Contents

1 Introduction

The emerging field of quantum technology has many promising applications, one of them is quantum computation (QC), which currently is a very active area of research. Quantum computers make use of quantum mechanical effects such as superposition, entanglement, and interference to design powerful algorithms. These algorithms could be used to solve some hard problems which would not be possible to solve using classical computation, such as efficient prime-number factoring [1]. It can also be used to reduce the time complexity of some commonly used algorithms [2]. Current quantum computers are very susceptible to decoherence and noise, and thus will not have any commercial use any time soon, but stand as an important proof of concept. The most common model for quantum computation is the circuit model, which is analogous to the classical circuits used for classical computers. Gates are replaced by unitary transformations (quantum gates) and bits by qubits. To achieve the computational advantage it is important to construct robust, noise-resilient quantum gates. A good candidate for this is holonomic quantum computation [3,4], which is based on the Berry phase [5] and its non-Abelian and/or non-adiabatic generalizations [6,7,8]. These methods are only dependent on the geometry of the system and thus are resilient to local errors in the dynamical evolution.

The idea that elements of computation should be limited to two-dimensional qubits is sort of an arbitrary choice that most likely rose out of convenience due to binary logic. So why binary logic? It is simply the easiest non-trivial example, in binary things can be either 1 or 0, `True` or `False`, **on** or **off**. Due to its simplicity, it is no wonder that this is how the first computer was designed. But are we limited to bits? As early as 1840 a mechanical trenary (three-valued logic) calculation device was built by Thomas Fowler [9], and in 1958 the first electronic trenary computer was built by the Soviet Union [10]. Even though it had many advantages over the binary computer it never saw the same widespread success. There is nothing in theory that forbids a higher dimensional computational basis, even more so when it comes to quantum computers, where the implementation of the elements of computation already surpasses the simplicity of **on** and **off**. There are promising qudit results that show potential [11,12,13], and in the review article [14] a good overview of the field is given and further research into the topic is encouraged.

In this report we will show how to find a new geometric phase based scheme to implement qudits which could be more efficient than some current ones by making use of dark paths for increased parameter control and auxiliary states for increased fidelity. We do this by generalizing the idea of the scheme from [15]. The report is structured as follows. The background section is split into two parts where the first part serves as a quick introduction to the most important aspects of quantum mechanics as well as the commonly used notation. Then follows a part more concerned with quantum computation, quantum information, and some of the more advanced quantum mechanical concepts that those are built upon. Then the main results are shown, first an explicit example for the qutrit and then how it generalizes in higher dimensions. The report ends with conclusions and a brief outlook.

2 Background

2.1 Basic quantum mechanics

This part offers a quick introduction to the necessary quantum mechanics for readers who are not familiar with the subject. Quantum mechanics is the framework developed to handle modern physics and is famously somewhat hard to understand intuitively; a good baseline understanding is however obtainable with knowledge of linear algebra combined with the postulates of QM. But to understand the contents of this project it should be enough with linear algebra and some physical intrepretation. The section focuses mostly on familiarising the reader with the notation used in quantum mechanics and some fundamental ideas. I suggest chapters 1 and 2 of [16] for a more complete introduction.

2.1.1 Quantum states and Dirac notation

First, let us define what is meant by the term **state**. In classical physics, a state is defined by the position and momentum of all its constituents. An example is a system of N particles, the state of which would be given by $\{(\vec{x}, \vec{p})_i\}_{i=1}^N$, where $\vec{x}, \vec{p} \in \mathbb{R}^3$ are the position and momentum in 3 dimensions. In Quantum Mechanics (QM) it is more subtle since exact information of the system cannot be obtained in the same way. A state is represented by a normalized vector in a **Hilbert space**, $\mathcal{H}$, a complex inner product space. For $u, v \in \mathcal{H}$ the inner product has the following properties:

1. The inner product is conjugate symmetric,
$$\langle u, v\rangle = \overline{\langle v, u\rangle} \in \mathbb{C}$$
.
2. The inner product is linear in the second argument, for constants $a, b \in \mathbb{C}$,
$$\langle u, av_1 + bv_2\rangle = a\langle u, v_1\rangle + b\langle u, v_2\rangle$$
.
3. The inner product is positive definite,
$$\langle u, u\rangle = 0 \iff u = 0$$
.

These properties can be combined to find some other useful facts. Combining the 1st and 2nd property, one obtains

$$\langle au_1 + bu_2, v\rangle = \overline{\langle v, au_1 + bu_2\rangle} = a^*\overline{\langle v, u_1\rangle} + b^*\overline{\langle v, u_2\rangle} = a^*\langle u_1, v\rangle + b^*\langle u_2, v\rangle, \quad (1)$$

which shows that the inner product is anti-linear in the second argument. The 1st and 3rd property yield

$$\langle u, u\rangle = \overline{\langle u, u\rangle} \implies \text{Im}(\langle u, u\rangle) = 0, \quad (2)$$

or in words, the inner product of two identical vectors is a real number.

A quantum state is represented by a normalized vector v, $\langle v, v\rangle = 1$, in a Hilbert space $\mathcal{H}$. Now a property of vector spaces is that any vector can be multiplied by a matrix, and the resulting vector will be in the same vector space. This is what is meant when a quantum state is **acted** upon. For a matrix A, we have that

$$\mathcal{H} \ni v \xrightarrow{A} Av = v' \in \mathcal{H}. \tag{3}$$

That is, acting on a state alters it in various ways. Now let us go from this linear algebra notation to the Dirac notation commonly used in quantum mechanics, also known as bra-ket notation. Vectors are replaced by **kets**,$|\rangle$, or **bras**,$\langle|$,

$$v \mapsto |\psi\rangle\,,$$
$$v^\dagger \mapsto \langle\psi|$$

and matrices are replaced by linear operators

$$A \mapsto \hat{A}.$$

The same rules apply to these as for the usual vectors. The labels inside the brackets do not in themselves have any meanings and are in some sense only that, labels, but more often than not it is used to represent some property of the state. With this notation the inner product between two states $|\psi\rangle\,, |\varphi\rangle$ is written and defined as

$$\langle\psi|\varphi\rangle = a_1^* b_1 + a_2^* b_2 + \cdots + a_n^* b_n = \begin{pmatrix} a_1^* & a_2^* & \dots & a_n^* \end{pmatrix} \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{pmatrix}. \tag{4}$$

From the properties defined earlier the relations $(|\psi\rangle)^\dagger = \langle\psi|$ and $(\langle\psi|\varphi\rangle)^\dagger = \langle\varphi|\psi\rangle$, suggests another way to define the states simply as

$$|\psi\rangle \doteq \begin{pmatrix} a_1 \\ a_2 \\ \vdots \\ a_n \end{pmatrix},\ \langle\psi| = (|\psi\rangle)^\dagger = (a_1^*, a_2^*, \dots, a_n^*),\ a_1, a_2, \dots, a_n \in \mathbb{C}, \tag{5}$$

which is nothing more than a complex vector. The dotted equal sign,$\doteq$, means *is represented by* and is often used when representing bras and kets as matrices and vectors. A common way to represent kets is as a linear combination of eigenkets, which corresponds to the set of eigenvectors of some Hermitian operator. Thus, any ket can be written as a linear combination of these eigenkets if they span the Hilbert space

$$|\Psi\rangle = \sum_\psi c_\psi\, |\psi\rangle\,. \tag{6}$$

In the following, there are some example calculations to get familiar with the Dirac-notation. Let us consider what the effect of the Pauli-Z operator of a spin-$\frac{1}{2}$ particle is

a) on the up position $|\psi\rangle = |\downarrow\rangle$

b) on an arbitrary superposition of up and down $|\psi\rangle = \alpha\,|\uparrow\rangle + \beta\,|\downarrow\rangle\,, |\alpha|^2 + |\beta|^2 = 1, \alpha, \beta \in \mathbb{C}$

Let us first show how this would be done using the matrix representation and then using Dirac notation. Here the kets $\{|\uparrow\rangle \doteq \begin{pmatrix}1\\0\end{pmatrix}, |\downarrow\rangle \doteq \begin{pmatrix}0\\1\end{pmatrix}\}$ span the Hilbert space of the particle, they also form an orthonormal basis. The Pauli-Z operator in the matrix representation in this basis is $\sigma_z = \begin{pmatrix}1 & 0\\0 & -1\end{pmatrix}$. Then we have that

a) We simply write out the matrices and vectors and get that

$$\sigma_z\,|\downarrow\rangle \doteq \begin{pmatrix}1 & 0\\0 & -1\end{pmatrix}\begin{pmatrix}0\\1\end{pmatrix} = \begin{pmatrix}0\\-1\end{pmatrix} \doteq -\,|\downarrow\rangle\,. \tag{7}$$

b) Same as before but with a superposition (linear combination) of up and down

$$\begin{aligned}\sigma_z\left(\alpha\,|\uparrow\rangle + \beta\,|\downarrow\rangle\right) \doteq\ & \begin{pmatrix}1 & 0\\0 & -1\end{pmatrix}\left[\alpha\begin{pmatrix}1\\0\end{pmatrix} + \beta\begin{pmatrix}0\\1\end{pmatrix}\right]\\ &= \begin{pmatrix}\alpha\\-\beta\end{pmatrix}\\ &= \alpha\begin{pmatrix}1\\0\end{pmatrix} - \beta\begin{pmatrix}0\\1\end{pmatrix}\\ &=\ \alpha\,|\uparrow\rangle - \beta\,|\downarrow\rangle\,.\end{aligned} \tag{8}$$

The matrix representation works great for small Hilbert spaces since manipulating small matrices by hand is not very time consuming. Consider now the same calculation using Dirac notation, here the inner product will be a key aspect, and making use of the orthonormality of $|\uparrow\rangle\,, |\downarrow\rangle$ it follows that $\langle\uparrow|\uparrow\rangle = \langle\downarrow|\downarrow\rangle = 1$ and $\langle\uparrow|\downarrow\rangle = \langle\downarrow|\uparrow\rangle = 0$. With Dirac notation operators can be expressed as outer products of bras and kets, thus Pauli-Z can be written as $\sigma_z = |\uparrow\rangle\langle\uparrow| - |\downarrow\rangle\langle\downarrow|$. From this follows:

a)

$$\begin{aligned}\sigma_z\,|\downarrow\rangle &= \left(|\uparrow\rangle\langle\uparrow| - |\downarrow\rangle\langle\downarrow|\right)|\downarrow\rangle\\ &= |\uparrow\rangle\underbrace{\langle\uparrow|\downarrow\rangle}_{=0} - |\downarrow\rangle\underbrace{\langle\downarrow|\downarrow\rangle}_{=1}\\ &= 0\,|\uparrow\rangle - 1\,|\downarrow\rangle\\ &= -\,|\downarrow\rangle\,.\end{aligned} \tag{9}$$

b)

$$\begin{aligned}\sigma_z\left(\alpha\,|\uparrow\rangle + \beta\,|\downarrow\rangle\right) &= \left(|\uparrow\rangle\langle\uparrow| - |\downarrow\rangle\langle\downarrow|\right)\left(\alpha\,|\uparrow\rangle + \beta\,|\downarrow\rangle\right)\\ &= \alpha\,|\uparrow\rangle\underbrace{\langle\uparrow|\uparrow\rangle}_{=1} - \alpha\,|\downarrow\rangle\underbrace{\langle\downarrow|\uparrow\rangle}_{=0} + \beta\,|\uparrow\rangle\underbrace{\langle\uparrow|\downarrow\rangle}_{=0} - \beta\,|\downarrow\rangle\underbrace{\langle\downarrow|\downarrow\rangle}_{=1}\\ &= \alpha\,|\uparrow\rangle - \beta\,|\downarrow\rangle\,.\end{aligned} \tag{10}$$

Note that the final answer is the same for both cases, but using Dirac notation there is no need to bother with explicit vectors and matrices, that is one of the strong advantages of the Dirac notation. These calculations are quite analogous to how a quantum gate acting on a qubit works which will be discussed further later in this section. In Dirac notation, any general matrix A can be written as $A = \sum_{ij} A_{ij} |i\rangle \langle j|$. Measuring a quantum state $|\psi\rangle = \sum_n c_n |n\rangle$, $c_n \in \mathbb{C}$, collapses it into a single base ket of the measured basis

$$|\psi\rangle \xrightarrow{\text{Measurement}} |n\rangle , \tag{11}$$

with probability $|c_n|^2$.

2.1.2 Time evolution and the Schrödinger equation

Understanding the dynamics of a quantum system is of great importance when implementing quantum gates. Thus we can ask; how does the system evolve when time goes from t_0 to some later time $t_1 > t_0$? We can write this as an operator equation $U(t_1, t_0) |\psi(t_0)\rangle = |\psi(t_1)\rangle$, where U is called the time evolution operator. To determine the form of U let us state some physically motivated constraints. To preserve probability U must be unitary, $U^\dagger U = UU^\dagger = 1$. The second property that needs to be fulfilled is composition, $U(t_2, t_0) = U(t_2, t_1)U(t_1, t_0)$, which is to say that applying two consecutive time transformations, first between $t_0 \to t_1$ and then $t_1 \to t_2$, is equivalent to applying one transformation between $t_0 \to t_2$. Proceeding, let us define how the time evolution operator would act under an infinitesimal change dt,

$$U(t + dt, t) = \mathbf{1} - i\,\Omega\,dt \tag{12}$$

with Ω being Hermitian, $\Omega = \Omega^\dagger$. Ω can be time dependent, and then must be evaluated at t. As dt approaches 0, U reduces to an identity operator, which is appropriate since if zero time passes the quantum state stay unchanged. Let us check the two other properties:

1. Unitarity

$$\begin{aligned} U(t_0 + dt, t_0)U(t_0 + dt, t_0)^\dagger &= (\mathbf{1} - i\,\Omega\,dt)\,(\mathbf{1} + i\,\Omega\,dt) \\ &= \mathbf{1} + i\,\Omega\,dt - i\,\Omega\,dt + \mathcal{O}(dt^2) \\ &\simeq \mathbf{1}. \end{aligned} \tag{13}$$

2. Composition (assuming, Ω time independent)

$$\begin{aligned} U(t + dt_1 + dt_2, t + dt_1)U(t + dt_1, t) &= (\mathbf{1} - i\,\Omega\,dt_2)(\mathbf{1} - i\,\Omega\,dt_1) \\ &= \mathbf{1} - i\,\Omega\,dt_1 - i\,\Omega\,dt_2 + \mathcal{O}(dt^2) \\ &\simeq \mathbf{1} - i\,\Omega(dt_1 + dt_2) \\ &= U(t + dt_1 + dt_2, t). \end{aligned} \tag{14}$$

From the operator Ω the Hamiltonian, H, is defined, $\Omega = \frac{H}{\hbar}$, where $\hbar$ is Planck's constant. Thus the Hamiltonian will always be Hermitian by definition. So for an infinitesimal time shift the operator has the form $U(t + dt, t) = \mathbf{1} - i\,\frac{H}{\hbar}\,dt$. Time evolution is closely tied to one of Quantum Mechanics postulates, which is that a closed

quantum state evolves according to the Schrödinger equation. From the Schrödinger equation it is clear how Ω is related to the Hamiltonian. It is common to set $\hbar = 1$, so let us do that from now on. Then the Schrödinger equation is given by

$$\begin{aligned} i\frac{\partial}{\partial t}\left|\psi(t)\right\rangle &= \lim_{dt\to 0} i\left[\frac{\left|\psi(t+dt)\right\rangle - \left|\psi(t)\right\rangle}{dt}\right] \\ &= \lim_{dt\to 0} i\left[\frac{U(t+dt,t)-\mathbf{1}}{dt}\right]\left|\psi(t)\right\rangle \\ &= \lim_{dt\to 0} i\left[\frac{\mathbf{1}-iH\,dt+\mathcal{O}(dt^2)-\mathbf{1}}{dt}\right]\left|\psi(t)\right\rangle \\ &= \lim_{dt\to 0}\left[H+\mathcal{O}(dt)\right]\left|\psi(t)\right\rangle = H\left|\psi(t)\right\rangle. \end{aligned} \tag{15}$$

Now it is clear that the time evolution operator is related to the Schrödinger equation: the solution $|\psi(t)\rangle$ can be written in terms of our time evolution operator as $|\psi(t)\rangle = U(t,t_0)\,|\psi(t_0)\rangle$. So given an initial condition we have solved the Schrödinger equation if we can determine $U(t,t_0)$. To find the time evolution operator for any time t, we could successively repeat the infinitesimal operator until the desired time is reached $t_0 \mapsto t_0 + dt \mapsto t_0 + 2\,dt \mapsto \cdots \mapsto t - dt \mapsto t$. For the case of a time independent Hamiltonian, H, with equidistant time steps, it could be written as

$$\lim_{N\to\infty}\left(\mathbf{1} - i\,H\frac{(t-t_0)}{N}\right)^N = \exp\left(-i\,H(t-t_0)\right) \tag{16}$$

For simplicity let us assume that $t_0 = 0$, then the time evolution operator for a finite time for a time independent Hamiltonian could be written as $U(t,0) = e^{-i\,H\,t}$. If the Hamiltonian has time dependence but it commutes with itself at different times, then an integral can be introduced in the exponential, $U(t,0) = e^{-i\int_0^t H(t')\,dt'}$. If the Hamiltonian does not commute at different times it becomes a bit more involved and will not be covered here. But the formal solution can be written as $U(t,0) = \mathcal{T}e^{-i\int_0^t H(t')\,dt'}$, where $\mathcal{T}$ is time-ordering. Closely related to the time evolution are dynamical and geometrical phases which act upon a quantum states as time goes by. These phases will be relevant for the holonomic implementation of the quantum gates, this is discussed further in Section 2.2.6.

To conclude, finding the time evolution operator corresponds to finding the solution to the Schrödinger equation, which governs the time evolution of quantum systems.

2.2 Quantum computation and quantum information theory

2.2.1 The qubit

A classical bit is a two-state (binary) system, which means it can occupy either one of two states, 0 or 1. So with n bits, there are 2^n possible states.

The general way to define a qubit would be any two dimensional quantum system, meaning that the Hilbert space is of dimension 2, $\dim(\mathcal{H}) = 2$. A qubit state is any normalized superposition (linear combination) of the basis kets $|0\rangle$ and $|1\rangle$, i.e.:

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle\,,\ \alpha,\beta\in\mathbb{C},\ |\alpha|^2 + |\beta|^2 = 1. \tag{17}$$

Most of the time it is better to work with the abstract concept of qubits without focusing on the exact physical implementation. Whether one chooses to encode the qubit states, $|0\rangle$ and $|1\rangle$, into a spin-$1/2$ or energy levels of an ion, the mathematical abstraction would remain the same. The coefficients α and β are the probability amplitudes, and tells with what probability one will find either $|0\rangle$ or $|1\rangle$ when measuring: one finds $|0\rangle$ or $|1\rangle$ with probability $|\alpha|^2$ and $|\beta|^2$, respectively.

The consequence of superposition is that the qubit is not limited to just either 0 or 1, but rather any combination of the two. Superposition together with other QM effects is what is used to create classically impossible algorithms [1,2]. The combined state of two qubits $|\psi_1\rangle$ and $|\psi_2\rangle$ is given by

$$|\psi_1\rangle \otimes |\psi_2\rangle = (\alpha_1 |0\rangle + \beta_1 |1\rangle) \otimes (\alpha_2 |0\rangle + \beta_2 |1\rangle) \tag{18}$$

the tensor product is often omitted and one would write $|\psi_1\rangle \otimes |\psi_2\rangle = |\psi_1\rangle |\psi_2\rangle = |\psi_1, \psi_2\rangle$. In the case where a state cannot be written as a tensor product between two other states, it is said to be an entangled state. Entanglement between qubits is an essential part of many QC applications. A combined state of n qubits is often written as $|x\rangle^{\otimes n}$, $x \in \{0, 1\}$ which is spanned by kets on the form $|x_0 x_1 x_2 ... x_n\rangle$, $x_i \in \{0, 1\}$, representing all possible bit strings of length n. However, this project will limited to single qubit(qudit) systems.

2.2.2 From classical logic gates to quantum gates

A model used in classical computation is the circuit model, which consists of wires and logic gates. The wires carry the bit information, 0, 1, and the gates apply different operations. A common gate is the NOT-gate which flips the content of the bit $\texttt{NOT}(0) = 1$ and $\texttt{NOT}(1) = 0$. Some other common logic gates are shown in Table 1, further, in Figure 1 is an example of a classical circuit converted into an equivalent quantum circuit.

Input		AND	XOR	OR
0	0	0	0	0
0	1	0	1	1
1	0	0	1	1
1	1	1	0	1

Table 1: Some common classical logic gates.

To make logic gates suitable for quantum computation the classical logic gate has to be adjusted. A quantum gate has some desired properties:

1. Reversibility,

2. Conservation of probability.

It will turn out that these properties have more or less the same consequence. The first property is not achieved by the classical gates. For example, given the output of the XOR-gate it is impossible to know the state of the bits that went into the gate. An

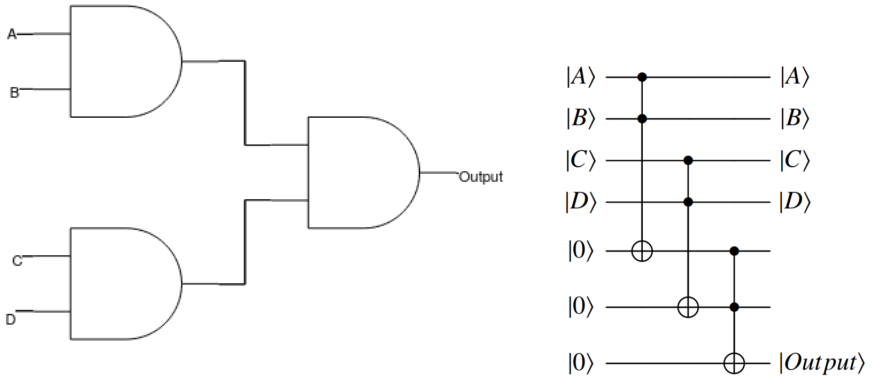

Figure 1: Example of how a classical circuit would be translated into a quantum circuit. The classical `AND` gates are replaced by the Toffoli gate.

output of 1 only conveys that one of the inputs were 1, but not which one. Adding another output that simply is one of the inputs can make the gate reversible. In the quantum mechanical context, a quantum logic gate is a quantum operator acting on the state of the qubit, $A|\psi\rangle = |\psi'\rangle$, so the equivalence of the classical reversibility would be the existence of an inverse operator A^{-1} such that the operation could be reversed $A^{-1}|\psi'\rangle = A^{-1}A|\psi\rangle = \mathbf{1}|\psi\rangle = |\psi\rangle$.

Conservation of probability does not in itself have any classical analogue due to it being a quantum property. More concretely it means conservation of the inner product. Given the states $U|\psi\rangle = |\psi'\rangle$ and $U|\varphi\rangle = |\varphi'\rangle$, then the inner product of $|\psi\rangle$ and $|\varphi\rangle$ is the same as for $|\psi'\rangle$ and $|\varphi'\rangle$. Thus one can compute

$$\langle\psi'|\varphi'\rangle = \langle\psi|U^\dagger U|\varphi\rangle = \langle\psi|\varphi\rangle \tag{19}$$

and see that this only holds if $U^\dagger U = UU^\dagger = \mathbf{1}$, which happens to be the condition for U to be an unitary operator. From the unitarity it is possible to also see that $U^\dagger = U^{-1}$. The reversibility is also a consequence of the operators being unitary.

There are many ways to achieve the creation and implementation of quantum gates. Some of them will be discussed in more detail in Section 2.2.6. Let us see in general how a quantum gate could be implemented. To apply a quantum gate to the qubit means to operate on a quantum state with a unitary operator. The time evolution operator discussed priorly is unitary by definition, and it acts on states according to the Schrödinger equation. The time evolution operator seems like a good candidate to be used to implement quantum gates since they share many common properties.

2.2.3 Some important gates and universal computation

Quantum gates are used to apply operations to qubit states. By using gates as standardized building blocks for algorithms we can restrict to the creation of these gates. A set of gates is called universal if those gates can be used to approximate any gate to arbitrary precision. Some of the most common gates are the Pauli gates, X, Y, Z,

which are equivalent to the Pauli matrices, $\sigma_x, \sigma_y, \sigma_z$. They are defined as

$$X \doteq \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \; Y \doteq \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}, \; Z \doteq \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}. \tag{20}$$

It is more common to use only X and Z since $Y = i\,X\,Z$. The X-gate is somewhat analogous to the classical NOT-gate, with the effect that it swaps the input qubit states,

$$\begin{cases} X\,|0\rangle = |1\rangle\,, \\ X\,|1\rangle = |0\rangle\,. \end{cases}$$

The Z-gate introduces a phase-flip between the inputs, it does nothing to $|0\rangle$ and reverses the sign of $|1\rangle$,

$$\begin{cases} Z\,|0\rangle = |0\rangle\,, \\ Z\,|1\rangle = -\,|1\rangle\,. \end{cases}$$

Superposition is one of the key aspects that make QC able to surpass classical algorithms. The Hadamard gate, H, (not to be confused with the Hamiltonian H) creates a superimposed quantum state, and is widely used all over QC. The gate is defined as

$$\mathrm{H} \doteq \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}. \tag{21}$$

Another noteworthy property is that H is its own inverse, $\mathrm{HH} = \mathbf{1}$, so it can be used both to open and close superposition. More explicitly, the gate effect is,

$$\begin{cases} \mathrm{H}\,|0\rangle = \dfrac{1}{\sqrt{2}}(|0\rangle + |1\rangle), \\ \mathrm{H}\,|1\rangle = \dfrac{1}{\sqrt{2}}(|0\rangle - |1\rangle) \end{cases} \quad \text{and} \quad \begin{cases} \mathrm{H}\dfrac{1}{\sqrt{2}}(|0\rangle + |1\rangle) = |0\rangle\,, \\ \mathrm{H}\dfrac{1}{\sqrt{2}}(|0\rangle - |1\rangle) = |1\rangle\,. \end{cases} \tag{22}$$

Many well known quantum algorithms make heavy use of this operator [1,2], making it is essential to QC. The concept of the phase flip, Z, can be extended to a phase shift of arbitrary angle. The gate is denoted by $P(\phi)$ or sometimes simply ϕ. It is defined as

$$P(\phi) \doteq \begin{pmatrix} 1 & 0 \\ 0 & e^{i\phi} \end{pmatrix}. \tag{23}$$

In terms of the phase shift gate, Z reduces to the case $P(\pi) = Z$. Some other notable cases of the phase shift are the T and S gates, defined by

$$T = P(\pi/4) \doteq \begin{pmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{pmatrix}, \; S = T^2 = P(\pi/2) \doteq \begin{pmatrix} 1 & 0 \\ 0 & e^{i\pi/2} \end{pmatrix}. \tag{24}$$

The T-gate plays an important role in universality as it constitutes a universal set for single-qubit gates together with any gate from the Clifford group. The Clifford group are the set of operators which normalizes the Pauli group, $\mathbf{P}$. Such that for an element V from the Clifford group, $V\mathbf{P}V^\dagger = \mathbf{P}$, where the Pauli group is the group containing the Pauli matrcies together with the multiplicative factors ± 1 and $\pm i$.

All gates so far are single-qubit gates. To achieve full universality it turns out that two-qubit gates are enough. The two-qubit gate most commonly used is the CNOT, which stands for controlled not. The effect is defined as $\texttt{CNOT}\,|x, y\rangle = |x, x \oplus y\rangle$, where $\oplus$ denotes addition modulo 2. The gate takes two inputs, the control, and the target. If the control is $|1\rangle$ then a NOT is applied to the target $|y\rangle \mapsto X\,|y\rangle$, otherwise the gate leaves the inputs unchanged.

2.2.4 Qudit generalizations

Inherently there is nothing special about bits and binary logic, the same goes for qubits. A qudit is a general name for a d-dimensional quantum computational element. The special case of $d = 3$ is called a *qutrit*, $d = 4$ a *ququart* and so on. A larger computational basis could even prove to be better. The information content per unit is higher, N qubits can be reduced down to $\dfrac{N}{\log_2(d)}$ qudits [17]. The qutrit has a reduction factor of ~ 0.64, the ququart 0.5, with less returns for higher dimensions. Due to more information per unit, fewer operations are required to build quantum gates, minimizing the loss of accuracy from operations. The drawback is the increased complexity in the schemes for physical implementation which could come with increased errors. There are already promising results using qudits [11,12,13], and many more are discussed in [14]. The interesting question is if the benefits of qudits can outweigh the extra costs associated with more complex structures.

In the following, let the dimension d be any integer such that $d \geq 3$. Some generalized qudits gates are defined and discussed. The definitions used are the same used in [14]. The effect of NOT is ambiguous when the number of basis states are more than 2. So instead of viewing it as NOT, it can be thought of as a cyclic permutation of the states $|0\rangle, |1\rangle, |2\rangle, \ldots, |d-1\rangle$. The higher dimensional correspondence of X is thus defined as

$$X_d = \begin{pmatrix} 0 & 0 & \ldots & 0 & 1 \\ 1 & 0 & \ldots & 0 & 0 \\ 0 & 1 & \ldots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \ldots & 1 & 0 \end{pmatrix}. \tag{25}$$

The effect works as a permutation of basis states; for $d = 3$, the effect would be $|0\rangle \xrightarrow{X_3} |1\rangle \xrightarrow{X_3} |2\rangle \xrightarrow{X_3} |0\rangle$. The higher dimensional Z-gate Z_d is defined as

$$Z_d = \begin{pmatrix} 1 & 0 & 0 & \ldots & 0 \\ 0 & \omega & 0 & \ldots & 0 \\ 0 & 0 & \omega^2 & \ldots & 0 \\ \vdots & \vdots & \vdots & \ddots & 0 \\ 0 & 0 & 0 & \ldots & \omega^{d-1} \end{pmatrix} \tag{26}$$

with $\omega = e^{2\pi i/d}$ being the dth root of unity. It shifts relative phase of the different basis states a certain amount. The version of the Hadamard gate for qudits, H_d, is

more convenient to define in Dirac notation, and reads

$$H_d = \frac{1}{\sqrt{d}} \sum_{i=0}^{d-1} \omega^{i\,j} \left|i\right\rangle \left\langle j\right|, \; j \in \{0, 1, 2, \ldots, d-1\}. \tag{27}$$

Last let us define the generalized qudit T-gate, it turns out that T_d is only defined for prime number dimensions [18] and reads

$$T_d = \sum_{k=0}^{d-1} \omega^{\nu_k} \left|k\right\rangle \left\langle k\right|. \tag{28}$$

The exponent ν_k can be recursively defined as

$$\begin{cases} \nu_0 = 0, \\ \nu_{k+1} = \nu_k + k(2^{-1}\gamma' k + z') + 2^{-1}z' + \varepsilon' \end{cases} \tag{29}$$

with $\gamma', z', \varepsilon' \in \mathbb{Z}_d$, as shown in [18].

In this report, T_3 with $z' = 1, \gamma' = 2$ and $\varepsilon' = 0$ will be implemented. This gate reads

$$T_3 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & e^{2\pi i/9} & 0 \\ 0 & 0 & e^{-2\pi i/9} \end{pmatrix}. \tag{30}$$

Likewise, as for qubits, single qudit universality can be obtained by a combination of some of these gates. The universal set of gates for the qudit is the generalization of the universal set for the qubit, $\{T_d, H_d\}$. Since the T_d-gate is only defined in prime number dimensions universality cannot be obtained in this manner in non-prime dimensions.

2.2.5 Dark states, bright states and dark paths

Pod-like systems are an important type of quantum system and are used frequently in quantum computing. A pod-like system refers to a type of system with most commonly, but not limited to, one excited state coupled to a number of ground states, with couplings which describes how the states transform, with no additional coupling between the ground states. This section defines some common terms and concepts that will be used in the later sections.

For a given Hamiltonian H, the eigenvalues correspond to the energies of the system. Let us denote a state with zero energy as $|d\rangle$. It holds then that

$$H\left|d\right\rangle = 0\left|d\right\rangle, \tag{31}$$

which is to say that $|d\rangle$ is an eigenstate with eigenvalue 0. We say that $|d\rangle$ is a **dark state** if it does not contain any excited state. $|d\rangle$ forms a 1-dimensional dark subspace, and in general the dimension of the full dark subspace is equal to the number of dark states. Furthermore we say that any state $|b\rangle$ such that

$$\langle d_k | b\rangle = 0 \tag{32}$$

is a **bright state** provided it is orthogonal to all dark states, $|d_k\rangle$, of the system. A pod-like system with n ground states and m excited states will have $n-m$ dark states and m bright states [19]. The Hamiltonian can be rewritten in the *dark-bright* basis (Morris-Shore basis), an equivalent basis spanned by the bright and dark states. Since the dark states are all individually decoupled from the Hamiltonian this new basis can be further split into two subspaces, one containing the bright states and one containing the dark states. This reduction can be useful since it simplifies the dynamics of the system. The dynamics is determined by the Hamiltonian of the system. The part of the time evolution affected by the Hamiltonian, for some ordered basis, is $H_{nm}(t) = \langle n| H(t) |m\rangle$. In the adiabatic case if the system starts in the dark subspace, it will remain there throughout the evolution and the dynamical phase will have no contribution to the total time evolution. In a non-adiabatic case the time evolution of the states is more complex due to the mixing of the eigenstates. So by constructing a Hamiltonian such that all H_{nm} elements are zero there will be no dynamical contribution. Another way to assure no dynamical phase is to evolve the system along a **dark path**, which is a state $|D(t)\rangle$ such that

$$\langle D(t)| H(t) |D(t)\rangle = 0, \, \forall t \in [0, T]. \tag{33}$$

This means that the expectation value of the energy will always be 0, and thus the corresponding dynamical phase will be 0. The dark state in itself satisfies this condition and as such is a trivial dark path. By explicitly finding a dark path the parameters in the Hamiltonian can be reverse-engineered from the Schrödinger equation, enabling more complex Hamiltonians that can fulfill the NHQC criteria.

2.2.6 Holonomic quantum computation

By making use of the geometric phases in periodic quantum system, schemes that perform computation are obtainable. Holonomic Quantum Computation (HQC) makes use of holonomies, which is to what extent parallel transport around some closed-loop fails to preserve the transported data. In the case of quantum mechanics this means that a state, gains an additional geometric phase alongside the dynamical phase. This concept was first introduced by Berry in [5] and thus the phase is usually referred to as the Berry phase, but is more generally called the geometric phase since the phase arises from the geometry rather than the dynamics of the system. The geometric phase exists also in non-adiabatic and non-Abelian contexts, which gives room for more complex structures [6,7,8]. In the adiabatic non-Abelian setting universal computation can be achieved through holonomies by transporting a set of control parameters along a loop(s) in a suitable space [3].

For the following part let us introduce an ordered, once-differentiable, time dependent basis $|1(t)\rangle, |2(t)\rangle, \ldots, |N(t)\rangle$ in some N-dimensional subspace $\mathcal{M}(t)$ of the full system. We require the subspace to cyclic, such that after some time T, $\mathcal{M}(0) = \mathcal{M}(T)$. Thus, for any state at $t = 0$, $|\psi(0)\rangle \in \mathcal{M}(0)$, then $|\psi(T)\rangle = U(T, 0) |\psi(0)\rangle \in \mathcal{M}(0)$. More formally, for the set of projection operators

$$\mathcal{P}(t) = \{|0(t)\rangle\langle 0(t)|, |1(t)\rangle\langle 1(t)|, \ldots, |N(t)\rangle\langle N(t)|\}$$

which project the full system onto $\mathcal{M}$, it holds that

$$\forall\, P(t) \in \mathcal{P}(t),\ P(0) = U(T,0)P(0)U^{\dagger}(T,0).$$

This is to say that the projection operators are transported in a loop with periodicity T. With this in mind we can examine how the time evolution of an arbitrary state $|\psi\rangle \in \mathcal{M}$ behaves (time dependence suppressed for clarity), it can be expanded in terms of the basis states as

$$|\psi\rangle = \sum_{n=1}^{N} c_{:n} \, |n\rangle \, . \tag{34}$$

By taking the time derivative one finds

$$|\dot{\psi}\rangle = \sum_{n=1}^{N} \left[\dot{c}_{:n} \, |n\rangle + c_{:n} |\dot{n}\rangle \right] . \tag{35}$$

By the Schrödinger equation, $|\dot{\psi}\rangle$ can alternatively be written as

$$|\dot{\psi}\rangle = -i \, H \, |\psi\rangle = -i \sum_{n=1}^{N} c_{:n} H \, |n\rangle \, . \tag{36}$$

By equating (35) and (36) and taking the inner product with $\langle m|$, one obtains

$$-\, i c_{mn} \, \langle m| \, H \, |n\rangle = \dot{c}_{mn} + \sum_{n=1}^{N} c_{mn} \, \langle m|\dot{n}\rangle \, , \tag{37}$$

which can be re-organized as

$$\dot{c}_{mn} = -i \left[c_{mn} \, \langle m| \, H \, |n\rangle - \sum_{n=1}^{N} i c_{mn} \, \langle m|\dot{n}\rangle \right] . \tag{38}$$

In matrix form this can be written as

$$\dot{\mathbf{C}} = i \, [\mathbf{A} - \mathbf{H}] \, \mathbf{C}, \tag{39}$$

where $\mathbf{A}_{mn}(t) = i \, \langle m(t)| \, \frac{\mathrm{d}}{\mathrm{d}t} \, |n(t)\rangle$ and $\mathbf{H}_{mn}(t) = \langle m(t)| \, H(t) \, |n(t)\rangle$. The formal solution is

$$U(T,0) = \mathcal{T} \exp \left(i \int_0^T \left[\underbrace{\mathbf{A}(t)}_{\text{Geometric}} - \underbrace{\mathbf{H}(t)}_{\text{Dynamic}} \right] \, dt \right) . \tag{40}$$

This is the non-adiabatic non-Abelian holonomy proposed by Anandan in [7]. Whith such a cyclic system a scheme can be constructed such that the dynamical phase will have no contribution to the time evolution during a full cycle. We can then describe the cycle as a closed loop, $\mathcal{C}$ in the space of N dimensional subspaces of the K dimensional Hilbert space $\mathcal{H}$ of the system, i.e., the Grassmannian $\mathcal{G}(K;N)$. So after one cycle the operator can be written as

$$U(T,0) = \mathcal{T} \exp \left(i \int_0^T \mathbf{A}(t) \, dt \right) = \mathcal{P} \exp \left(i \oint_{\mathcal{C}} \mathbf{A}(s) \, ds \right) = U(\mathcal{C}), \tag{41}$$

where $\mathcal{P}$ denotes path-ordering.

In the adiabatic case the closed loop $\mathcal{C}$ is simply a loop in parameter space, in this case equation 41 is refereed Wilczek-Zee holonomy which can be used to achieve holonomic quantum computation. In the adiabatic setting it is enough for the system to start in a dark sub-space since the states will evolve without mixing. Thus the energy associated with the Hamiltonian will be zero throughout the whole time evolution. Therefore it is possible to construct unitaries by purly geometric means by starting in the correct state [20,21].

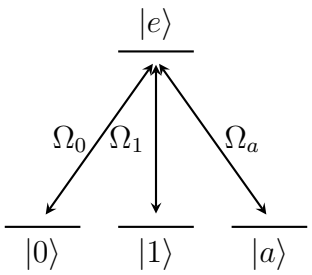

Figure 2: System described by the Hamiltonian in Equation 42.

A well known result that makes use of the Wilczek-Zee holonomy to obtain universal computation was presented by Duan et al. in [21]. By using a tripod system defined by the Hamiltonian

$$H = |e\rangle \left[\Omega_0 \langle 0| + \Omega_1 \langle 1| + \Omega_a \langle a|\right] + \text{ h.c.}, \tag{42}$$

where + h.c. is read as Hermitian conjugate, the system is shown in Figure 2. By manipulating the parameters, this Hamiltonian can be used to create a set of universal gates; $U_1 = e^{i\phi_1|1\rangle\langle 1|} = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\phi_1} \end{pmatrix}, U_2 = e^{i\phi_2\sigma_y} = R_y(\phi_2), U_3 = e^{i\phi_3|11\rangle\langle 11|}$. Let us calculate U_1, which is a phase gate, to see how this is obtained. First the parameters are chosen as $\Omega_0 = 0$, $\Omega_1 = -\Omega \sin\frac{\theta}{2} e^{i\varphi}$, $\Omega_a = \Omega \cos\frac{\theta}{2}$. The Hamiltonian in this case is

$$H = |e\rangle \left[-\Omega \sin\frac{\theta}{2} e^{i\varphi} \langle 1| + \Omega \cos\frac{\theta}{2} \langle a|\right] + \text{ h.c.} \tag{43}$$

Then find a dark state $|d\rangle$ to the Hamiltonian such that $H|d\rangle = 0$. In this state the energy will be zero and by evolving the system adiabatically the energy will remain zero. The dark state is $|d\rangle = \cos\frac{\theta}{2}|1\rangle + \sin\frac{\theta}{2} e^{i\varphi}|a\rangle$. By starting H at $\theta = 0$, then $|d\rangle = |1\rangle$ and evolving the new control parameters in a closed loop in parameter space we find that $U_1 = e^{i\phi_1|1\rangle\langle 1|}$, where ϕ_1 is the enclosed solid angle obtained during the evolution, i.e., $\phi_1 = \iint_{\mathcal{C}} \sin\theta \, d\theta \, d\varphi$. The other single-qubit gate U_2 is obtained in a similar way and U_3 requires some extra work by using detuning to create a two-qubit gate.

Due to the long runtimes associated with adiabatic evolution the state is prone to errors and decoherence from outside factors [4]. As a consequence, in the non-adiabatic case (NHQC), it would be possible to construct faster and more robust gates.

Since the states are mixing, some other way to control U must be engineered. One way this can be achieved is by removing the dependence on the dynamics and energies of the system from the time evolution. This means that we ensure that

$$\langle m(t)| H(t) |n(t)\rangle = 0, \ \forall m, n, t \in [0, T], \tag{44}$$

holds. Thus all terms in the dynamical phase in (40) are zero. This is possible given that there exist a two-pulse non-commuting loop, $\mathcal{C} = \mathcal{C}_1 \cup \mathcal{C}_2$, [4,22], then the unitary will be reduced to

$$U(T, 0) = \sum_{n,m=1}^{N} U_{nm} |n(0)\rangle \langle m(0)| \tag{45}$$

and can be used to perform any arbitrary quantum gate on the computational space.

The idea of NHQC can be extended by instead of explicitly constructing a Hamiltonian with non-dynamical interaction, one can construct explicit dark paths through parameter space that satisfy the NHQC criteria. In [15] a qubit scheme was constructed and experimentally implemented in a trapped $^{171}Yb^{+}$ ion, using a dark path. They conclude that arbitrary NHQC gates can be constructed with better robustness. We will build upon this scheme and create an explicit qutrit example, then outline how it can be generalized to higher dimensions.

2.3 Research Questions

Holonomic quantum computation is an interesting research area due to the potential of high resilience to errors arising from constructing gates from global holonomies. Furthermore non-adiabatic implementations are desirable since shorter run times can decrease decoherence during the time evolution of systems and thus make even sturdier gates. There have already been several holonomic implementations of qubits, both adiabatic and non-adiabatic [3,4] but here we aim at implementing higher dimensional qudits in a non-adiabatic holonomic setting. The project addresses the following research questions:

- **1: How does the NHQC qubit implementation using dark path generalize to higher computational elements?** We are insterested to see how the implementation of the qubit in [15] could be generalized to qudits. This is achieved by finding a way to extend the quantum system in such a way that a qudit can be encoded.

- **2: Can qudit gate fidelity be improved by inclusion of auxiliary states?** The upside of NHQC is the high error resilience in the quantum gates. For a qubit this can further be improved by including an auxiliary state in the system. We want to study if such an inclusion could improve fidelity for qudits as well.

3 Results

3.1 The qutrit

3.1.1 Construction of the system

In [15] a qubit is implemented by modifying the lambda system [19], which is a pod-like system with one excited state and two ground states, adding an auxiliary state, effectively turning it into a tripod. The generalization to implement a qutrit is not trivial but it is clear that one extra ground state is required to make the computational basis larger. Another thing that is needed is to limit the number of dark states to one. The number of dark states is the difference between the number of excited states and the number of ground states [19] (here the auxiliary state is **not** counted as a ground state). Therefore another excited state must be added as well. How these states are coupled is the hard part. We will see that the system given by the Hamiltonian

$$H = \sum_{j=1}^{2}\sum_{i=j}^{3} \omega_{ij} \left|i\right\rangle \left\langle e_j\right| + \frac{\Omega_a(t)}{2} \left|a\right\rangle \left\langle e_2\right| + \text{ h.c.} \tag{46}$$

which is shown in Figure 3, will work. It consists of two excited states, $\left|e_1\right\rangle, \left|e_2\right\rangle$, an auxiliary state $\left|a\right\rangle$ and the ground states $\left|1\right\rangle, \left|2\right\rangle, \left|3\right\rangle$, which make up the computational basis of the qutrit. By changing the basis of the Hamiltonian, it can be rewritten as

$$H_d = \sum_{j=1}^{2} \frac{\Omega_j(t)}{2} e^{-i\phi_j} \left|b_j\right\rangle \left\langle e_j\right| + \frac{\Omega_a(t)}{2} \left|a\right\rangle \left\langle e_2\right| + \text{ h.c.} \tag{47}$$

by a Morris-Shore transformation [23]. This new basis is the dark-bright basis. To find this basis find a dark state to the original Hamiltonian, an eigenstate with eigenvalue 0, $H\left|d\right\rangle = 0$, then find two bright states, orthogonal to both $\left|d\right\rangle$ and each other. This basis can be parametrized with 4 angles, $\theta, \varphi, \chi, \xi$, where the dark state is on the form

$$\left|d\right\rangle = \cos\theta \left|1\right\rangle + e^{i\chi} \sin\theta \cos\varphi \left|2\right\rangle + e^{i\xi} \sin\theta \sin\varphi \left|3\right\rangle, \tag{48}$$

and the two bright states are

$$\begin{aligned} \left|b_1\right\rangle &= N_1 \left(-e^{-i\chi} \sin\theta \cos\varphi \left|1\right\rangle + \cos\theta \left|2\right\rangle\right), \\ \left|b_2\right\rangle &= N_2 \left(\cos\theta \left|1\right\rangle + e^{i\chi} \sin\theta \cos\varphi \left|2\right\rangle + \Lambda \left|3\right\rangle\right), \end{aligned} \tag{49}$$

where N_1, N_2 are normalization factors and Λ is chosen such that $\left\langle d|b_2\right\rangle = 0$.

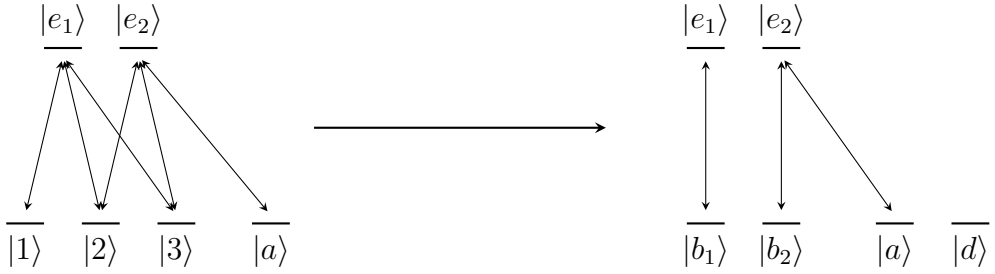

Figure 3: The system given by the Hamiltonian shown in Equation 46 (left) and the transformed system shown in Equation 47 (right).

Explicitly the states in the dark-bright basis are given by

$$\begin{aligned}
|d\rangle &= \cos\theta\,|1\rangle + e^{i\chi}\sin\theta\cos\varphi\,|2\rangle + e^{i\xi}\sin\theta\sin\varphi\,|3\rangle\,,\\
|b_1\rangle &= \frac{1}{\sqrt{1-\sin^2\theta\sin^2\varphi}}\left(-e^{-i\chi}\sin\theta\cos\varphi\,|1\rangle + \cos\theta\,|2\rangle\right),\\
|b_2\rangle &= \frac{1}{\sqrt{1-\sin^2\theta\sin^2\varphi}}\Bigg(\frac{1}{2}\sin 2\theta\sin\varphi\,|1\rangle + \frac{e^{i\chi}}{2}\sin^2\theta\sin 2\varphi\,|2\rangle\\
&\quad + e^{i\xi}(\sin^2\theta\sin^2\varphi - 1)\,|3\rangle\Bigg).
\end{aligned} \tag{50}$$

The parameters ω_{ij} in the original basis can be determined by replacing the states in H_d by their form in the $\{|1\rangle, |2\rangle, |3\rangle\}$ basis.

Now let us introduce two dark paths, $|D_1(t)\rangle, |D_2(t)\rangle$. Along these paths the average energy is always zero $\langle D_i(t)|\, H_d\, |D_i(t)\rangle = 0$, $i = 1, 2$. Thus no dynamical phase is accumulated during the time evolution of these states and therefore follows the conditions required for NHQC.

The following two states, parametrized by two angles $u(t), v(t)$, satisfy the dark path condition:

$$\begin{aligned}
|D_1(t)\rangle &= \cos u e^{-i\phi_1}\,|b_1\rangle + i\sin u\,|e_1\rangle\,,\\
|D_2(t)\rangle &= \cos u\cos v e^{-i\phi_2}\,|b_2\rangle - i\sin u\,|e_2\rangle - \cos u\sin v\,|a\rangle\,.
\end{aligned} \tag{51}$$

It can easily be verified that $\langle D_i(t)|\, H_d\, |D_j(t)\rangle = 0, i, j = 1, 2$. The angles can be chosen so as to satisfy the constraint $|D_i(0)\rangle\langle D_i(0)| = |D_i(T)\rangle\langle D_i(T)|$, $i = 1, 2$. This can be achieved by choosing $u(0) = u(T) = v(0) = v(T) = 0$. A valid choice is $u(t) = \frac{\pi}{2}\sin^2\frac{\pi t}{T}$ and $v(t) = \eta\,[1 - \cos u(t)]$, as for the qubit case [15]. η represents the coupling strength to the auxiliary state $|a\rangle$ and the system reverts into a tri-pod structure with two excited states when $\eta = 0$. Unless mentioned otherwise $\eta = 4.0$, which is the optimal choice for the qubit and shall be used here as well [15]. Each dark path starts in the respective bright state and travels along a curve and then returns to the bright state. Using the Schrödinger equation one can relate the dark path to the Hamiltonian,

$$i\frac{\partial}{\partial t}\,|D_i(t)\rangle = H_d\,|D_i(t)\rangle\,, \tag{52}$$

and thusly one can reverse engineer the time dependent parameters $\Omega_i(t)$ by matching the factors in front of each state. A calculation yields

$$\begin{aligned}
\Omega_1(t) &= -2\dot{u},\\
\Omega_2(t) &= 2\,(\dot{v}\cot u\sin v + \dot{u}\cos v)\,,\\
\Omega_a(t) &= 2\,(\dot{v}\cot u\cos v - \dot{u}\sin v)\,.
\end{aligned} \tag{53}$$

To construct a quantum gate, we make use of the method of multi-pulse single-loops [22], for which the part of the time evolution operator that acts on the computational space is

$$\begin{aligned}
U_1 &= |d\rangle\langle d| - i\,|e_1\rangle\langle b_1| - i\,|e_2\rangle\langle b_2|\,,\ \phi_1 = \phi_2 = 0,\\
U_2 &= |d\rangle\langle d| + ie^{i\gamma_1}\,|b_1\rangle\langle e_1| + ie^{i\gamma_2}\,|b_2\rangle\langle e_2|\,,\ \phi_1 = -\gamma_1,\ \phi_2 = -\gamma_2
\end{aligned} \tag{54}$$

so the operator for one full loop is

$$U = U_2 U_1 = |d\rangle \langle d| + e^{i\gamma_1} |b_1\rangle \langle b_1| + e^{i\gamma_2} |b_2\rangle \langle b_2| . \tag{55}$$

This transformation can be parametrized by 6 real parameters,$\chi, \xi, \theta, \varphi, \gamma_1, \gamma_2$, however it is not enough to construct all gates. For example X_3 requires 2 loops. This is since one loop does not cover all degrees of freedom. The reason for this is elaborated on later in Section 3.2. The full gate is given by repeating U with another set of parameters:

$$\mathbb{U} = U(\chi', \xi', \theta', \varphi', \gamma_1', \gamma_2') U(\chi, \xi, \theta, \varphi, \gamma_1, \gamma_2). \tag{56}$$

Now the problem is only a matter of finding all parameters to replicate the desired gate. This is non-trivial since it includes solving a system of 9 non-linear algebraic equations containing 12 variables for a gate requiring two loops. Some gates that only require a single loop can with some work be found analytically, but the problem quickly becomes complex.

3.1.2 Implementation

Implementation of the qutrit for simulation purposes can give valuable insights of the system. By starting from an initial state and expressing it in terms of the dark paths, the evolution of the state is given by evolving the explicit expressions of the dark paths. Let $|\psi(0)\rangle = (a, b, c)^T$ be the initial state in the computational basis $\{|1\rangle, |2\rangle, |3\rangle\}$. Then the following equation can be formulated, for some coefficients f_i,

$$\begin{pmatrix} a \\ b \\ c \end{pmatrix} = f_0 |d\rangle + f_1 |b_1\rangle + f_2 |b_2\rangle \tag{57}$$

since at time $t = 0$ the dark paths starts at the corresponding bright state. Now expand the dark state and bright state in terms of the original basis. Doing this one can obtain the relation

$$\begin{pmatrix} f_0 \\ f_1 \\ f_2 \end{pmatrix} = \begin{pmatrix} c_1 & -N_1 c_2 & N_2 c_1 \\ c_2 & N_1 c_1 & N_2 c_2 \\ c_3 & 0 & N_2 \Lambda \end{pmatrix}^{-1} \begin{pmatrix} a \\ b \\ c \end{pmatrix} . \tag{58}$$

So by choosing an initial state, the coefficients can be obtained. Then the system can be solved for any time $t \in T$ using

$$|\psi(t)\rangle = f_0 |d\rangle + f_1 |D_1(t)\rangle + f_2 |D_2(t)\rangle , t \in [0, T]. \tag{59}$$

To see the effect of the gates, the probability of finding the system in the standard computational basis states can be plotted against time to see how the gate transform the state from initial to final state, shown in Figure 4 and 5 for the X_3 and H_3 gates respectively. This is simply done in any computational software by sampling Eq. 59 over the time interval. All plots and calculations in this project are done using python 3.8.8.

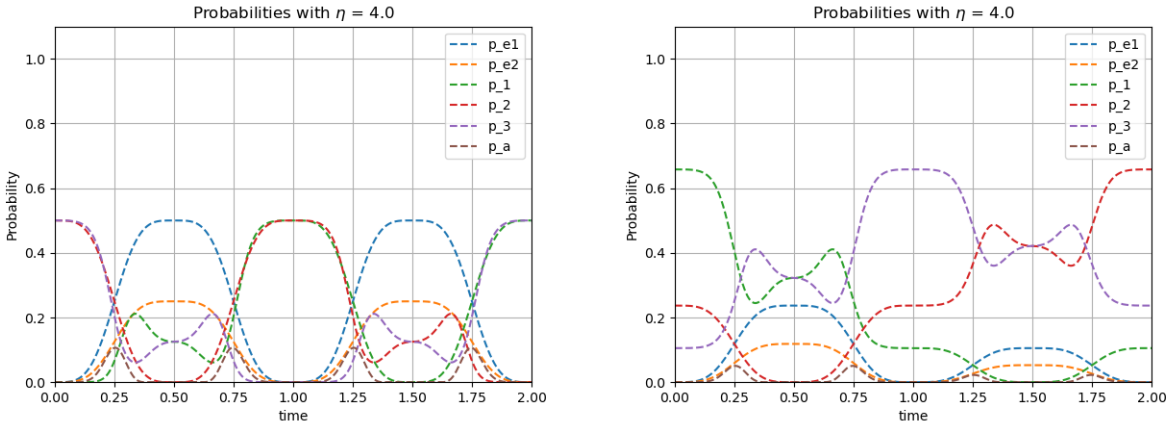

Figure 4: The effect of the X_3-gate on the initial states $\frac{1}{\sqrt{2}}[0,1,1]$ (left) and $\frac{1}{\sqrt{38}}[5,3,2]$ (right). Note that since the plot shows the probabilities, phases cannot be seen in the plot.

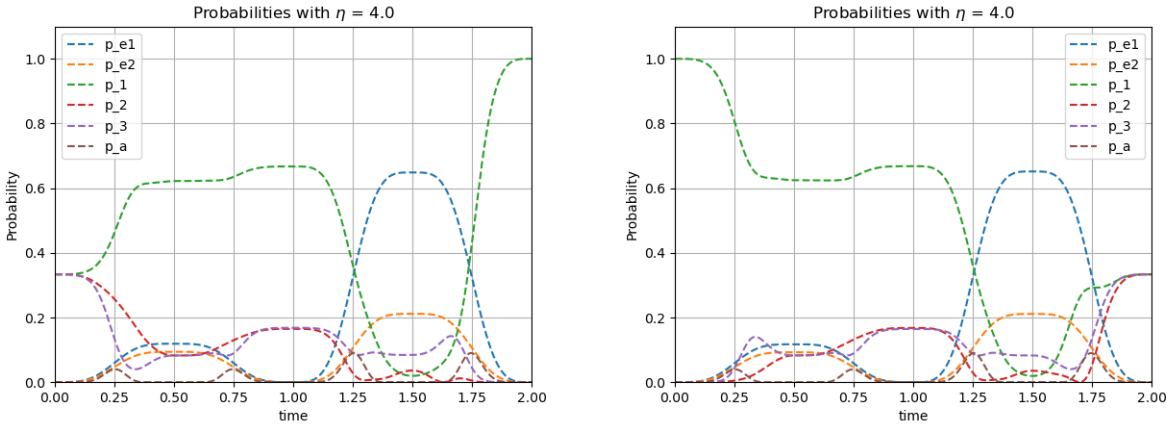

Figure 5: The effect of the H_3-gate on the initial states $\frac{1}{\sqrt{3}}[1,1,1]$ (left) and $[1,0,0]$ (right). Note that since the plot shows the probabilities, phases cannot be seen in the plot.

3.1.3 Important gates, universality and parameter selection

By specifying the parameters of U the qutrit gates from Section 2 can be constructed. A selection of important gates are obtained by the following;

$$
\begin{aligned}
X_3 &= U(0,0,\frac{\pi}{4},\frac{\pi}{2},0,\pi) \times U(0,0,\frac{\pi}{2},\frac{\pi}{4},0,\pi) = \begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix} \\
Z_3 &= U(0,0,0,0,\frac{2\pi}{3},\frac{4\pi}{3}) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & e^{\frac{2\pi i}{3}} & 0 \\ 0 & 0 & e^{\frac{4\pi i}{3}} \end{pmatrix} \\
T_3 &= U(0,0,0,0,\frac{2\pi}{9},\frac{-2\pi}{9}) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & e^{\frac{2\pi i}{9}} & 0 \\ 0 & 0 & e^{\frac{-2\pi i}{9}} \end{pmatrix} \\
\mathrm{H}_3 &= U(6.41\cdot 10^{-4}, 6.56\cdot 10^{-4}, 0.48, 0.79, 1.58, 1.56) \\
&\times U(9.81\cdot 10^{-3}, 0.00, 1.187, 2.15, 0.00, 1.57) \\
&\approx \frac{1}{\sqrt{3}} \begin{pmatrix} 1 & 1 & 1 \\ 1 & e^{\frac{2\pi i}{3}} & e^{\frac{4\pi i}{3}} \\ 1 & e^{\frac{4\pi i}{3}} & e^{\frac{2\pi i}{3}} \end{pmatrix}.
\end{aligned}
\tag{60}
$$

Note that the choice of parameters is not unique and there are multiple ways to create the same unitary. These gates are enough to achieve single-qutrit universality as discussed in Section 2. To achieve full universality an additional two-qutrit gate is needed. A common choice for qubits is the `CNOT` gate [14].

The parameters for any given gate can be specified by solving the non-linear system of equations obtained by setting the matrix representation of the gate equal to Equation 55 or 56. It can be found that some gates such as the X_3-gate cannot be replicated by only one loop but will require at least 2 loops. The physical effect of this is that during the first cycle a pair of states are swapped, and during the second cycle another pair is swapped. This can be seen in Figure 4. For the diagonal gates the analytical solution can be easily found by fixing $\theta = \varphi = \chi = \xi = 0$. The basis states reduce to $|d\rangle = |1\rangle, |b_1\rangle = |2\rangle, |b_2\rangle = -|3\rangle$. Then all diagonal unitaries can be specified by γ_1 and γ_2 up to a phase factor,

$$
U(0,0,0,0,\gamma_1,\gamma_2) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & e^{i\gamma_1} & 0 \\ 0 & 0 & e^{i\gamma_2} \end{pmatrix}. \tag{61}
$$

Analytical solutions are not always easy to find. The H_3-gate gives rise to a system of equations that is non-trivial to solve. In that case an approximative gate $\tilde{U}$ can be found by numerical optimization of the expression $\min||U-\tilde{U}||_F$, where $||\cdot||_F$ is the Frobenius matrix norm given by $||A||_F = \sqrt{\mathrm{Tr}\left[A^\dagger A\right]}$. Since $\tilde{U} \approx U$ the approximated gate will have an effect close to that of the exact gate.

To assess the robustness of the gate the fidelity metric, F, is used, the metric measures how close two quantum states are to each other. Here we are only considering pure states, then the fidelity is: $F(|\psi\rangle, |\varphi\rangle) = |\langle\psi|\varphi\rangle|$. The fidelity is averaged by

sampling initial states and letting them evolve with time by numerically solving the Schrödinger Equation using the SciPy implementation of backwards differentiation [24]. Then we introduce small shifts, $\Omega \mapsto \Omega(1+\delta)$, for all Ω_i pulses in Equation 47 and compare to the exact solution obtained by multiplying the gate with the initial state. The calculated fidelities are shown in Figure 6. In the figures it can be seen that the coupling with the auxiliary state is more resilient to errors in the parameters than the original NHQC scheme. This is very similar to the results in 2 dimensions from [15], which suggests an improvement of robustness even in higher dimensions.

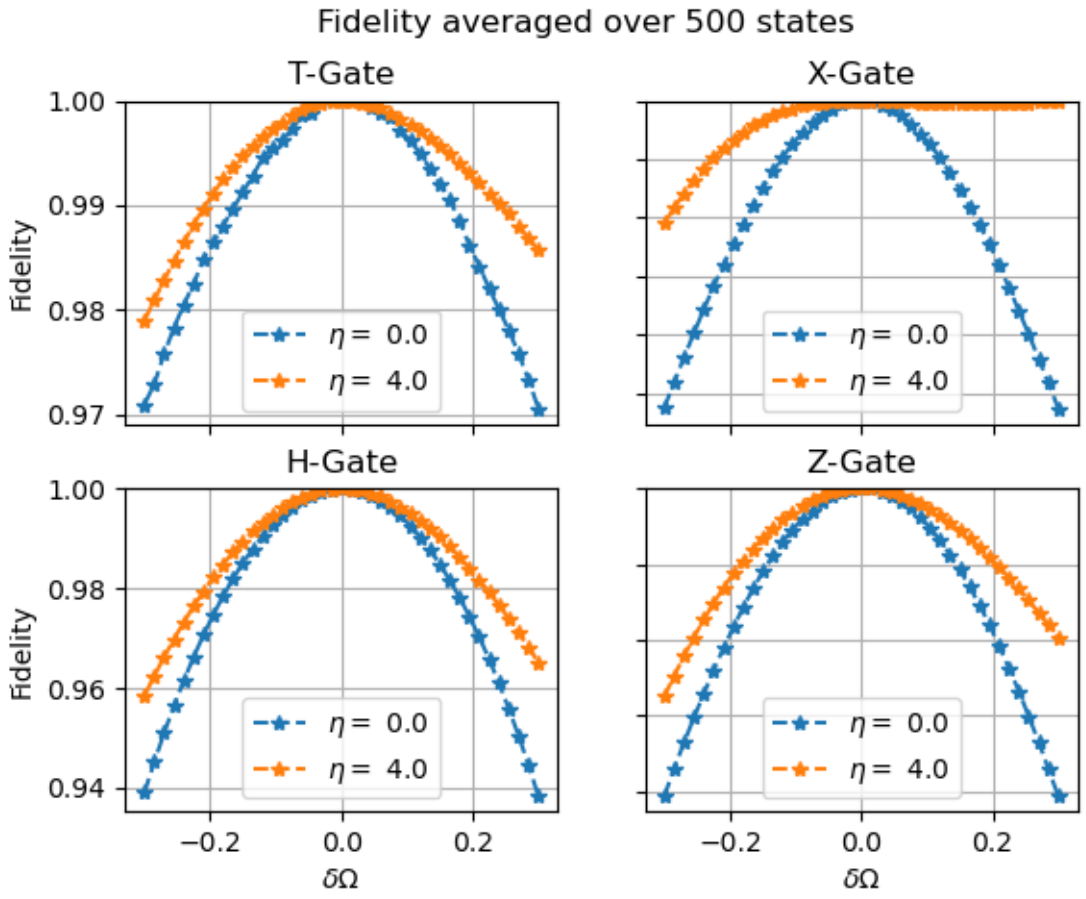

Figure 6: Robustness test, average fidelity of the T_3, X_3, Z_3, and H_3 gates. The averages are calculated by sampling over 500 randomized initial states with a perturbation of the Ω-pulses, $\Omega \mapsto \Omega(1+\delta)$.

3.2 Generalization

In the generalization n will be used instead of the more common d to refer to the dimension of a qudit. This is to avoid confusions with the dark states. To generalize the scheme for qudits with arbitrary dimension, n, the idea from the qutrit case can be extended: n ground states, $m = n - 1$ excited states and 1 auxiliary state. The dimension of the Hilbert space is $n + m + 1 = 2n + 1 - 1 = 2n$. Once again the couplings are non-trivial and will be somewhat intricate, but the couplings given by the Hamiltonian in Equation 62, will do the trick:

$$H = \sum_{j=1}^{m} \sum_{i=j}^{n} \omega_{ij} \left|i\right\rangle \left\langle e_j\right| + \frac{\Omega_a(t)}{2} \left|a\right\rangle \left\langle e_m\right| + \text{ h.c.} \tag{62}$$

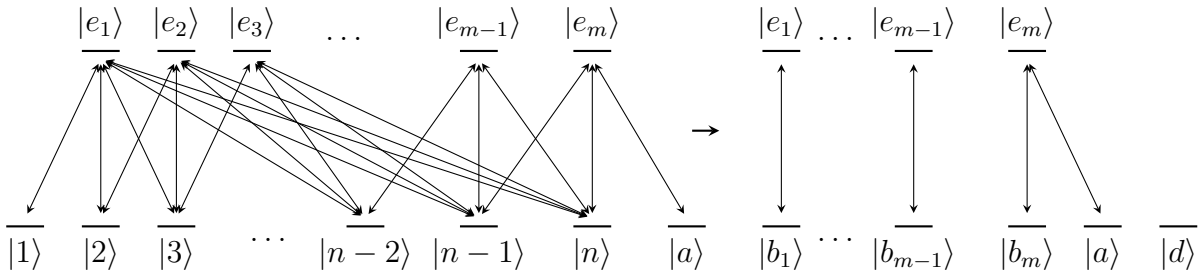

Figure 7: The system given by the Hamiltonian shown in Equation 62 (left) and the transformed system from Equation 69 (right).

The general system for a qudit is given by m excited states $|e_i\rangle$, $i = 0, 1, \ldots, m$, n ground states labeled $|i\rangle$, $i = 1, 2, \ldots, n$ and one auxiliary state $|a\rangle$. The number of states always satisfies that $n-m = 1$ to limit the number of dark states to one [19]. The transitions occur only between excited states and ground states. The couplings follow a pattern; the first excited state e_1 is connected to all ground states, e_2 is connected to all ground states except $|1\rangle$, and so on. In general, the excited state $|e_i\rangle$ is connected to the $(n-i+1)$ ground states with the largest indices. This holds for all excited states except the one with largest index, $i = m$, the excited state $|e_m\rangle$ is connected to the two highest labeled ground states and the auxiliary state $|a\rangle$. See Equation 62 and Figure 7 for a clarification. From the standard basis $\{e_1, e_2, \ldots, e_m, 1, 2, \ldots, n, a\}$, it is possible to change to the dark-bright basis given by $\{e_1, e_2, \ldots, e_m, b_1, b_2, \ldots, b_m, a, d\}$, in which the system is simpler to study. Assume there exists a dark state on the form

$$|d\rangle = c_1 \left|1\right\rangle + c_2 \left|2\right\rangle + c_3 \left|3\right\rangle \cdots + c_n \left|n\right\rangle , \; |c_1|^2 + |c_2|^2 + \cdots + |c_n|^2 = 1. \tag{63}$$

From this dark state one could define $n - 1 = m$ bright states. Starting from $|b_1\rangle$, construct it such that it will be orthogonal to the dark state and all other bright states,

$$|b_1\rangle = N_1 \left(-c_2^* \left|1\right\rangle + c_1^* \left|2\right\rangle\right) \tag{64}$$

with N_1 being a normalization factor. Then choose additional bright states on the form

$$\begin{aligned}
|b_2\rangle &= N_2 \left(c_1 |1\rangle + c_2 |2\rangle + \Lambda_3 |3\rangle\right), \\
|b_3\rangle &= N_3 \left(c_1 |1\rangle + c_2 |2\rangle + c_3 |3\rangle + \Lambda_4 |4\rangle\right), \\
&\vdots \\
|b_{m-1}\rangle &= N_{m-1} \left(c_1 |1\rangle + c_2 |2\rangle + c_3 |3\rangle + \cdots + c_{m-1} |m-1\rangle + \Lambda_m |m\rangle\right), \\
|b_m\rangle &= N_m \left(c_1 |1\rangle + c_2 |2\rangle + c_3 |3\rangle + \cdots + c_m |m\rangle + \Lambda_{m+1} |m+1\rangle\right).
\end{aligned} \tag{65}$$

By construction, $|b_1\rangle$ is orthogonal to the dark state and all other bright states. For $k \geq 2$, $|b_k\rangle$ is $(k{+}1)$-dimensional, so it consists of $k{+}1$ basis-kets, where the coefficient, Λ_{k+1} in front of the last ket is chosen such that $|b_k\rangle$ is orthogonal the dark state, $|d\rangle$. This will in turn make $|b_k\rangle$ orthonormal to any $|b_{>k}\rangle$ as they have the same states and coefficients as $|d\rangle$ for all the states involved in the inner product $\langle b_{>k}|b_k\rangle = \langle d|b_k\rangle$. Therefore, by choosing Λ_k such that these inner products are zero, the construction will results in orthonormal states, $\langle b_i|b_j\rangle = \delta_{ij}$. Explicitly for state $|b_k\rangle$, $k \geq 2$, the coefficient Λ_{k+1} will be given by

$$\Lambda_{k+1} = -\frac{1}{c_{k+1}^*} \sum_{l=1}^{k} |c_l|^2 \tag{66}$$

and the normalization N_k is given by

$$N_k = \left(\sum_{l=1}^{k} |c_l|^2 + |\Lambda_{k+1}|^2\right)^{-1/2} = \left(\sum_{l=1}^{k} |c_l|^2 + \left|-\frac{1}{c_{k+1}^*} \sum_{l=1}^{k} |c_l|^2\right|^2\right)^{-1/2}. \tag{67}$$

The coefficients c_i can be parametrized by the Euclidean components of the radius of the unit-n-sphere with an added phase factor.

$$\begin{aligned}
c_1 &= \cos(\varphi_1), \\
c_2 &= e^{i\theta_1} \sin(\varphi_1) \cos(\varphi_2), \\
c_3 &= e^{i\theta_2} \sin(\varphi_1) \sin(\varphi_2) \cos(\varphi_3), \\
&\vdots \\
c_{n-1} &= e^{i\theta_{n-1}} \sin(\varphi_1) \ldots \sin(\varphi_{n-2}) \cos(\varphi_{n-1}), \\
c_n &= e^{i\theta_n} \sin(\varphi_1) \ldots \sin(\varphi_{n-2}) \sin(\varphi_{n-1}).
\end{aligned} \tag{68}$$

c_1 does not need a phase factor since the overall phase of a state is non-measurable and can be chosen such that the first phase factor will cancel out.

In the dark-bright basis, the Hamiltonian can be written as

$$H_d = \sum_{i=1}^{m} \frac{\Omega_i(t)}{2} e^{-i\phi_i} |b_i\rangle \langle e_i| + \frac{\Omega_a(t)}{2} |a\rangle \langle e_m| + \text{ h.c.} \tag{69}$$

with Ω_i being real-valued time dependent parameters and the ϕ_i time independent phase factors. The explicit basis transformation $H \mapsto H_d$ is not that important but could be obtained by expanding the bright state kets into the standard basis from (65)

With this Hamiltonian m independent dark paths can be constructed, and must satisfy $\langle D_i(t)| H_d |D_i(t)\rangle = 0, i = 1, 2, \ldots, m$ and $\langle D_i(t)|D_j(t)\rangle = \delta_{ij}$. The dark paths can be parametrized by two functions $u(t), v(t)$ that satisfy the conditions $u(0) = v(0) = u(T) = v(T) = 0$ and will have the form

$$\begin{aligned} |D_i(t)\rangle &= \cos u e^{-i\phi_i} |b_i\rangle + i \sin u |e_i\rangle \,,\ i = 1, 2, \ldots, m-1, \\ |D_m(t)\rangle &= \cos u \cos v e^{-i\phi_n} |b_m\rangle - i \sin u |e_m\rangle - \cos u \sin v |a\rangle \,. \end{aligned} \tag{70}$$

The dark paths start in the bright state and travel along a curve where the expectation value of the energy is constantly 0 and can thus be used non-adiabatically.

By using these states one can reverse engineer the Hamiltonian using the Schrödinger equation to determine Ω_i and Ω_a since

$$i\frac{\partial}{\partial t} |D_i(t)\rangle = H_d |D_i(t)\rangle \,,\ i = 1, 2, \ldots, m, \tag{71}$$

a calculation yields

$$\begin{aligned} \Omega_1(t) &= -2\dot{u}, \\ \Omega_2(t) &= -2\dot{u}, \\ &\vdots \\ \Omega_{m-1}(t) &= -2\dot{u}, \\ \Omega_m(t) &= 2\left(\dot{v}\cot u \sin v + \dot{u}\cos v\right), \\ \Omega_a(t) &= 2\left(\dot{v}\cot u \cos v - \dot{u}\sin v\right). \end{aligned} \tag{72}$$

The time evolution is split into k loops, each loop with two pulses, $0 \longrightarrow T/2$ and $T/2 \longrightarrow T$. The relevant part of the time evolution operator for one loop is

$$U_1(T/2, 0) = |d\rangle \langle d| - i \sum_{i=1}^{m} |e_i\rangle \langle b_i| \,,\ \phi_i = 0, i = 1, 2, \ldots, n, \tag{73}$$

and

$$U_2(T, T/2) = |d\rangle \langle d| + i \sum_{i=1}^{m} e^{i\gamma_i} |b_i\rangle \langle e_i| \,,\ \phi_i = 0, i = 1, 2, \ldots, n. \tag{74}$$

The full operator for one loop is then given by

$$U(T, 0) = U_2 U_1 = |d\rangle \langle d| + \sum_{i=1}^{m} e^{i\gamma_i} |b_i\rangle \langle b_i| \,. \tag{75}$$

It is clear that U is unitary in the subspace $\{d, b_1, b_2, \ldots, b_n\}$. The unitary is parametrized by $3m = 3(n-1)$ parameters for $n \geq 2$,

$$U(\varphi_1, \ldots, \varphi_m, \theta_1, \ldots, \theta_m, \gamma_1, \ldots, \gamma_m). \tag{76}$$

Applying the unitary with different parameters in sequence up to k times is enough to create any desirable gate $\mathbb{U} = U^k$. The unitary is controlled by $3(n-1)$ parameters

while n dimensional qudit has more degrees of freedom than covered with a single loop.

The qudit state space of dimension n is equivalent to that of the special unitary group, $SU(n)$, which has dimensionality $\dim(SU(n)) = n^2 - 1$. To cover all degrees of freedom, k must satisfy $3(n-1)k \geq n^2-1$, which require $k \geq \frac{n+1}{3}$. Thus the number of loops needed to create any unitary scales linearly at worst since some gates can be created with fewer loops, see Table 2.

In the case of equality $k = \frac{n+1}{3}$, when $n = 3j + 2$, $j \in \mathbb{N}$, the fewest amount of loops per dimension is achieved and could potentially be more efficient carriers of information since the same number of loops must be carried out while higher dimension has higher information capacity.

n	$3(n-1)$	n^2-1	k	'='
2	3	3	1	✓
3	6	8	2	
4	9	15	2	
5	12	24	2	✓
6	15	35	3	
7	18	48	3	
8	21	63	3	✓
9	24	80	4	
10	27	99	4	
11	30	120	4	✓
12	33	143	5	

Table 2: Table for some dimensions.

The scaling of the qudit gate is linear at worst both in the number of loops and in the number of parameters needed for control. In fact, any diagonal gate only requires one loop: by setting $\varphi_1 = \cdots = \varphi_m = \theta_1 = \cdots = \theta_m = 0$ the unitary reduces to the form

$$U(0, \ldots, 0, \gamma_1, \ldots, \gamma_m) = |1\rangle \langle 1| + \sum_{k=2}^{n} e^{i\gamma_k} |k\rangle \langle k| . \tag{77}$$

The effect of this variable choice makes $c_1 = 1, c_{i\neq 1} = 0$. By (65) the only thing not clear is how this affects the Λ coefficients. For the state $|b_k\rangle$, $k \geq 3$, fixing $c_1 = 1$ and letting all other states approach 0 yields:

$$\begin{aligned}
\lim_{c_{i\neq 1}\to 0} |b_k\rangle &= \lim_{c_{i\neq 1}\to 0} \frac{1}{N_k} \Lambda_{k+1} |k+1\rangle \\
&= \lim_{c_{i\neq 1}\to 0} \left(1 + \left|-\frac{1}{c_{k+1}^*}\right|^2\right)^{-1/2} \left(-\frac{1}{c_{k+1}^*}\right) |k+1\rangle \\
&= \lim_{c_{i\neq 1}\to 0} \left|-\frac{1}{c_{k+1}^*}\right|^{-1} \left(-\frac{1}{c_{k+1}^*}\right) |k+1\rangle \\
&= -|k+1\rangle .
\end{aligned} \tag{78}$$

Therefore $|d\rangle = |1\rangle, |b_1\rangle = |2\rangle, |b_2\rangle = -|3\rangle, |b_3\rangle = -|4\rangle, \ldots, |b_m\rangle = -|m+1\rangle$, and thus the unitary will take the form (77) so any diagonal unitary can be created up to an overall phase. Thus, any phase flip gate, such as Z_n, can be constructed in any dimension using only one loop. The gates essential for single-qudit universality are $\{T_n, \mathrm{H}_n\}$, where T_n can be obtained by a single loop. Thus, the Hadamard gate will be the most costly part of the scheme as it requires at most $k = \frac{n+1}{3}$ loops in n dimensions. Since the T_n gate only is defined for prime n the dimensions of greatest interest will be the prime dimensions where $3(n-1)k = n^2 - 1$. The information content in these dimensions seems to be the efficient since they carry the same information with fewest number of loops where universality could be implemented. The first few of these perfect-prime dimensions are $n = 5, 11, 17, 23, \ldots$. It has already been shown that fault tolerant QC is implementable in odd prime dimensions [25], so a high fidelity controllable scheme in these dimensions would be desirable.

The quantum gate $\mathbb{U}$ can be formulated as a linear combination of the dark state and dark paths just as in the qutrit case,

$$|\psi(t)\rangle = f_0 |d\rangle + \sum_{i=1}^{m} f_i |D_i(t)\rangle, t \in [0, T], \tag{79}$$

where the coefficients can be solved for by choosing an initial state $|\psi(0)\rangle$. This corresponds to one loop, by using $|\psi(T)\rangle$ as an initial state for the next loop. By iterating this method one can simulate $\mathbb{U} = U^k$ where each loop U has a different set of parameters. In theory this can be implemented and simulated for any qudit dimension, just as for the case of the qutrit.

4 Conclusions and outlook

We have shown how to explicitly create a quantum mechanical system, which could be used to emulate a qutrit and corresponding single-qutrit gates. This is done by expanding the dark path qubit scheme [15] into a higher dimension. We have shown how it will generalize in the qudit case and using auxiliary states to improve the robustness of the gates. Universality for the qutrit can be obtained using the Hadamard and T gates in 3 dimensions. The qutrit gates have a high fidelity and their robustness is improved by the inclusion of the auxiliary state in a similar way as for the qubit [15], which suggests that this method can be beneficial for higher dimensional qudits to improve robustness. In the general case for the qudit we have also shown how any dimensional single-qudit diagonal unitary could be created by a single multi-pulse loop in parameter space and that non-diagonal unitaries scale linearly at worst in the number of loops and parameters required for control of each loop scale linearly. The possibility that the scheme expands efficiently into certain prime dimension have been discussed.

One of the greatest challenges of this project was how to choose parameters to replicate the unitary gates. Analytical construction of any diagonal unitary is solved, but there are many other gates that would be interesting to study in more detail. Firstly, taking a deeper look at how imposing different constraints on the parameters will impact the possible constructible gates. As well as doing it the other way around, which constraints are imposed on the parameters if one wants to construct a gate with a certain type of effect. Secondly, since some gates require more parameter space loops it raises the question of how the number of loops impacts the degrees of freedom of the final gate. If it would be possible to find approximate gate that would require fewer loops it would be interesting to see if scheme efficiency would be more valuable than gate accuracy for some cases. This problem is very interesting since reduction of loops decreases the number of parameters needed for control, fewer loops imply less error and make the scheme less costly to run. A direct improvement to the scheme presented in this thesis would be to find a more efficient way to create the Hadamard gate, since it together with the T-gate constitutes a universal set. By finding an efficient implementation of the Hadamard gate the whole scheme could be improved.

For these stated questions, the usage of numerical optimization poses interesting problems to study. Will the numerically optimized gates perform worse than analytical ones, and are there any deeper problems associated with this method? Numerical optimization in the high dimensional parameter spaces present in these system is most likely not well-behaved and may contain a lot of false minima. There might be some structure related to the parametrization that could be used to study the parameter space to gain insight into how to optimally solve the problem.

Another obvious concern that could be addressed is the lack of multi-qudit gates. Entanglement is one of the fundamentals of quantum computing and extending the scheme to include these would be an obvious improvement. This could maybe be achieved through some detuning or by some more explicit physical implementation similar to the qubit case in [15].

Lastly, we have shown how to construct the qudit in any dimension but only analyzed and simulated the qutrit. With the generalization in hand, writing a script which

could apply the scheme to generate qudit gates of arbitrary dimension seems quite possible. Studying the larger prime dimensions to compare efficiency and check fidelity of the gates would be interesting to look into more.

Overall many aspects would be interesting to delve deeper into, and I am certain that multiple improvements to this scheme would be possible.

References

[1] P. W. Shor, Algorithms for quantum computation: discrete logarithms and factoring in *35th Proceedings Annual Symposium on Foundations of Computer Science*, pp. 124-134 (1994).

[2] L. K. Grover, Quantum Mechanics Helps in Searching for a Needle in a Haystack, Phys. Rev. Lett. **79**, 325 (1997).

[3] P. Zanardi and M. Rasetti, Holonomic quantum computation, Phys. Lett. A **264**, 94 (1999).

[4] E. Sjöqvist, D. M. Tong, L. M. Andersson, B. Hessmo, M. Johansson and K. Singh, Non-adiabatic holonomic quantum computation, New J. Phys. **14**, 103035 (2012).

[5] M. V. Berry, Quantal phase factors accompanying adiabatic changings, Proc. Roy. Soc. London Ser. A **392**, 45 (1984).

[6] Y. Aharonov and J. Anandan, Phase change during cyclic quantum evolution, Phys. Rev. Lett. **58**, 1593 (1987).

[7] J. Anandan, Non-adiabatic non-abelian geometric phase, Phys. Lett. A **133**, 171 (1988).

[8] F. Wilczek and A. Zee, Appearance of gauge structure in simple dynamical systems, Phys. Rev. Lett. **52**, 2111 (1984).

[9] M. Glusker, D. M. Hogan and P. Vass, The ternary calculating machine of Thomas Fowler, IEEE Annals of the History of Computing **27**, 4 (2005).

[10] N. P. Brusentsov and J. Ramil Alvarez, Ternary Computers: The Setun and the Setun 70, **357**, 74 (2011).

[11] B. Li and Z. H. Yu and S. M. Fei, Geometry of quantum computation with qutrits, Scientific Reports **3**, 2594 (2013).

[12] H. Lu, Z. Hu, M. S. Alshaykh, A. J. Moore, Y. Wang, P. Imany, A. M. Weiner and S. Kais, Quantum Phase Estimation with Time-Frequency Qudits in a Single Photon, Advanced quantum technologies **3**, 1900074 (2020).

[13] M. Luo and X. Wang, Universal quantum computation with qudits, Science China Physics, Mechanics & Astronomy **57**, 1712 (2014).

[14] Y. Wang, Z. Hu, B. C. Sanders and S. Kais, Qudits and High-Dimensional Quantum Computing, Frontiers in Physics **8**, 589504 (2020).

[15] M. Z. Ai, S. Li, R. He, Z. Y. Xue, J. M. Cui, Y. F. Huang, C. F. Li and G. C. Guo, Experimental Realization of Nonadiabatic Holonomic Single-Qubit Quantum Gates with Two Dark Paths in a Trapped Ion, accepted in Fundamental Research (2021); arXiv:2101.07483

[16] J. J. Sakurai and J. Napolitano, Modern quantum mechanics; 2nd ed, San Francisco, CA: Addison-Wesley, (2011).

[17] A. Pavlidis and E. Floratos, Quantum-Fourier-transform-based quantum arithmetic with qudits, Phys. Rev. A **103**, 032417 (2021).

[18] M. Howard and J. Vala, Qudit versions of the qubit gate, Phys. Rev. A **86**, 022316 (2012).

[19] V. O. Shkolnikov and G. Burkard, Effective Hamiltonian theory of the geometric evolution of quantum systems, Phys. Rev. A **101**, 042101 (2020).

[20] M. Born and V. Fock, Beweis des Adiabatensatzes, Z. Phys. **51**, 165 (1928).

[21] L.-M. Duan, J. I. Cirac and P. Zoller, Geometric Manipulation of Trapped Ion for Quantum Computations, Science **292**, 1695 (2001).

[22] E. Herterich and E. Sjöqvist, Single-loop multiple-pulse nonadiabatic holonomic quantum gates, Phys. Rev. A **94**, 052310 (2016).

[23] J. R. Morris and B. W. Shore, Reduction of degenerate two-level excitation to independent two-state system, Phys. Rev. A **27**, 906 (1983).

[24] L. F. Shampine and M. W. Reichelt, The MATLAB ODE Suite, SIAM journal on scientific computing **18**, 1 (1997).

[25] E. T. Campbell, H. Anwar and D. E. Browne, Magic-State Distillation in All Prime Dimensions Using Quantum Reed-Muller Codes, Phys. Rev. X **2**, 041021 (2012).

A Code

A.1 Script A - Simulate Dark path and Unitary

```
import numpy as np
import matplotlib.pyplot as plt
import itertools
from scipy.optimize import minimize

T = 1.0 # Period of one loop
t = np.linspace(0, T, 1000)
eta = 4.0 # Coupling "strength" to auxilary state

# Basis vectors
kete1 = np.array([1, 0, 0, 0, 0, 0])
kete2 = np.array([0, 1, 0, 0, 0, 0])
ket1 = np.array([0, 0, 1, 0, 0, 0])
ket2 = np.array([0, 0, 0, 1, 0, 0])
ket3 = np.array([0, 0, 0, 0, 1, 0])
ket4 = np.array([0, 0, 0, 0, 0, 1])

# Parametrizes the dark paths
def u(t):
    return (np.pi/2)*(np.sin(np.pi*t/T))**2
def v(t):
    return eta*(1 - np.cos(u(t)))

def D1(t,phi1,b1):
    if t¡T/2.0:
        phi1 = 0
    return np.exp(-1j*phi1)*np.cos(u(t))*b1 + 1j*np.sin(u(t))*kete1

def D2(t,phi2,b2):
    if t¡T/2.0:
        phi2 = 0
    return np.cos(u(t))*np.cos(v(t))*np.exp(-1j*phi2)*b2 - 1j*np.sin(
        u(t))*kete2 - np.cos(u(t))*np.sin(v(t))*ket4



# Generates the new basis states in the dark-bright basis given a set
    of parameters
def genStates(par):
    chi = par[0]; xi = par[1]; theta = par[2]; phi = par[3];

    c1 = np.cos(theta)
    c2 = np.exp(1j*chi)*np.sin(theta)*np.cos(phi)
    c3 = np.exp(1j*xi)*np.sin(theta)*np.sin(phi)


    if np.isclose(np.abs(c1)**2, 1.0):
        d = ket1
        b1 = ket2
        b2 = -np.exp(1j*xi)*ket3
```

```
49
50      else:
51          N1 = 1/(np.sqrt(1-np.abs(c3)**2))
52          N2 = np.abs(c3)/(np.sqrt(1-np.abs(c3)**2))
53          d = c1*ket1 + c2*ket2 + c3*ket3;
54          b1 = N1* ( -np.conj(c2)*ket1 + c1*ket2 );
55          b2 = N2*(c1*ket1 + c2*ket2 + (c3 - 1/np.conj(c3))*ket3 );
56
57      return d,b1,b2
58
59  # Generates the starting coeffcients required for a given initial
        state
60  def genInitialCoeff(d,b1,b2,initialState):
61      A = np.transpose(np.array([d[2:5],b1[2:5],b2[2:5]]))
62      f = np.matmul(np.linalg.inv(A),initialState)
63
64      return f
65
66  # Returns the a analytical unitary based on the parameters
67  def unitaryGate(d,b1,b2,gamma1, gamma2):
68      return np.outer(d,np.conj(d)) + np.exp(1j*gamma1)*np.outer(b1,np.
            conj(b1)) + np.exp(1j*gamma2)*np.outer(b2,np.conj(b2))
69
70
71  # Steps through one loop and saves the state populations
72  def simulate(d,b1,b2,F,gamma1,gamma2):
73      Prob = []
74      state = []
75      for x in t:
76          X = F[0]*d + F[1]*D1(x,-gamma1,b1) + F[2]*D2(x, -gamma2, b2)
77          state.append(X)
78          pe1 = np.abs(X[0])**2
79          pe2 = np.abs(X[1])**2
80          p1 = np.abs(X[2])**2
81          p2 = np.abs(X[3])**2
82          p3 = np.abs(X[4])**2
83          pa = np.abs(X[5])**2
84          Prob.append([pe1, pe2, p1, p2, p3, pa])
85
86      finalState = np.array(state[-1])
87      return Prob, finalState
88
89
90  # par - parameters = (chi, xi, theta, phi, gamma1, gamma2)
91  # I - inital state - 1x3 normalized vector
92  # returns Prob, U
93  def singleLoop(initialState,par):
94      gamma1 = par[4]; gamma2 = par[5]
95      d,b1,b2 = genStates(par)
96      F = genInitialCoeff(d,b1,b2,initialState)
97      check(d,b1,b2,F)
98      Prob, finalState = simulate(d,b1,b2,F,gamma1,gamma2)
99      U = unitaryGate(d,b1,b2, gamma1, gamma2)
100
101     return Prob, U, finalState
```

```
102
103 # Applies singleLoop() twice, first with par1, then par2
104 def doubleLoop(initialState, par1, par2):
105     prob1, U1, intermedState = singleLoop(initialState, par1)
106     prob2, U2, finalState = singleLoop(intermedState[2:5], par2)
107     U = np.matmul(U2,U1)
108     prob = prob1 + prob2
109
110     return prob, U, finalState
111
112 # Parameters for X-gate
113 def XGate(initialState):
114     par1 = [0, 0, np.pi/4, np.pi/2, 0, np.pi]
115     par2 = [0, 0, np.pi/2, np.pi/4, 0, np.pi]
116     prob, U, finalState = doubleLoop(initialState, par1, par2)
117
118     return prob, U, finalState
119
120 # Parameters for Z-gate
121 def ZGate(initialState):
122     par = [0,0,0,0,2*np.pi/3,4*np.pi/3]
123     prob, U, finalState = singleLoop(initialState,par)
124     return prob, U, finalState
125
126 # Parameters for T-gate
127 def TGate(initialState):
128     par = [0,0,0,0,2*np.pi/9,-2*np.pi/9]
129     prob, U, finalState = singleLoop(initialState,par)
130     return prob, U, finalState
131
132
133 # parameters found via optimization and thus not fancy as for X,Z and
      T
134 def HGate(initialState):
135     par = [6.41010859e-04, 6.55568952e-04, 4.75667128e-01, 7.85362474
        e-01,1.58054108e+00, 1.56302702e+00, 9.81289849e-03,
        3.56878815e-18,1.18743379e+00, 2.15063745e+00, 9.74301696e-17,
         1.56882773e+00]
136
137     prob, U, finalState = doubleLoop(initialState,par[:6],par[6:])
138     return prob, U, finalState
139
140
141 print(""nRunning main...“n")
142 # Analytical qutrit gates (used for comparision/optimization)
143 Tg = np.array([[1,0,0],[0,np.exp(2*np.pi*1j/9),0],[0,0,np.exp(-2*np.
    pi*1j/9)]])
144 Hg = (1/np.sqrt(3))*np.array([[1, 1, 1], [1, np.exp(2*np.pi*1j/3), np
    .exp(4*np.pi*1j/3)], [1, np.exp(4*np.pi*1j/3), np.exp(2*np.pi*1j
    /3)]])
145 Xg = np.array([[0,0,1], [1,0,0], [0,1,0]])
146 Zg = np.array([[1,0,0], [0,np.exp(2*np.pi*1j/3),0], [0,0,np.exp(4*np.
    pi*1j/3)]])
147
148 # Set initialstate and normalize it
```

```
initialState = np.array([1,1,1]);
initialState = initialState/np.linalg.norm(initialState)


prob, U, finalState = HGate(initialState)
#prob, U, finalState = singleLoop(initialState, par1)
#prob, U, finalState = doubleLoop(initialState, par[:6],par[6:])

print("Final states ")
print(f"exact: -np.round(U[2:5,2:5] @ initialState,6) ")
print(f"numerical: -np.round(finalState[2:5],6) ")

plt.figure()
tt = np.linspace(0,len(prob)/len(t),len(prob))
labels = ["pe1","pe2","p1","p2","p3","pa"]
probPlot = plt.plot(tt,prob,'--')
plt.legend(iter(probPlot), labels)
plt.title(f"Probabilities with $“eta$ = -eta ")
plt.xlabel("time")
plt.ylabel("Probability")
plt.axis([0, len(prob)/len(t), 0, 1.1])
plt.grid()

plt.show()
```

A.2 Script B - Robustness test

```
import numpy as np
from scipy.integrate import complexode, solveivp
from scipy.linalg import sqrtm
import matplotlib.pyplot as plt
from collections import defaultdict

T = 1.0 # Period of one loop

# Basis vectors
kete1 = np.array([1, 0, 0, 0, 0, 0])
kete2 = np.array([0, 1, 0, 0, 0, 0])
ket1  = np.array([0, 0, 1, 0, 0, 0])
ket2  = np.array([0, 0, 0, 1, 0, 0])
ket3  = np.array([0, 0, 0, 0, 1, 0])
keta  = np.array([0, 0, 0, 0, 0, 1])
################################

def cot(t):
    return np.cos(t)/np.sin(t)

def u(t):
    return (np.pi/2)*(np.sin(np.pi*t/T))**2

def v(t):
    return eta*(1 - np.cos(u(t)))

```

```
def up(t):
    return ((np.pi**2)/(2*T))*np.sin(2*np.pi*t/T)

def vp(t):
    return eta*up(t)*np.sin(u(t))

def nmean(data):
    result = [sum(x) / len(x) for x in zip(*data)]
    return result


def genStates(par):
    chi = par[0]; xi = par[1]; theta = par[2]; phi = par[3];

    c1 = np.cos(theta)
    c2 = np.exp(1j*chi)*np.sin(theta)*np.cos(phi)
    c3 = np.exp(1j*xi)*np.sin(theta)*np.sin(phi)


    if np.isclose(np.abs(c1)**2, 1.0):
        d = ket1
        b1 = ket2
        b2 = -np.exp(1j*xi)*ket3

    else:
        N1 = 1/(np.sqrt(1-np.abs(c3)**2))
        N2 = np.abs(c3)/(np.sqrt(1-np.abs(c3)**2))
        d = c1*ket1 + c2*ket2 + c3*ket3;
        b1 = N1* ( -np.conj(c2)*ket1 + c1*ket2 );
        b2 = N2*(c1*ket1 + c2*ket2 + (c3 - 1/np.conj(c3))*ket3 );

    return d,b1,b2



# returns (Omega1, Omega2, Omegaa) at time t
def omegas(t,delta):
    if t ==  0.0 or t == T:
        O1 = 0; O2 = 0; Oa = 0;
    else:
        O1 = -2*up(t)
        O2 =  2*(vp(t)*cot(u(t))*np.sin(v(t)) + up(t)*np.cos(v(t)))
        Oa =  2*(vp(t)*cot(u(t))*np.cos(v(t)) - up(t)*np.sin(v(t)))


    return [O1*(1+delta),O2*(1+delta),Oa*(1+delta)]


def Ham(t,b1,b2,phi1,phi2,delta):
    if  t ¡ T/2.0:
        phi1 = 0; phi2 = 0

    O = omegas(t,delta)

    T1 = (O[0]/2)*np.exp(-1j*phi1)*np.outer(np.conj(b1),kete1)
```

```
    T2 = (O[1]/2)*np.exp(-1j*phi2)*np.outer(np.conj(b2),kete2)
    Ta = (O[2]/2)*np.outer(np.conj(keta),kete2)

    H = np.array(T1 + T2 + Ta)
    H = np.array(H + np.conj(np.transpose(H)))

    return H

# Schrodinger equation
def f(t,y,delta,b1,b2,phi1,phi2):
    return -1j*Ham(t,b1,b2,phi1,phi2,delta)@y

# Generate unitary from parameters
def unitaryGate(d,b1,b2,gamma1, gamma2):
    return np.outer(d,np.conj(d)) + np.exp(1j*gamma1)*np.outer(b1,np.
        conj(b1)) + np.exp(1j*gamma2)*np.outer(b2,np.conj(b2))

def Fidelity(p,q):
    return np.abs(np.dot(np.conj(p),q))

# Calculates the fidelities with a given initial condition for a set
    of deltas
def fidLoop(y0,par,method, deltas):
    FLAG = False
    if len(par) ¿ 6:
        FLAG = True
        par1 = par[:6]
        par2 = par[6:]
        phi1 = -par1[-2]; phi2 = -par1[-1]
        phi12 = -par2[-2]; phi22 = -par2[-1]

        d, b1, b2 = genStates(par1)
        d2, b12, b22 = genStates(par2)

        U = unitaryGate(d2,b12,b22,-phi12,-phi22) @ unitaryGate(
            d,b1,b2,-phi1,-phi2)

    else:
        phi1 = -par[-2]; phi2 = -par[-1]
        d, b1, b2 = genStates(par)
        U = unitaryGate(d,b1,b2,-phi1,-phi2)


    fids = []
    for delt in deltas:

        sol1 = solveivp(f,(0,T),y0,method=method,
                        args=(delt, b1,b2,phi1,phi2))
        ynum = np.array(sol1.y[:,-1]); #ynum = ynum/np.linalg.norm
            (ynum)

        if FLAG:
            sol2 = solveivp(f,(0,T),ynum,method=method,
                        args=(delt, b12,b22,phi12,phi22))
```

```
132             ynum = np.array(sol2.y[:,-1]); ynum = ynum/np.linalg.
                    norm(ynum)
133
134         fids.append(Fidelity(yexact[2:5],ynum[2:5]))
135     return deltas, fids
136
137 # Averages fidelity over many initial states
138 def averageFid(par,n):
139     k = 20 # number of points sampled [+k*delta, -k*delta]
140     delta = 0.015 # error size
141     deltas = [x*delta for x in range(-k,k+1)]
142     fids = []
143     for k in range(n):
144             y0 = np.array(np.random.rand(3),dtype="complex")
145             y0 = y0/np.linalg.norm(y0)
146             y0 = np.concatenate(([0.0,0.0],y0,[0.0]),dtype="complex"
                    )
147
148             x,y = fidLoop(y0,par,'BDF', deltas)
149             fids.append(y)
150
151     return x, nmean(fids)
152
153 # Calculates the average fidelity sampled over n states using the
       parameters of parfunc
154 def Fidplot(n, etas,parfunc):
155     par = parfunc()
156     name =  parfunc. name ; name = name[0]
157     print(name)
158
159     global eta
160
161     eta = etas[0]
162     x1,y1 = averageFid(par,n, False)
163     eta = etas[1]
164      ,y2 = averageFid(par,n, False)
165
166     return x1,y1,y2
167
168 ## Gate parameters
169 def Xpar():
170     par1 = [0, 0, np.pi/4, np.pi/2, 0, np.pi]
171     par2 = [0, 0, np.pi/2, np.pi/4, 0, np.pi]
172     return par1 + par2
173
174 def Zpar():
175     par = [0,0,0,0,2*np.pi/3,4*np.pi/3]
176     return par
177
178 def Tpar():
179     par = [0,0,0,0,2*np.pi/9.0,-2*np.pi/9.0]
180     return par
181
182 def Hpar():
```

```
183     par = [6.41010859e-04, 6.55568952e-04, 4.75667128e-01, 7.85362474
            e-01,1.58054108e+00, 1.56302702e+00, 9.81289849e-03,
            3.56878815e-18,1.18743379e+00, 2.15063745e+00, 9.74301696e-17,
             1.56882773e+00]
184     return par
185
186 etas = [0.0, 4.0] # eta values
187 n = 500 # Number of sampled states
188
189 x, Ty1, Ty2 = Fidplot(n,etas,Tpar)
190   , Xy1, Xy2 = Fidplot(n,etas,Xpar)
191   , Hy1, Hy2 = Fidplot(n,etas,Hpar)
192   , Zy1, Zy2 = Fidplot(n,etas,Zpar)
193
194 fig, axs  = plt.subplots(2,2)
195 #gs = fig.addgridspec((2,2), hspace=0)
196 #axs = gs.subplots(sharex=True, sharey=True)
197 axs[0,0].plot(x,Ty1,'*--',label=f"$“eta =$ -etas[0]")
198 axs[0,0].plot(x,Ty2,'*--',label=f"$“eta =$ -etas[1]")
199 axs[0,0].settitle(f"T-Gate")
200 axs[0,1].plot(x,Xy1,'*--',label=f"$“eta =$ -etas[0]")
201 axs[0,1].plot(x,Xy2,'*--',label=f"$“eta =$ -etas[1]")
202 axs[0,1].settitle(f"X-Gate")
203 axs[1,0].plot(x,Hy1,'*--',label=f"$“eta =$ -etas[0]")
204 axs[1,0].plot(x,Hy2,'*--',label=f"$“eta =$ -etas[1]")
205 axs[1,0].settitle(f"H-Gate")
206 axs[1,1].plot(x,Zy1,'*--',label=f"$“eta =$ -etas[0]")
207 axs[1,1].plot(x,Zy2,'*--',label=f"$“eta =$ -etas[1]")
208 axs[1,1].settitle(f"Z-Gate")
209
210 fig.suptitle(f'Fidelity averaged over -n  states')
211 for ax in fig.getaxes():
212     ax.setylim(ymax=1)
213     ax.set(xlabel='$“delta“Omega$', ylabel='Fidelity')
214     ax.labelouter()
215     ax.grid()
216     ax.legend()
217 plt.show()
```