable and that the low-pass filter has a significant smoothing effect on piece-wise constant controls.



(a) Ω_x and Ω_y in time domain **(b)** Fourier transform $\Omega_x(t) + i\Omega_y(t)$

Figure 4.9: Analysis of the GRAPE optimized $t_g = 8$ ns π-pulse controls. 500 optimization steps were performed. Fig. 4.9a shows the GRAPE-optimized controls $\Omega_x(t), \Omega_y(t)$ with no filter, a 750 MHz low-pass filter, and a Gaussian distorted low-pass filter. Fig. 4.9b shows the Fourier transform of $\Omega_x(t) + i\Omega_y(t)$. Like in the DRAG case, there is a local minimum at the anharmonicity α, suppressing leakage.

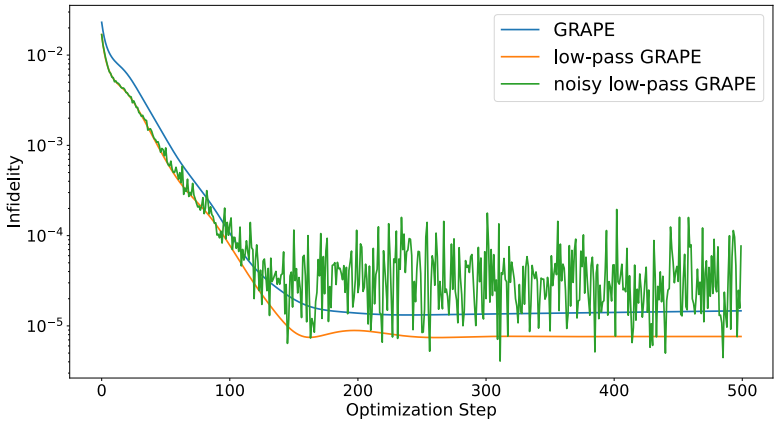
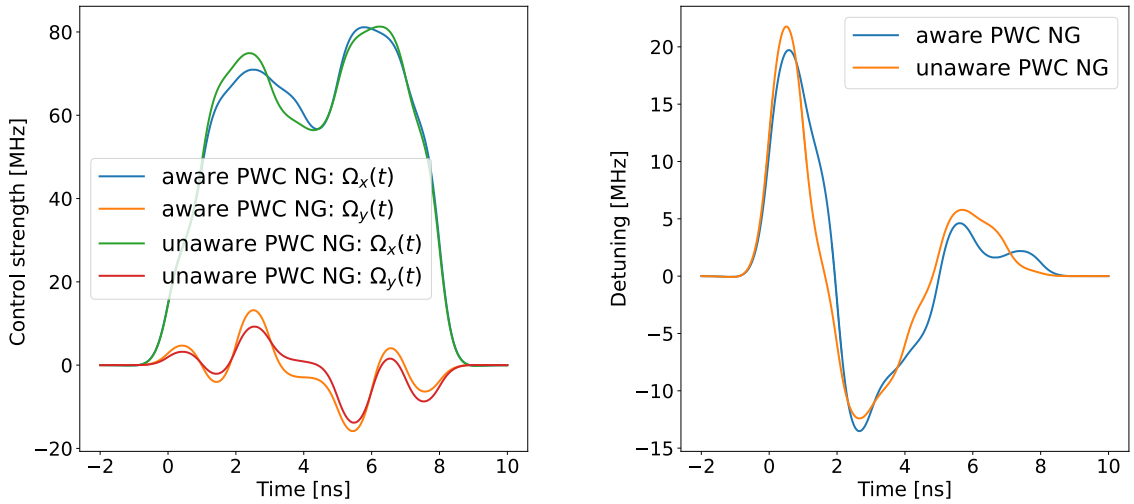


Figure 4.10: Infidelity of GRAPE optimized π-pulse with $t_g = 8$ ns when there is no filter, a low-pass filter, and a small Gaussian distortion of the low-pass filtered signal. The low-pass GRAPE curve being below the GRAPE curve shows that the low-pass filter can consistently lower the infidelity, and small distortions can be very detrimental to the infidelity.

Fig. 4.10 shows the corresponding optimization process to reach the control pulses

shown in Fig. 4.9. Like Fig. 4.8, Fig. 4.10 shows that the low-pass filter lowers the infidelity notably, even if not considered when optimizing. This demonstrates that piece-wise constant pulses likely can be improved by smoothing them with a filter. This is not so surprising, as the low-pass filter dampens high-frequency components, components that could cause leakage to higher energy levels. Fig. 4.10 also shows that a small pulse distortion (barely noticeable in Fig. 4.9a) can have a large impact on the infidelity, stressing the need to be robust against pulse distortions and/or consider distortions when optimizing the pulse.

What is the difference between the aware and the low-pass filter unaware piece-wise constant controls directly optimized using Nevergrad? Fig. 4.11 shows the piece-wise constant $t_g = 8$ ns controls optimized with Nevergrad when the optimizer is aware and unaware of the filter. The figure shows little qualitative difference between being aware and unaware of the filter. There is perhaps a tendency that the aware case exhibits more features than the unaware case, for example, the oscillation between 6 and 8 ns in the detuning $\delta(t)$ shown in Fig. 4.11b.



(a) Ω_x and Ω_y controls

(b) Detuning $\delta(t)$

Figure 4.11: Comparison of the piece-wise constant controls optimized by Nevergrad when aware and unaware of the low-pass filter when optimizing. The controls $\Omega_x(t), \Omega_y(t), \delta(t)$ for the aware and unaware case seem qualitatively similar. The detuning seems qualitatively different from what DRAG suggests.

Interestingly, the detuning $\delta(t)$ of the piece-wise constant controls optimized by NG shown in Fig. 4.11b does not seem well described by the detuning shape suggested by DRAG, $\delta_{\text{DRAG}}(t) = \gamma \frac{\Omega_x^2}{4\alpha} + \delta_0$ as there is many local extremums. This indicates that piece-wise constant controls optimized by Nevergrad are qualitatively different from DRAG.

4.1.4 Overall Comparison of Bloch Sphere Dynamics

Many methods can yield high-fidelity single-qubit gates. Here, I visualize the dynamics of the obtained gates from the different methods. Fig. 4.12 visualizes the

dynamics of some single-qubit methods and the square-shaped ansatz. The dynamics is shown on a Bloch sphere with $t_g = 8$ ns. The figure shows rather significant differences in the way the Bloch sphere is traversed. DRAG follows an S-shaped trajectory. DRAG controls are symmetric around $t = t_g/2$; this is evident in the dynamics too. The piece-wise constant methods (GRAPE and directly optimizing piece-wise constant controls with Nevergrad) stick much closer to the $x = 0$ line. The $x = 0$ line corresponds to a pure rotation around the x-axis. The smoothing effect of the low-pass filter is also visible in the dynamics, with the normal piece-wise constant GRAPE showing larger fluctuations than the low-pass GRAPE. Intuitively, smooth traversal of the phase space seems easier to control.

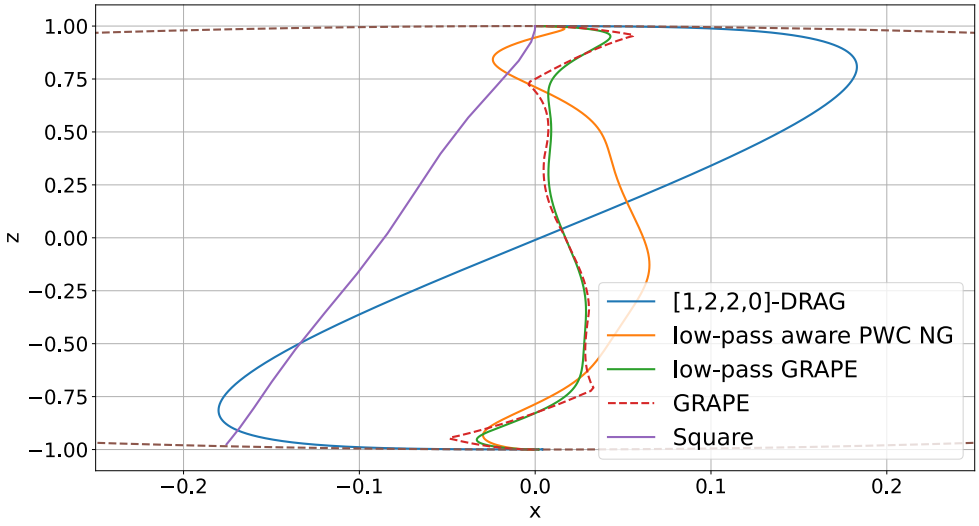


Figure 4.12: Comparison of π-pulse dynamics from different methods visualized on the Bloch sphere. The time evolution of $|\psi(t)\rangle = |0\rangle$ for the different methods is shown on the XZ-plane projection of the Bloch sphere. The optimization task was to realize a $t_g = 8$ ns π-pulse. Ideally, a π-pulse $\sigma_x |0\rangle = |1\rangle$ corresponding to $(x = 0, z = 1) \rightarrow (x = 0, z = -1)$. Square is the square-shaped ansatz. GRAPE and directly optimizing low-pass filtered piece-wise constant controls with Nevergrad (low-pass aware NG) seems to stay closer to $x = 0$ than DRAG.

It is not so surprising that qualitatively different ways to implement high-fidelity single-qubit gates exist, as DRAG does not rule out alternatives to Ω_x being Gaussian. Using another valid choice of Ω_x or a more advanced version of DRAG, such as in [28], might yield DRAG dynamics closer to $x = 0$. It is still noteworthy that numerically optimized DRAG and low-pass aware Nevergrad yield similar high-fidelity single-qubit gates despite being qualitatively different in the dynamics.

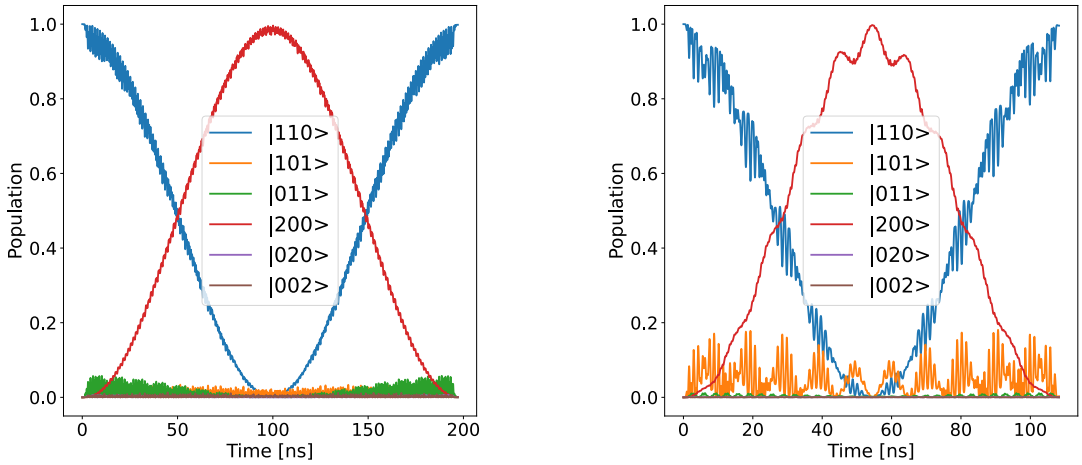
4.2 Two-qubit Results

Here, the results related to the two-qubit set-up with the aim of optimizing a CZ gate are presented. Two-qubit systems are much slower than single-qubit systems

to simulate, a practical problem when investigating quantum gates with a classical computer. In addition to analyzing two-qubit ansatzes optimized with Nevergrad, I investigate whether the ansatz plus NN deep reinforcement learning approach is able to increase the fidelity of a CZ gate compared to using just an optimized ansatz. Moreover, the dynamics of the optimized CZ-gate ansatzes are investigated in detail. Here, the low-pass filter is not considered.

4.2.1 Optimized Ansatzes

When optimizing the two-qubit ansatz (Eq. (3.19)), many different solutions can be obtained depending on how the optimization procedure is initialized. A short gate time t_g is desired for decoherence reasons, but short gate times require larger amplitudes, activating other transitions due to a broader frequency spectrum. The aim is to find solutions with a favorable intensity-to-bandwidth ratio. The solutions can be used as ansatzes for deep reinforcement learning.



(a) 197.1 ns CZ-gate

(b) 108.3 ns CZ-gate

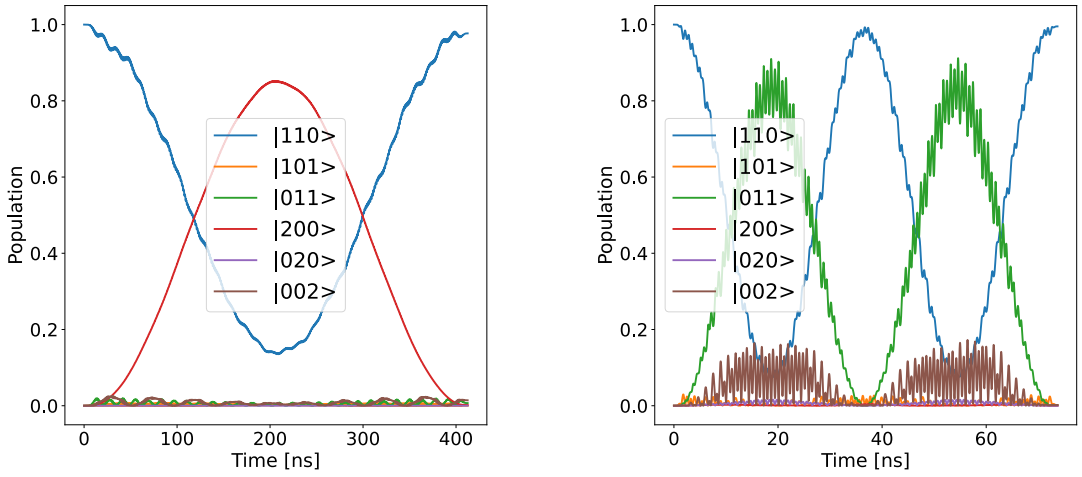
Figure 4.13: Comparison of population $|C_{|i,j,k\rangle}(t)|^2$ when the initial state $|\Psi_0\rangle = |110\rangle$ ($C_{110}(t) = 1$) for a 108.3 ns CZ gate and slower 197.1 ns CZ gate. The slower gate mainly activates the $|110\rangle \Leftrightarrow |200\rangle$ transition, while for the fast gate, $|101\rangle$ is populated.

Fig. 4.13 shows the population dynamics $|C_{|i,j,k\rangle}(t)|^2$ when starting in $|110\rangle$ for two different optimized ansatzes. Fig. 4.13a shows a 197.1 ns gate where mainly the $|110\rangle \leftrightarrow |200\rangle$ transition is active, while Fig. 4.13b shows a 108.3 ns gate activating $|101\rangle$ in addition to the $|110\rangle \leftrightarrow |200\rangle$ transition. The ideal case is $|C_{|1,1,0\rangle}(t_g)|^2 = 1$ while all other populations are zero at $t = t_g$. The 197.1 ns gate has infidelity $I = 1 - F \approx 0.8 \times 10^{-3}$, but the fidelity reduction due to decoherence is $I_c \approx 7.6 \times 10^{-3}$. The faster 108.3 ns gate has infidelity $I \approx 1.9 \times 10^{-3}$ with fidelity reduction $I_c \approx 4.2 \times 10^{-3}$. This illustrates that whether or not a fast gate is preferable depends on the decoherence model. Here $T_1 = 80$ µs, $T_2 = 40$ µs and the decoherence

infidelity I_c is modeled by Eq. (2.24). The complete description of the optimized 108.3 ns and 197.1 ns optimized ansatzes is given in Table C.1.

One can perform CZ gates in many different ways. I have focused on $|110\rangle \leftrightarrow |200\rangle$ CZ gates, but one could also devise CZ gates that, for example, excite the coupler. The coupler can be excited by the ansatz cosinus frequency ω_c being far from $|E_{110} - E_{200}|$. Fig. 4.14b shows a 73.7 ns gate where mainly the $|110\rangle \leftrightarrow |011\rangle$ transistion is active. It is a valid CZ gate as far as the simulation is concerned, as the realized unitary time-evolution operator $\hat{U}(t_g)$ is close to the target operator $\hat{U}_T$ of a CZ gate. Fig. 4.14a shows a very slow, significantly detuned CZ-gate. The gate is detuned because the minimum population of $|110\rangle$ is far from zero. The parameters of the detuned and coupler exciting gate are given in Table C.2. The simulated infidelity of the detuned gate $I \approx 6.9 \times 10^{-3}$ with decoherence infidelity $I_c \approx 16 \times 10^{-3}$.

The simulated infidelity of the coupler exciting gate $I \approx 2.3 \times 10^{-3}$ with $I_c \approx 2.9 \times 10^{-3}$. This corresponds to a total infidelity $I_{\text{total}} = I + I_c$ lower than the optimized $|110\rangle \leftrightarrow |200\rangle$ CZ gates. However, a gate exciting the coupler should be more sensitive to noise in the flux signal $\Phi(t)$ reaching the coupler, something that I do not model with I_c. A reasonable assumption based on the experimental set-up at Chalmers is that the coupler has $(T_1, T_2) = (25, 1)$ µs which is much worse than the coherence times of the qubits modeled to be $(T_1, T_2) = (80, 40)$ µs. How this impacts the gate decoherence is a topic for future research, but the decoherence infidelity $I_c \approx 2.9 \times 10^{-3}$ is most likely an underestimate for the 73.7 ns coupler exciting gate's decoherence infidelity.



(a) 412.5 ns CZ-gate **(b)** 73.7 ns CZ-gate

Figure 4.14: Comparison of population $|C_{|i,j,k\rangle}(t)|^2$ dynamics when the initial state $|\Psi_0\rangle = |110\rangle$ ($C_{110}(t) = 1$) for two optimized CZ gates. The 412.5 ns gate is very detuned but almost only activates $|110\rangle \Leftrightarrow |200\rangle$ while the 73.7 ns gate does not mainly use the $|110\rangle \Leftrightarrow |200\rangle$ transition, instead primarily exciting the coupler.

For the optimized ansatzes included in this thesis $0.35 \leq \theta \leq \max(|\Phi(t)|) \leq \theta +$

$\delta_c + \delta_s < 0.45 < 0.5$ (unit defined by $\Phi_0 = 1$). Quantifying what effect $\max(|\Phi(t)|) = 0.35$ relative to $\max(|\Phi(t)|) = 0.45$ has on the coherence times T_1, T_2 is a subject for future studies. It should have an effect as a larger $\max(|\Phi(t)|)$ implies more sensitivity to flux noise.

4.2.2 Combining an Ansatz with Deep Reinforcement Learning

Fig. 4.15 shows the infidelity when applying deep reinforcement learning to the optimized 108.3 ns ansatz. Fig. 4.15 indicates that ansatz plus NN deep reinforcement learning is able to decrease the infidelity relative to the ansatz for the 108.3 ns CZ gate. Similar results can be obtained for other ansatzes. The study has focused on ansatz plus NN for fast gates like the 108.3 ns gate because decoherence limits the infidelity for slower gates.

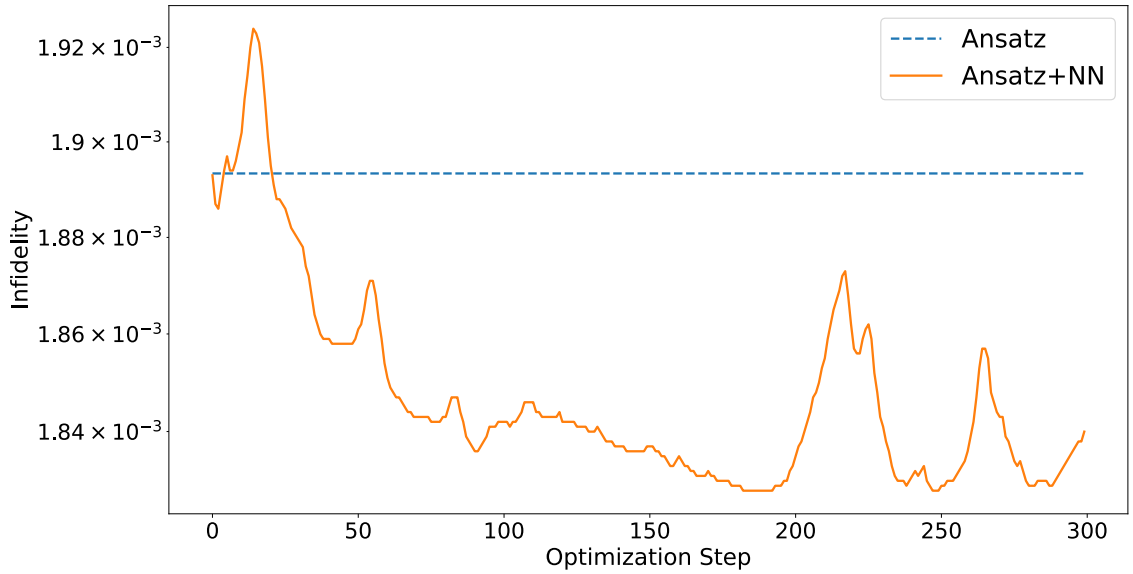


Figure 4.15: Infidelity of CZ gate when applying deep reinforcement learning to the optimized 108.3 ns ansatz. Deep reinforcement learning decreases the infidelity a little bit relative to the ansatz. Decoherence is not considered in the figure. The number of neural network updates is synonymous with the number of optimization steps.

Fig. 4.15 shows that using ansatz plus NN deep reinforcement learning, it is possible to improve the infidelity of the CZ-gate relative to just using an optimized ansatz. Unfortunately, the improvement is not significant; it could be due to the ansatz already being very optimized or not using the most optimal deep reinforcement learning hyperparameters. There might be a way to integrate deep reinforcement learning and the ansatz that will lead to a larger improvement.

4.2.3 CZ Gate Dynamics

So far, CZ gate has been mainly analyzed numerically. How can the dynamics of the CZ gate be interpreted? Ref. [18, 47] uses that CZ gates implemented through performing a $|110\rangle \leftrightarrow |200\rangle$ transition, can be modeled with

$$\hat{H} = g_{\text{eff}}(|200\rangle \langle 110| + |110\rangle \langle 200|), \tag{4.2}$$

where g_{eff} is the effective coupling strength. The derivation of Eq. (4.2) makes multiple assumptions, for example, that the transition is driven at resonance and that the coupler is not excited. [18] uses the model (Eq. (4.2)) to derive the decoherence infidelity model (Eq. (2.24)) used for the CZ gate. With initial state $|\psi(0)\rangle = |110\rangle$ ($C_{110}(0) = 1$) Eq. 4.2 implies

$$C_{110}(t) = \cos(g_{\text{eff}}t). \tag{4.3}$$

This $C_{110}(t)$ model does not seem to always explain the dynamics for optimized ansatzes simulated with CSQR, as they, for example, exhibit signs of detuning. An example with significant detuning is the 412.5 ns CZ-gate shown in Fig. 4.14a. Detuning means that there is never a full population transfer, meaning that

$$\min_t |C_{110}(t)|^2 > 0 \tag{4.4}$$

Detuning is not captured when assuming the transition is driven at resonance.

A well-known model that captures detuning is the Rabi model,

$$\hat{H} = \frac{\hbar\omega_0}{2}\sigma_z + \hbar A \cos(\omega t)\sigma_x. \tag{4.5}$$

The solution for a two-level system starting in the ground state is

$$C_g(t) = e^{-i\frac{\Delta}{2}t}\left(\cos\left(\frac{\Omega_R t}{2}\right) + i\frac{\Delta}{\Omega_R}\sin\left(\frac{\Omega_R t}{2}\right)\right). \tag{4.6}$$

where $\Delta = \omega_0 - \omega$, $\Omega_R = \sqrt{A^2 + \Delta^2}$. For the two-qubit system parameters used $E_{110} < E_{200}$. Starting with $C_{110}(0) = 1$, assuming only the $|110\rangle \leftrightarrow |200\rangle$ transition is active, reduces to a two-level system starting in the ground state.

For time-independent Hamiltonians, the dynamics of an eigenstate is given by $C_i(t) = e^{-iEt}$. This means $|110\rangle$ could gain an extra relative phase even if the system is in an eigenstate, $C_{110}(t) = e^{-i(E_{110}-E_{100}-E_{010}+E_{000})t}$, where $\hat{H}|ijk\rangle = E_{ijk}|ijk\rangle$. Often this is interpreted as there being an extra σ_z term in the hamiltonian with coefficient $\zeta_{ZZ} = (E_{110} - E_{100} - E_{010} + E_{000})$. This observation is known as ZZ coupling, see [50] for a detailed investigation. In general, E_{ijk} and therefore ζ_{ZZ} could be time dependent, that would make

$$C_{110}(t) = e^{-i\int_{t_0}^{t}\zeta_{ZZ}(t')dt'}. \tag{4.7}$$

The Rabi model and ZZ coupling motivate fitting

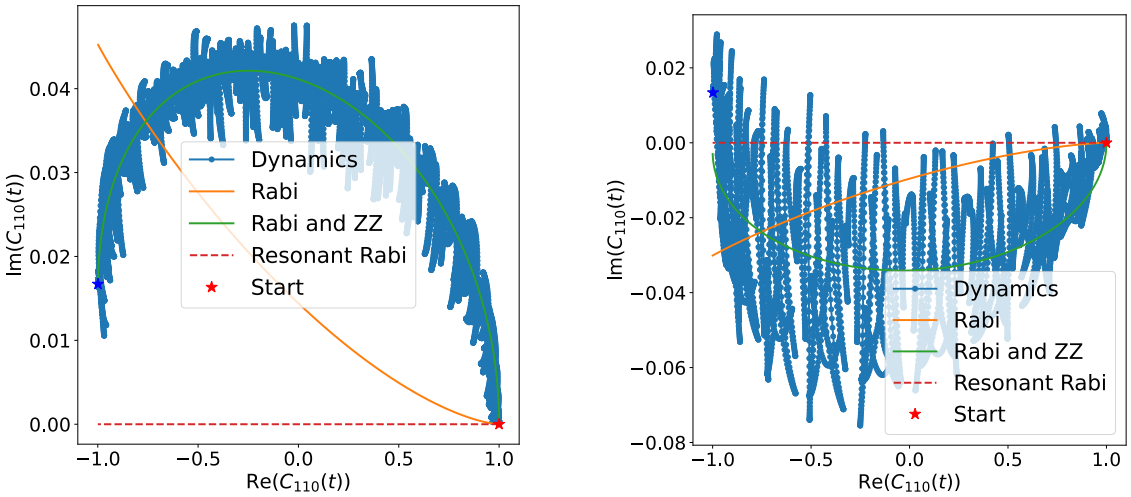
$$C_{110}(t) = e^{-i\left(\zeta_{ZZ}+\frac{\Delta}{2}\right)t}\left(\cos\left(\frac{\Omega_R t}{2}\right) + i\frac{\Delta}{\Omega_R}\sin\left(\frac{\Omega_R t}{2}\right)\right) \tag{4.8}$$

to the data obtained from simulating the full Hamiltonian model of the system. The model Eq. (4.8) assumes a constant ZZ coupling in order not to overfit to the Hamiltonian data. The $C_{110}(t)$ model Eq. (4.8 is fit to data where the phase has been calibrated to be the phase gained by $|110\rangle$ relative to $|100\rangle$, $|010\rangle$, and $|000\rangle$. The model reduces to the commonly used Eq. (4.2) if $\Delta = 0, \zeta_{ZZ} = 0$, corresponding to a resonant Rabi oscillation. The constants are initialized $A_0 = \frac{\pi}{2t_g}$, where t_g is the gate time, $\Delta_0 = 0, \zeta_{ZZ_0} = 0$, in order to initialize $\cos\left(\frac{\Omega_R t_g}{2}\right) = -1$. The loss function for the fit is the root mean square,

$$\text{Loss} = \sqrt{\frac{1}{N}\sum_{i=1}^{N} |C_{110}^{\text{fit}}(t_i) - C_{110}^{\text{simulation}}(t_i)|^2}, \tag{4.9}$$

where $C_{110}^{\text{fit}}(t)$ is given by Eq. (4.8), $C_{110}^{\text{simulation}}(t)$ is obtained from the two-qubit Hamiltonian, and the optimization method is Nelder-Mead.

Fitting A, Δ, ζ_{ZZ} for the 197.1 ns and 108.3 ns ansatzes yields Fig. 4.16. In the figure, the fit when $\zeta_{ZZ} = 0$ corresponding to a Rabi oscillation is marked too. Fig. 4.16 shows that a detuned Rabi oscillation with a constant ZZ coupling much better describes the dynamics of the optimized CZ-gate than the resonant Rabi model. The ZZ coupling is key to a good fit; using just the Rabi model does not work well. Notably, the imaginary and real axis scales in Fig. 4.16 are different; a resonant Rabi oscillation corresponding to Eq. (4.2) is a decent description of the dynamics, but not as good as a Rabi model with ZZ coupling.



(a) 197.1 ns CZ-gate

(b) 108.3 ns CZ-gate

Figure 4.16: Fits of different models to the $C_{110}(t)$ dynamics obtained from the two-qubit system's Hamiltonian. The phase of $C_{110}(t)$ is the relative phase gained by $|110\rangle$. The red and blue stars mark the start $t = 0$ and end $t = t_g$.

There is likely something additional than a Rabi oscillation with ZZ coupling going on in Fig. 4.16, as there are significant oscillations, especially for the 108.3 ns gate. A possible explanation could be other transitions than $|110\rangle \leftrightarrow |200\rangle$ being active, something that is plausible given other states' populations being active in Fig. 4.13.

Another factor could be time-dependent ZZ coupling. The Rabi oscillation and ZZ coupling do not consider the rise and fall time, which should make the start and end dynamics difficult to capture correctly.

The $C_{110}(t)$ dynamics for 412.5 ns gate and 73.7 ns gate is shown in Fig. C.2, supporting the conclusion that Rabi oscillation ZZ coupling works works well for $|110\rangle \leftrightarrow |200\rangle$ CZ-gates while the coupler exciting 73.7 ns gate is a very different kind of gate.

The detuning and ZZ coupling understanding could be useful for future studies. Considering ZZ coupling could be helpful to understand what θ to use as ZZ coupling varies in the energy spectrum. ZZ coupling might also be important when designing the device parameters, for example, the coupling strengths g_{1c} and g_{2c} (defined in Eq. (2.45)). In this study, the two-qubit model parameters were fixed at the values shown in Table B.1.

4.2.4 Robustness

There seems to be considerable variations in robustness between different two-qubit ansatzes. It seems like perturbations of the offset θ have the most considerable impact on the infidelity of the two-qubit ansatz. Fig. 4.17 compares the infidelity of the 197.1 ns and the 108.3 ns ansatzes when θ is perturbed. Fig. 4.17 indicates that the 197.1 ns pulse is much more robust than the 108.3 ns pulse, as the infidelity when θ is perturbed is increased a lot more for the 108.3 ns ansatz than the 197.1 ns ansatz. It makes sense that the 108.3 ns gate is more sensitive, as the $C_{110}(t)$ dynamics of the 108.3 ns gate shown in Fig. 4.16 exhibit larger oscillations than the dynamics of the 197.1 ns gate. The lack of robustness in the 108.3 ns ansatz could be a factor in why deep reinforcement learning was unable to lower the infidelity significantly. A more in-depth study of robustness, including modeling the noise in experimental hardware, is needed.

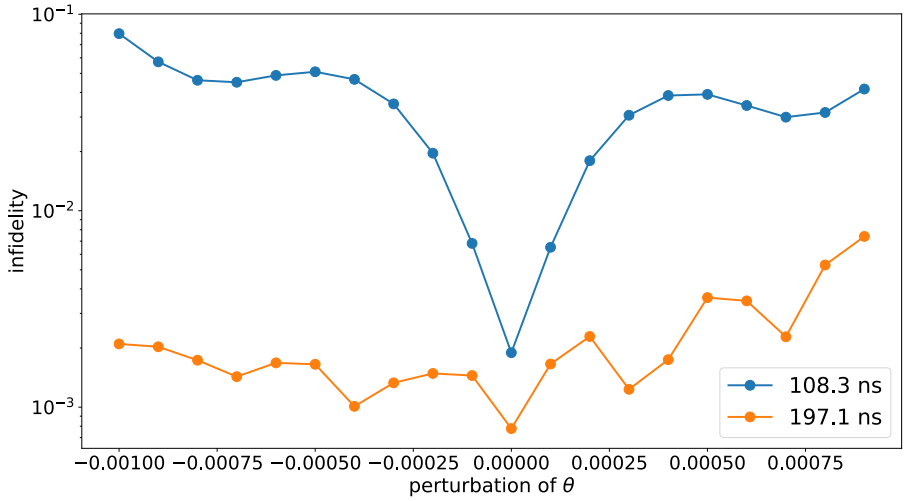


Figure 4.17: The infidelity (without decoherence infidelity) of the 197.1 ns gate and the 108.3 ns CZ gates when the two-qubit constant offset θ is modified. Both are of the $|110\rangle \leftrightarrow |200\rangle$ type of CZ gates. There seems to be an order of magnitude difference in sensitivity to perturbations in θ.

The magnitude of the oscillations seen in Fig. 4.16 could be connected with the robustness shown in Fig. 4.17. The ansatz that minimizes infidelity when there is noise might be qualitatively different from the ansatz that minimizes infidelity when there is no noise. If so, noise and robustness should be considered when optimizing the two-qubit ansatz; noise cannot be an afterthought. This makes sense, if the Earth is shaking, a bowl is a much better way to carry things around than a plate. Robustness to noise might, hence, imply qualitatively different control shapes. This line of reasoning is in line with [51].

5
Discussion

More things than the single-qubit infidelities shown in the results section should be considered when comparing different quantum gate optimization methods. Here, for example, the degree of dependency on having a theoretical model and the time it takes to do an optimization step are discussed. Some takeaways are possible from just the infidelities, for example, that using a suitable ansatz can benefit deep reinforcement learning for single-qubit gates.

Many methods can optimize single-qubit gates well. The detailed analysis of the $t_g = 8$ ns case (shown in Fig. 4.1) seems to indicate that GRAPE is preferable as it converges fast to a reliably low value, while DRAG performs well for some seeds but not all. Deep reinforcement learning seems more reliable than DRAG, which only reaches low infidelities for some seeds. Looking at many different t_g shown in Fig. 4.5a and Fig. 4.8, it seems like DRAG optimized with Nevergrad and directly optimizing piece-wise constant controls with Nevergrad when a low-pass filter is considered, performs similarly well. It should be said that GRAPE and DRL do not seem significantly worse than DRAG and piece-wise constant controls optimized with Nevergrad when looking at many different t_g in terms of infidelity. DRAG and GRAPE might be advantageous for gate times $t_g < 5$ ns.

An aspect that is hidden when looking at just the number of optimization steps is the cost per optimization step. It is crucial to note that the deep reinforcement learning approaches use a batch size of 50 for the single-qubit case. For each optimization step, the system is probed with 50 different controls. This means that, compared to the two methods optimized with Nevergrad, each optimization step is much more expensive, as the Nevergrad methods only use the infidelity of one control signal for each optimization step. With the batch size used here, deep reinforcement learning would be about 50 times slower per optimization step to implement in an experiment. However, my deep reinforcement learning implementation could likely be massively enhanced, as it uses much less sophisticated techniques than Nevergrad. A way would be to use a more advanced DRL algorithm, like, for example, proximal policy optimization [33]. One could perhaps decrease the batch size N as low as $N = 5$ using more advanced deep reinforcement learning techniques than the ones used here.

It is important to consider how much an optimization method depends on having an accurate theoretical model of quantum hardware. In this thesis, the theoretical model used for the model-based approaches is similar to the model used for simulations. The low-pass filter is an example of something not considered by the theoretical model but included in the simulations. If substantial differences exist between the theoretical model and the actual hardware, model-based methods should

perform worse. GRAPE requires a complete Hamiltonian model for the dynamics and, as such, might be difficult to implement experimentally. In this study, the model given to GRAPE is similar to the model used to simulate the dynamics. If there is a significant difference between the model used to calculate the GRAPE gradient and the model of the actual system, worse performance should be expected. DRAG is also model-dependent, as the derivation of the control shapes is based on a theoretical model. The DRAG coefficients can, however, be optimized numerically based on the infidelities obtained from real hardware. Deep reinforcement learning and directly optimizing piece-wise constant control with Nevergrad are themself model-free, even though using an ansatz introduces some model dependency.

It is not so surprising that Nevergrad, a state-of-the-art optimizer, seems to perform better (at least in terms of batch size) than the deep reinforcement learning approach I have implemented from scratch. A fairer comparison might be comparing my deep reinforcement learning approach with a simpler black-box optimizer. It is also expected that black-box optimization can address gate optimization, as it has been demonstrated that it can generate results competitive relative to state-of-the-art reinforcement learning for challenging problems [52]. Based on this study, it seems that a general-purpose state-of-the-art black-box optimizer might be preferred for quantum gate optimization instead of implementing a tailor-made deep-reinforcement learning model, at least without significant implementation effort. Other machine learning paradigms than deep reinforcement learning could be more competitive relative to a state-of-the-art black-box optimizer baseline. An alternative to deep reinforcement learning is model learning, investigated for example in [20].

It is not certain that methods performing well for single-qubit gates will perform well for two-qubit gates. DRAG is explicitly designed for single-qubit gates and, as such, does not apply to the two-qubit case. My implementation of GRAPE needs to be adopted to handle the calibration of $\hat{U}(t_g)$ done in the two-qubit case. This thesis indicates that two-qubit gates, as expected based on the literature, need longer gate times than single-qubit gates. This study has not addressed how well directly optimizing piece-wise constant controls with Nevergrad will scale to very long gates. The thesis finds that two-qubit gates are a much more difficult problem than single-qubit gates and recommends future research to focus on the topic. There are already many different methods to optimize single-qubit gates that perform well. The fact that this thesis is unable to find a significant improvement by applying deep reinforcement learning to an optimized two-qubit ansatz might indicate that a high-dimensional control signal (for example the PWC signal generated by DRL Eq. (3.20)) maybe does not have a significant advantage over a low-dimensional signal (for instance the two-qubit ansatz Eq. (3.19)).

6
Conclusion

I have compared many different quantum gate optimization techniques on simulated hardware similar to what is used at Chalmers. I have also interpreted and compared the control shapes that were obtained. At least for the simulated hardware and the way techniques were implemented, the conclusions presented in this chapter seem reasonable. A different implementation could yield slightly different conclusions, for example, stronger deep reinforcement learning results. The number of random seeds has also been limited.

Combining deep reinforcement learning with an ansatz seems beneficial. For the single-qubit π-pulse, the number of optimization steps needed to reach the coherence limit is decreased from about 250-350 to about 100-200 by using a simple ansatz. This means the simulation time required is approximately halved. The algorithm should also scale linearly with the number of optimization steps in experiments. The pre-training approach suggested by [7] and adding the ansatz directly to the output of the deep reinforcement learning neural network performs similarly. Note that in this study, just one simple single-qubit ansatz is considered. For two-qubit gates, it is possible to decrease the infidelity of an optimized ansatz by applying deep reinforcement learning, but I am only able to show a minor decrease. That could be because the ansatz is already very optimized, but other constraints might exist, such as robustness. Modifying the implementation might yield a larger improvement.

Feeding the reinforcement learning agent information s_i about the quantum state of the system, as suggested by [10, 9], does not seem to make a considerable difference. It might even have a detrimental effect. Hence, not including information about the quantum state of the system, similar to the approach suggested by [7] and used in [8], might be preferable as it is easier to implement experimentally. I do not rule out that for a different DRL implementation, including information about the quantum state could make a difference.

I have little evidence that my deep reinforcement learning implementation would be the best approach to optimizing single-qubit quantum gates, even if including an ansatz. Optimizing DRAG or a piece-wise constant pulse directly using Nevergrad seems to yield at least as good results, despite them being easier to implement experimentally due to DRL using a larger batch size. Notably, DRAG is very model-dependent, meaning that DRAG might perform significantly worse in experiments where distortions of control signals are to be expected. From this study, the most practical, high-performing, least model-dependent method to optimize single-qubit gates seems to be directly optimizing low-pass-filtered piece-wise constant controls with Nevergrad, a state-of-the-art, general-purpose, black-box optimizer. Most of the considered methods seem to yield decoherence-limited single-qubit gates.

Furthermore, the thesis finds indications based on single-qubit gates that experimental devices' limited bandwidth, in this thesis modeled with a low-pass filter, seems not to hurt the performance of piece-wise constant controls. On the contrary, low-pass filtered piece-wise constant controls seem to perform better than piece-wise constant controls without a filter. This indicates that piece-wise constant controls are experimentally relevant. A piece-wise constant control cannot be generated experimentally because control electronics have a finite bandwidth. Still, the performance of piece-wise constant controls considering the limited bandwidth distortion seems better, not worse. Piece-wise constant controls are common in other studies, for example [9, 10], often without considering the experimental implementation.

The thesis highlights that optimizing two-qubit gates is a more challenging problem than single-qubit gates and is less well understood. The thesis uses Nevergrad to optimize an ansatz based on the energy spectrum of the two-qubit set-up, finding many different possible CZ-gates. Which of the optimized CZ-gates is the best likely depends on how robust the gate needs to be and how decoherence is modeled. Simulating two-qubit gates is time-consuming.

The investigation of two-qubit gates has highlighted that the Rabi model combined with ZZ coupling describes CZ gates based on the $|110\rangle \leftrightarrow |200\rangle$ transition well. Sometimes [47, 18], an even simpler model is used to model CZ gates; that model does not seem to describe the dynamics of $|110\rangle \leftrightarrow |200\rangle$ based CZ gates as well.

6.1 Outlook

Throughout the thesis, I have highlighted things that future studies could focus on. One example is using both microwave drive and an external flux signal to implement two-qubit gates, see Eq. (2.55). In general, two-qubit gates are a much more open question than single-qubit gates. Many techniques can be used to generate high-fidelity single-qubit gates limited by coherence.

A recommendation for future studies is to first focus on the set-up of the two-qubit optimization problem, then look into what optimization technique to use to solve it. A state-of-the-art black-box optimizer might be a good starting choice. Deep reinforcement learning is perhaps not a great choice, as there are a lot of hyperparameters that could be tuned. Ensure that realistic control distortions are modeled, investigate the validity of approximations such as the rotating wave approximation if it is used to derive the simulation model, and think about the system's parameters, for example, what coupling strengths to use. Hopefully, the ZZ coupling interpretation of the CZ-gate dynamics presented in this study will be useful.

Regarding machine learning, model learning is an approach that future studies could investigate. One could, for example, explore the effect of including an ansatz based on a theoretical model. Model-based reinforcement learning, the combination of reinforcement learning and model learning, is another interesting approach. A different implementation of deep reinforcement learning could perhaps yield strong two-qubit results. A starting point could be [53], which claims a high-fidelity 10 ns CZ-gate is possible.

Bibliography

[1] Alexander M. Dalzell et al. "Quantum algorithms: A survey of applications and end-to-end complexities". In: (Oct. 2023). URL: http://arxiv.org/abs/2310.03011.

[2] Ashley Montanaro. "Quantum algorithms: an overview". In: *npj Quantum Information* 2.1 (2016), pp. 1–8.

[3] Rajeev Acharya et al. *Quantum error correction below the surface code threshold*. 2024. arXiv: 2408.13687 [quant-ph]. URL: https://arxiv.org/abs/2408.13687.

[4] Sophie Choe. *Quantum computing overview: discrete vs. continuous variable models*. 2022. arXiv: 2206.07246 [quant-ph]. URL: https://arxiv.org/abs/2206.07246.

[5] Austin G. Fowler et al. "Surface codes: Towards practical large-scale quantum computation". In: *Phys. Rev. A* 86 (3 Sept. 2012), p. 032324. DOI: 10.1103/PhysRevA.86.032324. URL: https://link.aps.org/doi/10.1103/PhysRevA.86.032324.

[6] M. Werninghaus et al. "Leakage reduction in fast superconducting qubit gates via optimal control". In: *npj Quantum Information* 7 (1 Dec. 2021). ISSN: 20566387. DOI: 10.1038/s41534-020-00346-2.

[7] V. V. Sivak et al. "Model-Free Quantum Control with Reinforcement Learning". In: *Physical Review X* 12 (1 Mar. 2022). ISSN: 21603308. DOI: 10.1103/PhysRevX.12.011059.

[8] Leon Ding et al. "High-Fidelity, Frequency-Flexible Two-Qubit Fluxonium Gates with a Transmon Coupler". In: *Physical Review X* 13 (3 July 2023). ISSN: 21603308. DOI: 10.1103/PhysRevX.13.031035.

[9] Yuval Baum et al. "Experimental Deep Reinforcement Learning for Error-Robust Gate-Set Design on a Superconducting Quantum Computer". In: *PRX Quantum* 2 (4 Nov. 2021), p. 040324. DOI: 10.1103/PRXQuantum.2.040324. URL: https://link.aps.org/doi/10.1103/PRXQuantum.2.040324.

[10] Ho Nam Nguyen et al. "Reinforcement learning pulses for transmon qubit entangling gates". In: *Machine Learning: Science and Technology* 5.2 (June 2024), p. 025066. ISSN: 2632-2153. DOI: 10.1088/2632-2153/ad4f4d. URL: http://dx.doi.org/10.1088/2632-2153/ad4f4d.

[11] Marco Roth et al. “Analysis of a parametrically driven exchange-type gate and a two-photon excitation gate between superconducting qubits”. In: *Phys. Rev. A* 96 (6 Dec. 2017), p. 062323. DOI: 10.1103/PhysRevA.96.062323. URL: https://link.aps.org/doi/10.1103/PhysRevA.96.062323.

[12] J.R. Johansson, P.D. Nation, and Franco Nori. “QuTiP: An open-source Python framework for the dynamics of open quantum systems”. In: *Computer Physics Communications* 183.8 (Aug. 2012), pp. 1760–1772. ISSN: 0010-4655. DOI: 10.1016/j.cpc.2012.02.021. URL: http://dx.doi.org/10.1016/j.cpc.2012.02.021.

[13] Anders Persson. “Numerical methods for solving the time-dependent Schrödinger equation”. In: (2012). URL: https://www.lu.se/lup/publication/3363166.

[14] Thomas Mehen. *Notes on Baker-Campbell-Hausdorff (BCH) Formulae*. URL: https://webhome.phy.duke.edu/~mehen/760/ProblemSets/BCH.pdf.

[15] Q Ansel et al. “Introduction to theoretical and experimental aspects of quantum optimal control”. In: *Journal of Physics B: Atomic, Molecular and Optical Physics* 57.13 (June 2024), p. 133001. ISSN: 1361-6455. DOI: 10.1088/1361-6455/ad46a5. URL: http://dx.doi.org/10.1088/1361-6455/ad46a5.

[16] Line Hjortshøj Pedersen, Niels Martin Møller, and Klaus Mølmer. “Fidelity of quantum operations”. In: *Physics Letters A* 367.1 (2007), pp. 47–51. ISSN: 0375-9601. DOI: https://doi.org/10.1016/j.physleta.2007.02.069. URL: https://www.sciencedirect.com/science/article/pii/S0375960107003271.

[17] Tahereh Abad et al. “Universal Fidelity Reduction of Quantum Operations from Weak Dissipation”. In: *Physical Review Letters* 129 (15 Oct. 2022). ISSN: 10797114. DOI: 10.1103/PhysRevLett.129.150504.

[18] Tahereh Abad et al. “Impact of decoherence on the fidelity of quantum gates leaving the computational subspace”. In: *Quantum* 9 (Apr. 2025), p. 1684. ISSN: 2521-327X. DOI: 10.22331/q-2025-04-03-1684. URL: https://doi.org/10.22331/q-2025-04-03-1684.

[19] Re-Bing Wu et al. “Learning robust and high-precision quantum controls”. In: *Phys. Rev. A* 99.4 (2019), p. 042327. DOI: 10.1103/PhysRevA.99.042327.

[20] Elie Genois et al. “Quantum optimal control of superconducting qubits based on machine-learning characterization”. In: *arXiv preprint arXiv:2410.22603* (2024).

[21] P. Krantz et al. “A quantum engineer’s guide to superconducting qubits”. In: *Applied Physics Reviews* 6.2 (June 2019). ISSN: 1931-9401. DOI: 10.1063/1.5089550. URL: http://dx.doi.org/10.1063/1.5089550.

[22] Jens Koch et al. “Charge-insensitive qubit design derived from the Cooper pair box”. In: *Physical Review A - Atomic, Molecular, and Optical Physics* 76 (4 Oct. 2007). ISSN: 10502947. DOI: 10.1103/PhysRevA.76.042319.

[23] Dongxin Gao et al. “Establishing a New Benchmark in Quantum Computational Advantage with 105-qubit Zuchongzhi 3.0 Processor”. In: *Physical Review Letters* 134 (9 Mar. 2025), p. 090601. ISSN: 0031-9007. DOI: 10.1103/PhysRevLett.134.090601. URL: https://link.aps.org/doi/10.1103/PhysRevLett.134.090601.

[24] J. M. Gambetta et al. “Analytic control methods for high-fidelity unitary operations in a weakly nonlinear oscillator”. In: *Phys. Rev. A* 83 (1 Jan. 2011), p. 012308. DOI: 10.1103/PhysRevA.83.012308. URL: https://link.aps.org/doi/10.1103/PhysRevA.83.012308.

[25] Martijn F. S. Zwanenburg et al. “Single-Qubit Gates Beyond the Rotating-Wave Approximation for Strongly Anharmonic Low-Frequency Qubits”. In: (Mar. 2025). URL: http://arxiv.org/abs/2503.08238.

[26] Akram Youssry et al. “Experimental graybox quantum system identification and control”. In: *npj Quantum Information* 10 (1 Dec. 2024). ISSN: 20566387. DOI: 10.1038/s41534-023-00795-5.

[27] Felix Motzoi et al. “Simple pulses for elimination of leakage in weakly nonlinear qubits”. In: *Physical review letters* 103.11 (2009), p. 110501.

[28] Eric Hyyppä et al. “Reducing leakage of single-qubit gates for superconducting quantum processors using analytical control pulse envelopes”. In: (Feb. 2024). ISSN: 26913399. DOI: 10.1103/PRXQuantum.5.030353. URL: http://arxiv.org/abs/2402.17757.

[29] Uluk Rasulov and Ilya Kuprov. “Instrumental distortions in quantum optimal control”. In: *The Journal of Chemical Physics* 162.16 (Apr. 2025), p. 164107. ISSN: 0021-9606. DOI: 10.1063/5.0264092. eprint: https://pubs.aip.org/aip/jcp/article-pdf/doi/10.1063/5.0264092/20502267/164107_1_5.0264092.pdf. URL: https://doi.org/10.1063/5.0264092.

[30] 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 (2005), pp. 296–305.

[31] Callum W. Duncan et al. “Taming quantum systems: A tutorial for using shortcuts-to-adiabaticity, quantum optimal control, and reinforcement learning”. In: (Jan. 2025). URL: http://arxiv.org/abs/2501.16436.

[32] OpenAI. *Part 3: Intro to Policy Optimization*. URL: https://spinningup.openai.com/en/latest/spinningup/rl_intro3.html#part-3-intro-to-policy-optimization.

[33] John Schulman et al. “Proximal Policy Optimization Algorithms”. In: (July 2017). URL: http://arxiv.org/abs/1707.06347.

[34] Pontus Lindgren. *Master’s Thesis in Quantum Optimal Control*. https://github.com/startegen01/QOC-MSc/. 2025.

[35] Johannes Herrmann et al. “Frequency Up-Conversion Schemes for Controlling Superconducting Qubits”. In: (Oct. 2022). URL: http://arxiv.org/abs/2210.02513.

[36] Qblox. *QRM-RF / 2-18.5 GHz Qubit Readout Module Cluster Series 19" Rack Mounted*. URL: https://cdn.prod.website-files.com/653289e64ff83c71222f6bf2/67124d3e8ce642eb374dc7b4_QBLOX_PRODUCTSHEET_QRM-RF_V1_7.pdf.

[37] Niklas J. Glaser et al. "Sensitivity-Adapted Closed-Loop Optimization for High-Fidelity Controlled-Z Gates in Superconducting Qubits". In: (Dec. 2024). URL: http://arxiv.org/abs/2412.17454.

[38] Roy Frostig, Matthew Johnson, and Chris Leary. "Compiling machine learning programs via high-level tracing". In: 2018. URL: https://mlsys.org/Conferences/doc/2018/146.pdf.

[39] Diederik P. Kingma and Jimmy Ba. "Adam: A Method for Stochastic Optimization". In: (Dec. 2014). URL: http://arxiv.org/abs/1412.6980.

[40] Jacob Adkins, Michael Bowling, and Adam White. "A Method for Evaluating Hyperparameter Sensitivity in Reinforcement Learning". In: *Advances in Neural Information Processing Systems*. Ed. by A. Globerson et al. Vol. 37. Curran Associates, Inc., 2024, pp. 124820–124842. URL: https://proceedings.neurips.cc/paper_files/paper/2024/file/e1cadf5f02cc524b59c208728c73f91c-Paper-Conference.pdf.

[41] Shengyong Li et al. "Robust Quantum Control using Reinforcement Learning from Demonstration". In: (Mar. 2025). URL: http://arxiv.org/abs/2503.21085.

[42] Keiko Nagami and Mac Schwager. "HJB-RL: Initializing Reinforcement Learning with Optimal Control Policies Applied to Autonomous Drone Racing." In: *Robotics: science and systems*. 2021, pp. 1–9.

[43] S. A. Caldwell et al. "Parametrically Activated Entangling Gates Using Transmon Qubits". In: *Physical Review Applied* 10 (3 Sept. 2018). ISSN: 23317019. DOI: 10.1103/PhysRevApplied.10.034050.

[44] Emanuel Knill et al. "Randomized benchmarking of quantum gates". In: *Physical Review A—Atomic, Molecular, and Optical Physics* 77.1 (2008), p. 012307.

[45] David C McKay et al. "Efficient Z gates for quantum computing". In: *Physical Review A* 96.2 (2017), p. 022330.

[46] J. Rapin and O. Teytaud. *Nevergrad - A gradient-free optimization platform*. https://GitHub.com/FacebookResearch/Nevergrad. 2018.

[47] Liangyu Chen. *Readout and Control Techniques Towards Scalable Superconducting Quantum Processors*. 2025. ISBN: 9789181031850.

[48] Amrit De. "Fast Quantum Control for Weakly Nonlinear Qubits: On Two-Quadrature Adiabatic Gates". In: (Sept. 2015). URL: http://arxiv.org/abs/1509.07905.

[49] Zijun Chen et al. "Measuring and suppressing quantum state leakage in a superconducting qubit". In: *Physical review letters* 116.2 (2016), p. 020501.

[50] Simon Pettersson Fors, Jorge Fernández-Pendás, and Anton Frisk Kockum. "Comprehensive explanation of ZZ coupling in superconducting qubits". In: (Aug. 2024). URL: http://arxiv.org/abs/2408.15402.

[51] Andre R.R. Carvalho et al. “Error-Robust Quantum Logic Optimization Using a Cloud Quantum Computer Interface”. In: *Physical Review Applied* 15 (6 June 2021). ISSN: 23317019. DOI: 10.1103/PhysRevApplied.15.064054.

[52] Tim Salimans et al. “Evolution Strategies as a Scalable Alternative to Reinforcement Learning”. In: (Mar. 2017). URL: http://arxiv.org/abs/1703.03864.

[53] Bijita Sarma and Michael J. Hartmann. “Designing fast quantum gates using optimal control with a reinforcement-learning ansatz”. In: *Phys. Rev. Appl.* 23 (1 Jan. 2025), p. 014015. DOI: 10.1103/PhysRevApplied.23.014015. URL: https://link.aps.org/doi/10.1103/PhysRevApplied.23.014015.

A
Derivations

A.1 Single-qubit Gaussian Pulse

The standard Gaussian pulse used in DRAG can be derived by considering $\Omega_y(t) = 0, \delta(t) = 0, \lambda_i = 0 \forall i > 0$. For this case, the qubit subspace is decoupled from the other energy levels and

$$H_{\text{qb}}(t) = -\frac{\Omega_x(t)}{2}\sigma_x. \tag{A.1}$$

As $\hat{H}_{\text{qb}}$ in this case commutes with itself $\forall t$ the time evolution is

$$U_{\text{qb}}(t) = \exp\left(-i\frac{1}{2}\int_0^t \Omega_x(t)dt\hat{\sigma}_x\right) \tag{A.2}$$

This is the form of an arbitrary rotation on the Bloch sphere, which in general is given by

$$\hat{R}_{\vec{n}}(\delta) = \exp(-i\frac{\delta}{2}\vec{n}\cdot\vec{\sigma}) = \cos(\frac{\delta}{2})\mathbf{I} - i\sin(\frac{\delta}{2})\vec{n}\cdot\vec{\sigma} \tag{A.3}$$

where δ is the rotation angle on the Bloch sphere and $\vec{n}$ is the unit length rotation axis. This implies that the time evolution operator $\hat{U}(t_g) = \sigma_x$, a π-pulse, if $\delta = \pi + 2\pi n$, where n is an integer. A criterion is then

$$\int_0^{t_g} \Omega_x(t)dt = \pi \tag{A.4}$$

The criterion is nice as it implies the smallest Ω_x, implying less leakage. The criterion is fulfilled by a Gaussian pulse $\Omega_x(t) = \Omega_G(t)$. $\Omega_G(0) = 0$ and $\Omega_G(t_g) = 0$ has also been enforced to obtain Eq. (2.61).

A.2 $[\hat{a}^\dagger\hat{a}, \hat{a}^\dagger\hat{a}^\dagger\hat{a}\hat{a}] = 0$

Observe that

$$[\hat{a}^\dagger\hat{a}, \hat{a}^\dagger\hat{a}^\dagger\hat{a}\hat{a}] = \hat{a}^\dagger(\hat{a}\hat{a}^\dagger\hat{a}^\dagger\hat{a} - \hat{a}^\dagger\hat{a}\hat{a}\hat{a}^\dagger)\hat{a}. \tag{A.5}$$

Rewrite

$$\hat{a}\hat{a}^\dagger\hat{a}^\dagger\hat{a} = (1 + \hat{a}^\dagger\hat{a})(\hat{a}\hat{a}^\dagger - 1) = \hat{a}^\dagger\hat{a}\hat{a}\hat{a}^\dagger - 1 + \hat{a}\hat{a}^\dagger - \hat{a}^\dagger\hat{a} = \hat{a}^\dagger\hat{a}\hat{a}\hat{a}^\dagger \tag{A.6}$$

using $[\hat{a}, \hat{a}^\dagger] = 1$. Then

$$[\hat{a}^\dagger\hat{a}, \hat{a}^\dagger\hat{a}^\dagger\hat{a}\hat{a}] = 0. \tag{A.7}$$

B

System Parameters

Table B.1: Two-qubit system.

Parameter	**Value**
Qubit Frequency ω_1 (GHz)	3.528528
Qubit Frequency ω_2 (GHz)	4.056419
Coupler Frequency ω_c^0 (GHz)	6.287653
Anharmonicity α_1 (GHz)	-0.228498
Anharmonicity α_2 (GHz)	-0.216387
Anharmonicity α_c (GHz)	-0.120
Coupling Strength g_{1c} (GHz)	0.029770
Coupling Strength g_{2c} (GHz)	0.031085
Coupling Strength g_{12} (GHz)	0
Truncation of $\|i, j, k\rangle$	$i, j, k \in \{0, 1, 2\}$
Relaxation Time T_1 (µs)	80
Dephasing Time T_2 (µs)	40

Table B.2: Single-qubit system. Note that due to the rotating frame, the qubit frequency ω_q does not matter for the simulation.

Parameter	**Value**
Anharmonicity α (MHz)	-350
Truncation d	7
Relaxation Time T_1 (µs)	80
Dephasing Time T_2 (µs)	40

C
Supporting Results

C.1 Single-qubit Supporting Results

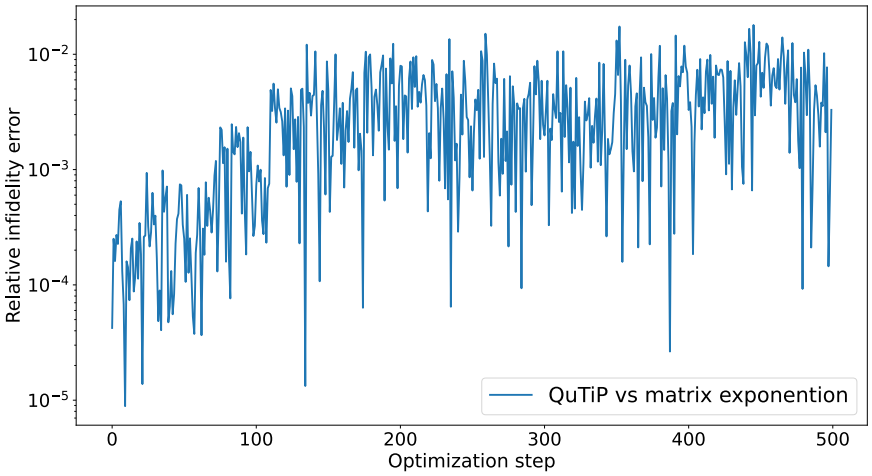


Figure C.1: The figure shows the relative error in infidelity for the low-pass filtered GRAPE with $t_g = 8$ ns. The figure compares the matrix exponential simulation method's infidelity with QuTiP's simulation method. The relative infidelity error is $\frac{|I_{\text{QuTiP}} - I_{\text{exp(A)}}|}{I_{\text{exp(A)}}}$. The takeaway is that the two methods yield infidelities that differ by at most about 1%. This shows that the matrix exponentiation approach and QuTiP's ODE solver approach to solving the Schrödinger equation yield results that are similar enough for this study.

C.2 Two-qubit Supporting Results

Parameter	Value
Infidelity	0.0008
Decoherence	0.0076
t_{rise}, t_{fall}	3.47 ns
$t_{plateau}$	190.14 ns
θ	0.3880
$\Delta\omega_{cos}$	-0.0222 GHz
δ_{cos}	0.0578
$\Delta\omega_{sin}$	0.0000 GHZ
δ_{sin}	0.0000

(a) 197.1 ns CZ-gate

Parameter	Value
Infidelity	0.0019
Decoherence	0.0042
t_{rise}, t_{fall}	2.04 ns
$t_{plateau}$	104.20 ns
θ	0.3671
$\Delta\omega_{cos}$	-0.0457 GHz
δ_{cos}	0.0385
$\Delta\omega_{sin}$	0.0074 GHz
δ_{sin}	-0.0001

(b) 108.3 ns CZ-gate

Table C.1: The parameters of the 197.1 ns and 108.3 ns optimized ansatzes. The unit of $\delta_{cos}, \delta_{sin}, \theta$ is defined by $\Phi_0 = 1$. The parameters have been rounded.

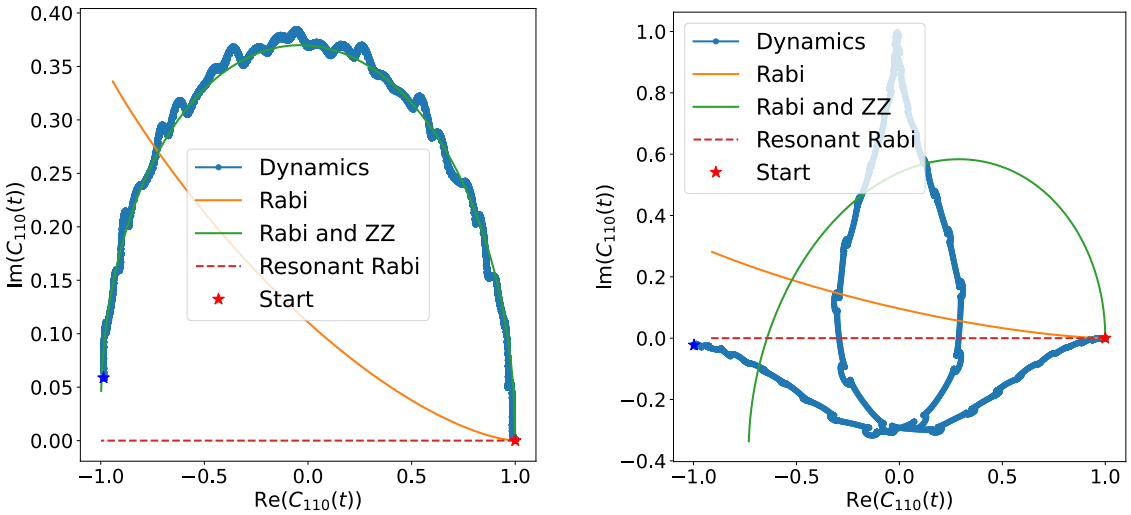
Parameter	Value
Infidelity	0.0023
Decoherence	0.0029
t_{rise}, t_{fall}	2.03 ns
$t_{plateau}$	69.63 ns
θ	0.3468
$\Delta\omega_{cos}$	-0.3556 GHz
δ_{cos}	0.0590
$\Delta\omega_{sin}$	0.0000 GHz
δ_{sin}	0.0000

(a) 73.7 ns CZ-gate

Parameter	Value
Infidelity	0.0069
Decoherence	0.0160
t_{rise}, t_{fall}	14.00 ns
$t_{plateau}$	384.52 ns
θ	0.3503
$\Delta\omega_{cos}$	0.0119 GHz
δ_{cos}	0.0245
$\Delta\omega_{sin}$	0.0181 GHz
δ_{sin}	0.0000

(b) 412.5 ns CZ-gate

Table C.2: The parameters of the 73.7 ns and 412.5 ns optimized ansatzes. The 73.7 ns CZ gate does not use the $|110\rangle \Leftrightarrow |200\rangle$ transition. The unit of $\delta_{cos}, \delta_{sin}, \theta$ is defined by $\Phi_0 = 1$. The parameters have been rounded.



(a) 412.5 ns CZ-gate **(b)** 73.7 ns CZ-gate

Figure C.2: The phase space dynamics of $C_{110}(t)$ with fit models for the slow detuned 412.5 ns CZ gate and the coupler exciting 73.7 ns CZ-gate. For the 412.5 ns case shown in Fig. C.2a, oscillations seem to be a lot smaller relative to $\Im(C_{110}(t))$. It is not surprising that the arch has a larger amplitude as the minimal population $\min(|C_{110}(t)|^2)$ shown in Fig. 4.14a is much larger than for the other ansatzes. Fig. C.2b reinforces the difference between this ansatz and $|110\rangle \Leftrightarrow |200\rangle$ CZ gates. The fit in Fig. C.2b is a full period Rabi oscillation instead of half a period as for the other fits.

DEPARTMENT OF MICROTECHNOLOGY AND NANOSCIENCE
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden
www.chalmers.se

CHALMERS
UNIVERSITY OF TECHNOLOGY