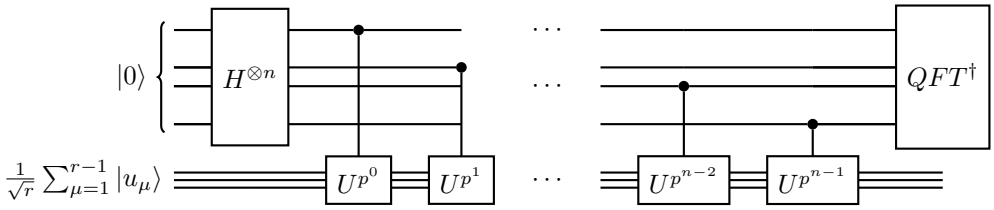
PY402 Final Project

Nick Drda, Grant Sherrill, Drew Connelly
(Dated: November 26, 2023)

I. INTRODUCTION

The most valuable aspect of a quantum computer is its potential to handle exponentially-scaling state spaces during computation. Both quantum and classical computers use fundamental computational units, "bits", which, in classical computing, are assigned a value of either 0 or 1. However, quantum bits–called qubits– are able to take superposition values of 0 and 1 (we discuss the specifics of this in the next section on spinors). When discussing the complexity of a computational problem, we use O ("Big-O") notation. A problem having complexity of order $O(N^3)$, for example, tells us that the time required to perform a calculation that yields an N-bit result will scale according to $t(N) \propto N^3$. For instance, the complexity of the best known algorithm that solves the integer factorization problem is $O(e^{N^{1/3}})$. This is an example of an exponentially-scaling problem, meaning that as N increases, it quickly begins to take too long to be reasonably computed on a classical computer. However, quantum computers can perform this computation using "Shor's Algorithm" (circuit diagram pictured below), which reduces the complexity of this problem to $O(\log(N))$!



The details of how Shor's algorithm works are outside the scope of this report, including the implementation of a quantum fourier transform (QFT) gate, and cleverly-controlled U-gates. For more information on Shor's algorithm, see Ref. [1]. Quantum computers achieve this and other speedups by using the superposition properties of qubits. This means that we can not only express simple states like $|0\rangle$ and $|1\rangle$, but also $\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$, and various combinations of $|0\rangle$ and $|1\rangle$. While taking advantage of these superposition states is not a trivial task, in certain situations (like integer factorization) it can be extremely powerful![1] Furthermore, quantum computers promise efficient simulation of quantum chemistry problems and quantum field theories. These are two fields where the system complexity potentially scales exponentially, making classical simulation extremely difficult.

While quantum computers promise so much, one important step is still needed before this potential is realized; current-day quantum computers are very error-prone due to the low thermal energy needed to disrupt the qubits which exist in relatively-cold energy levels. For this reason the hardware which keeps quantum computers extremely cold and error-correcting algorithms are top priorities for many researchers so that we can achieve fault-tolerant qubits. In this report, we will discuss the basics of quantum computing which are required to begin reading about more advanced topics in the field which may interest you. Primarily we aim to introduce the reader to representations of qubits as spinors, quantum cicuit diagrams and their interpretations, the basics of quantum gates and how to construct them, and finally measurement on a quantum computer.

II. QUBIT STATES AS SPINORS

In quantum mechanics, we define the spin states of a given particle using spinors. A spinor is a two-element column vector used to represent the general state of a particle with spin, taking the following form:

$$\chi = \begin{bmatrix} a \\ b \end{bmatrix} = a\chi_+ + b\chi_-,$$

[1] One can think of a classical computer as only being able to populate the various poles of the Bloch sphere, meaning the χ_+, χ_-, χ_{x+}, χ_{x-}, χ_{y+}, and χ_{y-} states, while a quantum computer can take any state on the entire surface of the Bloch sphere. The Bloch sphere is discussed in more detail in a later section.

where a and b are arbitrary complex constants obeying $|a|^2 + |b|^2 = 1$, and:

$$\chi_+ = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad \chi_- = \begin{bmatrix} 0 \\ 1 \end{bmatrix}.$$

This notation is helpful for understanding quantum computations, since our states can be represented by any possible linear combination of these two basis states, given they obey the normalization requirement. This is in stark contrast to the available space 1 classical bit can span. Taking a look at classical computations, we have 2 ways we can express a state using one bit: 1or0. In this representation we get more information by using the properties of quantum mechanics!

When we are using qubits in quantum computing, we often write them using "bra-ket" notation, where:

$$|0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \quad \text{and} \quad \langle 0| = |0\rangle^\dagger = \begin{bmatrix} 1 & 0 \end{bmatrix},$$

noting that our bra is the hermitian conjugate of our ket state. An important note is this representation treats $\langle a|b\rangle$ as the inner product of state a with state b. Due to normalization of states $\langle a|a\rangle = 1$. Lastly, in quantum computers the two basis states are then:

$$|0\rangle = \chi_+ = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \quad \text{and} \quad |1\rangle = \chi_- = \begin{bmatrix} 0 \\ 1 \end{bmatrix}.$$

Let us consider superpositions of our spin states:

$$|00\rangle, \quad |10\rangle, \quad |01\rangle, \quad |11\rangle.$$

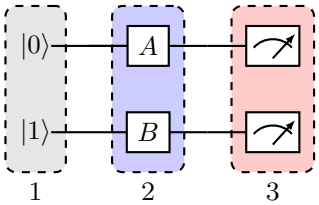
In this case, two qubits have provided us four unique bits of information. This can be extended to a larger amount of spin states, with the ratio of n qubits providing us with 2^n classical bits worth of information. Each additional quibit added to a system is increasingly important as the amount of yield from the combinations of spinors exponentially increases, highlighting the need for companies to develop higher number quibit quantum computers. The increased amount of computational space will be discussed later, highlighting the utility of the Bloch sphere.

III. QUANTUM CIRCUITS & DIAGRAM NOTATION

In classical computation, one may instruct a computer to perform a specific sequence of tasks by writing and executing a script written in a coding language such as Python, C++, Java, etc. The quantum-computational equivalent of the script is a "quantum circuit", and these can be written in multiple languages–Qiskit, Q#, XACC, and so on–but they share a common representation as a "circuit diagram". These diagrams have three key parts:

1. The initial state
2. The operations (gates) performed on the initial state
3. The measurement.

Let us now examine a simple example of a 2-qubit circuit diagram, with each of the above parts higlighted:



Notice that each qubit has its own horizontal line, and the operations performed by the computer on each qubit are thus the symbols which intersect with the qubit's line. Since each qubit is fundamentally a particle in a spinor state[2],

[2] In real digital quantum computers, qubits are often actually in near-spinor states which are engineered to be **very good** approximations of perfect spinors.

we can tell from region 1 that the first (uppermost) qubit is initialized in the state:

$$|0\rangle \equiv \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \chi_{-},$$

while the second qubit is initialized in the orthogonal state:

$$|1\rangle \equiv \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \chi_{+}.$$

Most often, we will see that each qubit is initialized in the $|0\rangle$ state, but sometimes an initial state is analytically known and it may be useful to represent a circuit that starts one or more qubits in a specific ground state before doing any operations.

Next, we see in region 2 that each qubit is acted on by an operator. Each quantum operator is represented by a box with a label. The label indicates the name (and sometimes properies of) the operator, and the box tells us about the nature of the operator. The number of qubit lines that come into and out of the box tell us how many qubits are acted upon by the operator. In circuit diagram notation, the operators act upon the state to their left, meaning that the operator A acts on the first qubit to produce $A|0\rangle$, and B on the second qubit to product $B|1\rangle$. Since there are no gates which (entangle) act on both qubits, we can use tensor products to more conveniently write out the mathematics of this system. In quantum circuit diagrams, tensor act vertically between the lines, meaning that this diagram can be interpreted as an instruction to perform the following calculation:

$$\begin{array}{l} |0\rangle \text{—}\boxed{A}\text{—} \\ |1\rangle \text{—}\boxed{B}\text{—} \end{array} \quad \begin{matrix} A \\ = \otimes \\ B \end{matrix} \begin{pmatrix} |0\rangle \\ \otimes \\ |1\rangle \end{pmatrix} = \begin{pmatrix} A|0\rangle \\ \otimes \\ B|1\rangle \end{pmatrix}, \tag{1}$$

where $\otimes$ denotes the tensor product (specifically, the Kronecker product)[3].

Finally in region 3 when we measure the system, our calculations in Eqn. 1 tell us that we will find that the first (uppermost) qubit is in the state $A|0\rangle \equiv |c\rangle$, and the second qubit is in the state $B|1\rangle \equiv |d\rangle$. Thus our measurement (assuming no error) will yield

$$|\psi\rangle = \begin{pmatrix} |c\rangle \\ \otimes \\ |d\rangle \end{pmatrix}.$$

If we were to measure only the first qubit, we would get $|\psi_0\rangle = |c\rangle$, and if we were to measure only the second qubit, we would find $|\psi_1\rangle = |d\rangle$. This is a result of the fact that the qubits were computed seperately, as there were no 2-qubit gates in this circuit. If there were any 2-qubit gates, the states of the two outcomes may be mathematically linked such that measuring one informs us about the state of the other. Now that the diagram notation is understood, we need to explore the properties of the operators which populate region 2 of these diagrams, so that we can begin to perform calculations!

---

[3] The kronecker product is often informally "defined" as a matrix operation such that $(A \otimes B)(C \otimes D) = AC \otimes BD$. This is equivalent to the distribution step I perform in Eqn. 1, however I prefer to write tensor products like this vertically to make its relation to circuit notation clear, and make this "mixed-product" property of $\otimes$ intuitive.

IV. BLOCH SPHERE & SINGLE-QUBIT GATES

A. The Bloch Sphere

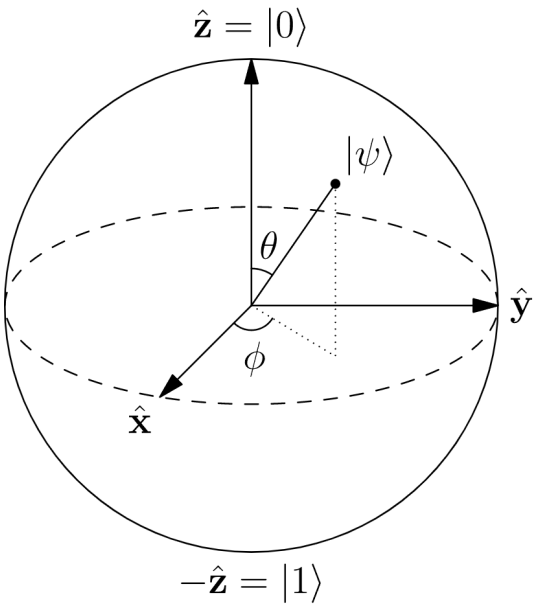


FIG. 1: A Diagram of the Bloch Sphere

This is the Bloch sphere, a common diagram to represent a quantum state and how successive transformations change that state. While it is not shown here, this is a unit sphere, meaning it has radius 1. From the formalism of quantum mechanics, we know states must be normalized, so each state has magnitude 1. Thus any possible state can be drawn living on a sphere of radius 1, which the Bloch sphere accomplishes. An important characteristic of this diagram is the placement of our three axes, and the corresponding states. The Z-axis corresponds to our χ_+ and χ_- states. Next our orthogonal X-axis corresponds to the superposition states: $\chi_{x+} = \frac{1}{\sqrt{2}}[|0\rangle + |1\rangle]$ and $\chi_{x-} = \frac{1}{\sqrt{2}}[|0\rangle - |1\rangle]$. Our final orthogonal axis adds in a phase between the superimposed $|0\rangle$ and $|1\rangle$ states: $\chi_{y+} = \frac{1}{\sqrt{2}}[|0\rangle + i\,|1\rangle]$ and $\chi_{y-} = \frac{1}{\sqrt{2}}[|0\rangle - i\,|1\rangle]$. Thus when we say we rotate by π around the Z-axis in the χ_+ state, it is better to think of rotating the axes of the bloch sphere rather than the state itself. This is because the state χ_+ itself does not move under rotation about the Z-axis, but any successive transformation will send us in a different direction down the side of the sphere, as the orientation of the bloch sphere as a whole has changed. The Bloch sphere itself aids in processing and predicting where states should land, especially using R_x, R_y and R_z gates (which are discussed at the end of this section in detail). The single-qubit gates introduced in our next section can be visualized by moving the dot representing the state $|\psi\rangle$ to a different point on the Bloch sphere.

B. Single-Qubit Gates

In classical computing we consider logical gates which take binary inputs, interpret them, and arrive at an output. In a quantum computer, we no longer have a binary input which can be processed by the gate, rather we have a spin state. We are able to operate on this spin state using a different type of operation: a matrix. Thinking of matrices as the quantum-computational equivalent of logical gates allows us to apply our knowledge of linear algebra to engineer the correct circuits to run on a quantum device.

With this in mind, we can start constructing gates. Consider an operation which takes in a state (spin-up or spin-down) and returns the other state. This is called the X gate, as it rotates the spin state around the x-axis. There are many different ways to represent this, but starting with the matrix representation:

$$X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} = \sigma_x$$

where we can see:

$$X\chi_+ = \begin{bmatrix}0 & 1\\ 1 & 0\end{bmatrix}\begin{bmatrix}1\\0\end{bmatrix} = \begin{bmatrix}0\\1\end{bmatrix} = \chi_-$$

which fits our expected outcome, it takes in a spin state and returns the other state. We have seen this form before, called the Pauli spin matrices.
But we can write this more concisely using knowledge of the formalism of quantum mechanics:

$$X = |1\rangle\langle 0| + |0\rangle\langle 1|\,.$$

This representation may not look the same, but it performs the same task as the matrix above. If we input the spin up state:

$$(|1\rangle\langle 0| + |0\rangle\langle 1|)\,|1\rangle = |1\rangle\langle 0|\,|1\rangle + |0\rangle\langle 1|\,|1\rangle = 1\,|0\rangle + 0\,|1\rangle = |0\rangle\,,$$

so we have taken our spin up state and returned the spin down state. This representation is more intuitive when it comes to constructing the operators we want, and will be used from here on out for spin matrices. Spin matrices can be reconstructed from the bra-ket notation like so:

$$|1\rangle\langle 0| + |0\rangle\langle 1| = \begin{bmatrix}0\\1\end{bmatrix}\begin{bmatrix}1 & 0\end{bmatrix} + \begin{bmatrix}1\\0\end{bmatrix}\begin{bmatrix}0 & 1\end{bmatrix} = \begin{bmatrix}0 & 1\\0 & 0\end{bmatrix} + \begin{bmatrix}0 & 0\\1 & 0\end{bmatrix} = \begin{bmatrix}0 & 1\\1 & 0\end{bmatrix} = X$$

We have a similar appearing Y gate as:

$$Y = -i\,|0\rangle\langle 1| + i\,|1\rangle\langle 0| = \begin{bmatrix}0 & -i\\ i & 0\end{bmatrix},$$

where a factor of -1 was attained to allow us to return to the original state after acting on a state twice.
And finally a gate in our Z-axis:

$$Z = -\,|1\rangle\langle 1| + |0\rangle\langle 0| = \begin{bmatrix}1 & 0\\0 & -1\end{bmatrix}.$$

While these sigma matrices are great, we can't yet perform arbitrary calculations (We will see below that these can be generalized to allow us to do just that). Using these we are only able to rotate in π radians step sizes around any of the three axes. If we want to end up somewhere other than in a spin-up or spin-down state (in order to take advantage of the Hilbert space's complexity), we must go beyond spin matrices. One simple next step is known as the Hadamard gate. Let us consider landing on the x-axis, given by the state

$$|\psi\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$$

This state is a $\pi/2$ rotation into the x basis. But we know that if we rotate from χ_- we end up in a different state than from χ_+, the negative in fact. Thus we can write

$$H = \frac{1}{\sqrt{2}}[|0\rangle\langle 0| + |0\rangle\langle 1|] + \frac{1}{\sqrt{2}}[|1\rangle\langle 0| - |1\rangle\langle 1|] = \frac{1}{\sqrt{2}}\begin{bmatrix}1 & 1\\1 & -1\end{bmatrix}.$$

With this we achieve the ability to start in a spin state and end on an orthogonal basis. With this now we can consider consequences of the gates we have been constructing. We may not be able to measure the X basis of our system, but we do not need to! Consider the HZH transformation:

$$HZH = \frac{1}{2}\begin{bmatrix}1 & 1\\1 & -1\end{bmatrix}\begin{bmatrix}1 & 0\\0 & -1\end{bmatrix}\begin{bmatrix}1 & 1\\1 & -1\end{bmatrix} = \frac{1}{2}\begin{bmatrix}1 & 1\\1 & -1\end{bmatrix}\begin{bmatrix}1 & 1\\-1 & 1\end{bmatrix} = \begin{bmatrix}0 & 1\\1 & 0\end{bmatrix} = X$$

which is familiar to our X-gate. Thus we can make measurements in our Z basis without losing information.
Above we have considered π and $\pi/2$ rotations, which leads to the question of if we can perform arbitrary rotations, which we can! Our bottom left matrix element in our Z matrix was -1, which correlates to a π rotation. For our given angle $e^{i\pi} = -1$, so for an arbitrary rotation of θ, we might guess that we replace this value with $e^{i\theta}$, like so:

$$P(\theta) = \begin{bmatrix}1 & 0\\0 & e^{i\theta}\end{bmatrix},$$

from which we can recover arbitrary Z-rotations. Many quantum computers allow implementations of many standard forms of this gate. The first of which is the identity matrix, a familiar identity element of our basis:

$$P(2\pi) = I = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}.$$

And then quarter- and eighth-turns around our z-axis:

$$P(\pi/2) = S = \begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix}, \quad P(\pi/4) = T = \begin{bmatrix} 1 & 0 \\ 0 & \frac{1}{2}(1+i) \end{bmatrix}$$

Just as the P-gate is the most general form for rotation about the Z-axis, we can generalize rotations about the X, Y, and Z axes with $R_x, R_y,$ and R_z gates respectively ($R_z(\theta)$ is equivalent to $P(\theta)$, as it differs only by a global phase). In matrix form:

$$R_x(\theta) = \begin{bmatrix} \cos(\theta/2) & -i\sin(\theta/2) \\ -i\sin(\theta/2) & \cos(\theta/2) \end{bmatrix} \quad R_y(\theta) = \begin{bmatrix} \cos(\theta/2) & -\sin(\theta/2) \\ \sin(\theta/2) & \cos(\theta/2) \end{bmatrix} \quad R_z(\theta) = \begin{bmatrix} e^{-i\theta/2} & 0 \\ 0 & e^{i\theta/2}. \end{bmatrix}$$

Where we can apply transformations we know, and recover the gates we have seen before. For example, if I want a gate that makes a π rotation about the X-axis:

$$R_x(\pi) = \begin{bmatrix} \cos(\pi/2) & -i\sin(\pi/2) \\ -i\sin(\pi/2) & \cos(\pi/2) \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}.$$

It turns out that we can then construct any arbitrary rotation via a combination of these three gates. In the next section we prove directly that this can be done and we provide a formula for doing so.

C. Extension: Single-Qubit Rotations are Spanned by Sigma Matrices

Here we will take a moment to depart from the text of the Qiskit Textbook in order to motivate the importance of the R_x, R_y, and R_z gates by examining the implementation of an arbitrary single-qubit gate on a quantum computer.

My claim is the following:

Any single-qubit gate can be implemented via a product of three rotation gates (R_x, R_y, and R_z).

I will show this by direct proof, or in other words by finding the angles required as a function of the defining parameters of the gate we want to implement. This is a concept that is analogous to finding Euler angles in Classical Mechanics, where we were able to track the arbitrary rotation of an object through three coordinate-axis rotations.

Proof: We begin by identifying the matrix form of an arbitrary single-qubit gate "A" from the unitarity requirement on such a gate.

By writing the matrix A in its most general form as a 2×2 matrix of complex numbers, we see:

$$AA^\dagger = I \implies \begin{bmatrix} ae^{i\alpha} & be^{i\beta} \\ ce^{i\gamma} & de^{i\delta} \end{bmatrix} \begin{bmatrix} ae^{-i\alpha} & ce^{-i\gamma} \\ be^{-i\beta} & de^{-i\delta} \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \implies \begin{bmatrix} a^2+b^2 & ace^{i(\alpha-\gamma)} + bde^{i(\beta-\delta)} \\ ace^{-i(\alpha-\gamma)} + bde^{-i(\beta-\delta)} & c^2+d^2 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix},$$

which implies the following system of equations:

$$\begin{aligned} a^2 + b^2 &= 1 \\ c^2 + d^2 &= 1 \\ ace^{-i\omega_1} &= -bde^{-i\omega_2} \\ ace^{i\omega_1} &= -bde^{i\omega_2}, \end{aligned}$$

where $\omega_1 \equiv \alpha - \gamma$, and $\omega_2 \equiv \beta - \delta$. Before we can make any progress, we examine the other implication of unitarity, which is that $A^\dagger A = I$:

$$A^\dagger A = I \implies \begin{bmatrix} ae^{-i\alpha} & ce^{-i\gamma} \\ be^{-i\beta} & de^{-i\delta} \end{bmatrix} \begin{bmatrix} ae^{i\alpha} & be^{i\beta} \\ ce^{i\gamma} & de^{i\delta} \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \implies \begin{bmatrix} a^2+c^2 & abe^{i(\beta-\alpha)} + cde^{i(\delta-\gamma)} \\ abe^{-i(\beta-\alpha)} + cde^{-i(\delta-\gamma)} & b^2+d^2 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}.$$

This gives us a similar system of equations:

$$\begin{aligned} a^2 + c^2 &= 1 \\ b^2 + d^2 &= 1 \\ abe^{i\omega_3} &= -cde^{i\omega_4} \\ abe^{-i\omega_3} &= -cde^{-i\omega_4}, \end{aligned}$$

with $\omega_3 \equiv \beta - \alpha$ and $\omega_4 \equiv \delta - \gamma$.

Note: Interestingly enough, we can take this moment to notice that we can interpret $a^2 + b^2 = a^2 + c^2 = c^2 + d^2 = b^2 + d^2 = 1$ as telling us that each column **and** each row of A must be a valid (normalized) quantum state! (With columns interpreted as kets, and rows as bras) This hints to us that unitary matrices can be considered as norm-preserving changes of bases over complex fields.

Now we simplify the problem. First, we notice that $a^2 + c^2 = 1 = a^2 + b^2$ gives us $b^2 = c^2 \implies \boxed{b = c}$ (since b and c are radii in the complex plane, which must be positive). Then, we have $a^2 + b^2 = 1 = b^2 + d^2$ which gives us $a^2 = d^2 \implies \boxed{a = d}$. Therefore, we have reduced the coefficients $a,\ b,\ c,\ d$ to a, b with $a^2 + b^2 = 1$. This can be convieniently written as $d = a \equiv \cos(\theta)$ and $c = b \equiv \sin(\theta)$.

So far we have reduced the arbitrary matrix A to the following form:

$$\begin{bmatrix} \cos(\theta)e^{-i\alpha} & \sin(\theta)e^{-i\gamma} \\ \sin(\theta)e^{-i\beta} & \cos(\theta)e^{-i\delta} \end{bmatrix}.$$

Next we use the final property of single-qubit gates, which is that they have determinant 1 (the combined requirements of unitarity and determinant 1 define the matrix group SU(2), which is commonly used by physicists to denote the set of all single-qubit rotations).

$$\begin{vmatrix} \cos(\theta)e^{-i\alpha} & \sin(\theta)e^{-i\gamma} \\ \sin(\theta)e^{-i\beta} & \cos(\theta)e^{-i\delta} \end{vmatrix} = 1 \implies \cos^2(\theta)e^{-i(\alpha+\delta)} - \sin^2(\theta)e^{-i(\gamma+\beta)} = 1$$

We can see from the form of this equation that it will only generally be solved when $\boxed{\alpha = -\delta}\ (\text{mod } 2\pi)$ and $\boxed{\gamma = -\beta + \pi}\ (\text{mod } 2\pi)$. With these constraints it becomes $\cos^2(\theta) + \sin^2(\theta) = 1$ ✓.

Thus, we substitute $\alpha = -\delta$ and $\gamma = -\beta + \pi$, and we can finally see the simplest form of our arbitrary single-qubit gate:

$$\begin{bmatrix} \cos(\theta)e^{-i\alpha} & \sin(\theta)e^{-i\gamma} \\ -\sin(\theta)e^{i\gamma} & \cos(\theta)e^{i\alpha} \end{bmatrix}$$

However the proof is not complete until we show that we can use R_x, R_y, and R_z operators to create this arbitrary rotation! For the sake of brevity I will simply show the formula for applying these rotations in order to get the arbitrary gate, as a function of θ, α, and γ:

$$\mathrm{R}_z(\alpha - \gamma) \cdot \mathrm{R}_y(-2\theta) \cdot \mathrm{R}_z(\gamma + \alpha) = \begin{bmatrix} \cos(\theta)e^{-i\alpha} & \sin(\theta)e^{-i\gamma} \\ -\sin(\theta)e^{i\gamma} & \cos(\theta)e^{i\alpha} \end{bmatrix} = A.$$

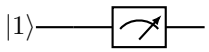
With the notation that $\mathrm{R}_\mu(\phi) = \exp\left(-i\frac{\phi}{2}\sigma_\mu\right)$. This concludes our proof, as given any arbitrary single-qubit gate, we have an algorithm for decomposing it into a product of R_z and R_y gates! ■

It turns out that the choice of Z and Y rotations is arbitrary, so we could derive a similar formula for doing the same with Z and X if we wanted, which is the typical decomposition used by most quantum-computing researchers. This proof also sheds light on the reason behind the importance of the σ_x, σ_y, and σ_z matrices, as they are the minimum requirement to perform any single-qubit rotation.

V. MEASUREMENTS OF QUANTUM GATES & PROJECTION OPERATORS

As we have seen above, our quantum circuits have measurements blocks, and when called they measure the value of the state. But what exactly are they, and how can we represent this measurement mathematically? It is simply a projection operator of the state we want to measure. To be able to perform computations with our qubits, we want to know the probability our state is in one of our bit states. Our projection operators of choice are then our base states, 0 and 1, or spin-up and spin-down. There is not a similar classical analog, this would not be needed as our measurements tell us all of the information present in the computer. However with a quantum computer we can't predict what specific state we will be in when our qubits are entangled, only the probability we are in each state.
In our quantum system, we cannot guarantee that we will always be able to measure the true state of a qubit due to unavoidable sources of error. What we can count on is that the expected value of an operator will likely converge to its true value in the ensemble (we can always take more samples).

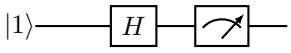
Let us consider the initial state $\begin{bmatrix}1\\0\end{bmatrix}$. Then after some time we measure our unperturbed system. A diagram of this circuit would be as follows:



We want to know what the probability of this state being in spin-up or spin-down is, so we use our projection operators:

$$P(\chi_+) = |\langle\chi_+|\Psi\rangle|^2 = |\langle\chi_+|\chi_+\rangle|^2 = \langle\chi_+|\chi_+\rangle\langle\chi_+|\chi_+\rangle = 1^2 = 1$$
$$P(\chi_-) = |\langle\chi_-|\Psi\rangle|^2 = |\langle\chi_-|\chi_+\rangle|^2 = \langle\chi_-|\chi_+\rangle\langle\chi_+|\chi_-\rangle = 0^2 = 0$$

So if we consider our quantum circuit shown above we see we can only measure our spin-up state. Which is something we would have expected, as no changes were made to the system. This is the idealized case however where the physical quantum computer will not spontaneously change, which is something we cannot guarantee.
Less trivially consider the circuit where we have our spin-up state and apply the Hadamard gate.



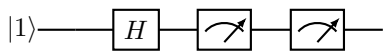
We know we end up on the X-axis where we have both components in the spin-up and spin-down direction, given by the state:

$$|\Psi\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle).$$

Once again we measure both states along our basis states $|0\rangle$ and $|1\rangle$. In terms of projection operators our measurement appears as:

$$P(\chi_+) = |\langle\chi_+|\Psi\rangle|^2 = |\langle\chi_+|\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)\rangle|^2 = \langle\frac{1}{\sqrt{2}}(\langle 0| - \langle 1|)|\chi_+\rangle\langle\chi_+|\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)\rangle = \frac{1}{\sqrt{2}}\cdot\frac{1}{\sqrt{2}} = 1/2$$
$$P(\chi_-) = |\langle\chi_-|\Psi\rangle|^2 = |\langle\chi_-|\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)\rangle|^2 = \langle\frac{1}{\sqrt{2}}(\langle 0| - \langle 1|)|\chi_-\rangle\langle\chi_-|\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)\rangle = \frac{-1}{\sqrt{2}}\cdot\frac{-1}{\sqrt{2}} = 1/2.$$

So our only possible measurements are our two basis vectors! Each with equal probability. If we consider the Bloch sphere representation, our initial state points perpendicular to the Z-axis, meaning that it has equal parts χ_+ and χ_- contribution. For this reason, our initial state's projections onto χ_+ and χ_- have the same magnitude.
This should not be too unfamiliar, as we have been using projection operators all year to determine the probability of finding a certain state. This presents some slight issues when considering our total set of states we can be in. We considered states where we were clearly in a spin-up or spin-down state and we projected directly onto the Z-axis. This was shown by the probabilities of measuring the two states adding to 1. In a more physical system, we may not be in a simple scenario where we can clearly see which two states we can possibly measure. For example, consider the infinite square well, which has an infinite number of measurable states, where the measurement probabilities still sum to 1. To create projection operators for each state would be impossible, and we can never numerically find the coefficient for each state due to the infinite numerical precision this would require.
Lastly, we have one more thing from our formalism to discuss: multiple measurements. We mention above that the quantum computer must fully complete the circuit and reset before making another measurement. This is due to the collapse of the state. If we place two measurements inside of our quantum circuit we see quickly how collapse causes our system to behave strangely. Consider this quantum circuit:



Where we have a spin-up state, a Hadamard gate, and two measurements. After our gate, the first measurement has a 50/50 chance of being either χ_+ or χ_-. When we make the measurement, our formalism requires we collapse into one of our two states. At the second measurement, the wave function's collapse ensures that we will get the same result as our first measurement. For this reason, there is not much value in performing repeated measurement in this way on a quantum computer, as no more unique information is gained, even in an ideal system. This can be used in physical systems to ensure the state isn't spontaneously changing.

References:

General Reference Material: Qiskit Textbook Chapters 1 and 2. Link

[1] **Shor's algorithm** - Qiskit Textbook Chapter 3, Section 7. Link

Participants:

Drew Connelly:

1. Section III (Quantum Circuits & Diagram Notation)
2. Extension (Single-Qubit Rotations are Spanned by Sigma Matrices)
3. Document formatting, proofreading, and edits to all sections

Nick Drda:

1. Section I (Introduction)
2. Section II (Qubit States as Spinors)

Grant Sherrill:

1. Section IV (Bloch Sphere & Single-Qubit Gates)
2. Section V (Measurements of Quantum Gates & Projection Operators)
3. Proofreading and edits to all sections